Ellipsoidal Polystyrene Particles

–

Advances in Manufacturing, Assembly and Applications

Dissertation

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

an der

Fakultät für Biologie, Chemie und Geowissenschaften

der Universität Bayreuth

vorgelegt von

Dominik Michael Benke

geboren in Göppingen

Aalen, 2024

Die vorliegende Arbeit wurde in der Zeit von Juli 2018 bis März 2024 in Bayreuth am Lehrstuhl Physikalische Chemie I unter Betreuung von Herrn Professor Dr. Markus Retsch angefertigt.

Vollständiger Abdruck der von der Fakultät für Biologie, Chemie und Geowissenschaften der Universität Bayreuth genehmigten Dissertation zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften (Dr. rer. nat.).

Form der Dissertation:	Monographie
Die Dissertation wurde eingereicht am:	02.04.2024
Zulassung durch die Prüfungskommission:	10.04.2024
Wissenschaftliches Kolloquium:	14.10.2024

Amtierender Dekan:	Prof. Dr. Cyrus Samimi

Prüfungsausschuss:

Prof. Dr. Markus Retsch	(Gutachter)
Prof. Dr. Anna Schenk	(Gutachterin)
Prof. Dr. André Gröschel	(Vorsitz)
Prof. Dr.-Ing. Holger Ruckdäschel	(Beisitz)

Die vorliegende Arbeit ist als Monographie verfasst.
/ The presented thesis is written as a monography.

Teile der Arbeit sind bereits in folgenden Publikationen veröffentlicht.
/ Some parts of this thesis are already published as the following publications:

Prolate Spheroidal Polystyrene Nanoparticles: Matrix Assisted Synthesis, Interface Properties, and Scattering Analysis

Dominik Benke [†a], Tanja Feller [†a], Marcel Krüsmann [†b], Anna Neuhöfer [a], Friederike Ganster [a], Matthias Karg [b], Markus Retsch [a]
([†] contributed equally to this work, [a] Universität Bayreuth, [b] Universität Düsseldorf)

Veröffentlicht / published:	Nov. 2023
Journal:	*Soft Matter*
Herausgegeber / publisher:	*Royal Society of Chemistry*
DOI:	10.1039/d3sm01002c

Die Publikation fasst hauptsächlich die Ergebnisse aus Kapitel II.2 und II.3 zusammen.
/ This publication summarizes in general the results of Chapter II.2 and II.3.

Die Publikation wird in der Arbeit als Literatur mit der Referenznummer [1] geführt.
/ The publication will have the literature reference number [1].

Float-Cast Microsieves with Elliptical Pores

Nadine Schwaar [c], Dominik Benke [a], Markus Retsch [a], Werner A. Goedel [c]
([a] Universität Bayreuth, [c] Technische Universität Chemnitz)

Veröffentlicht / published:	Oct. 2024
Journal:	*Langmuir*
Herausgegeber / publisher:	*American Chemical Society*
DOI:	10.1021/acs.langmuir.4c01232

Die Publikation entstand durch meine Kooperation mit der TU Chemnitz (Chapter VI).
/ This publication emerged from my cooperation with the TU Chemnitz (Chapter VI).

Die Publikation wird in der Arbeit als Literatur mit der Referenznummer [2] geführt.
/ The publication will have the literature reference number [2].

Zusammenfassung

In dieser Doktorarbeit wird die Herstellung, Analyse und zweidimensionale Anordnung von ellipsoiden kolloidalen Polystyrol-Partikeln präsentiert. Als Grundlage der Herstellung der Ellipsoiden wird ein seit 1966 literaturbekannter uniaxialer Streckprozess von monodispersen kugelförmigen Partikeln verwendet. Dieser Prozess, inklusive der Synthese monodisperser kugelförmiger Partikel, wird zunächst neu ausgearbeitet und grundlegende Eigenschaften der resultierenden Partikel werden definiert. Im nächsten Schritt wird die Herstellung der Partikel in den Gramm-Bereich skaliert. Hierzu wird eine Bandrakel und eine Rollenstreckapparatur aufgebaut und eingestellt. Die Parameter sowie die Lösungen zu verschiedenen aufgetretenen Herausforderungen werden dabei detailliert dargestellt. Zur Beschleunigung der Partikelanalyse wird ein per Python programmiertes Bildauswertungsprogramm (mit grafischer Benutzerschnittstelle) vorgestellt. Dies ist nötig geworden, da die Analyse der einzelnen hergestellten Partikelproben mit dem vorliegenden wissenschaftlichen Standard-Bildauswertungsprogramm Fiji/ImageJ sehr zeitintensiv ist. Durch das neue Bildauswertungsprogramm ist es möglich, Bilder mit tausenden Partikeln, wie sie zum Beispiel bei Partikelmonolagen vorliegen, zeitnah auszuwerten sowie statistische Betrachtungen durchzuführen. Bei Untersuchungen der dreidimensionalen Form der gestreckten Partikel per Seriellem-Blockschnitt-Verfahren im Elektronenmikroskop wird dann eine Methode zur Darstellung und Evaluierung von Polymerpartikeln in einer Polymermatrix entwickelt. Die dreidimensionalen Daten müssen dabei durch ein eigens dafür erstelltes Pythonskript von auftretenden Verzerrungen korrigiert werden. Im weiteren Verlauf der Thesis werden die ellipsoiden Partikel bei der Anordnung zu zweidimensionalen Partikellagen verwendet. Hierbei werden zwei verschiedene Methoden der Selbstanordnung auf Wasseroberflächen aufgezeigt. Die resultierenden schwimmenden, dichten Partikelmonolagen können dann durch einen literaturbekannten Transferprozess in offen-gepackte Monolagen überführt werden. Manche Partikel zeigen dabei trotz Anisotropie ein kristallines Verhalten. Durch einen mechanischen Reib-Prozess und damit durch eine erzwungene Anordnung von Partikeln kann gezeigt werden, dass sich die hergestellten Partikel auch sehr großflächig anordnen lassen. Die so erhaltenen dichten Monolagen werden als Nächstes durch einen Substrat-Streck-Prozess in offen-gepackte Monolagen mit definierten Partikelpositionen überführt. Durch die Neukombination der verschiedenen bisher verwendeten Methoden ist es möglich, einen Prozess zu entwickeln, der zu offen-gepackten kristallinen Monolagen bestehend aus anisotropen Partikeln mit gleicher Orientierung führt. Zum Abschluss der Arbeit wird auf ein Kooperationsprojekt eingegangen, bei dem die Partikel in

Membranen eingebettet werden, um diese zukünftig, nach der Umwandlung in Mikrosiebe, bei Wasserfiltationsexperimenten zu verwenden.

Die vorliegende Doktorarbeit behandelt somit verschiedene Projekte auf dem Gebiet der Prozess- und Methodenentwicklung und vereint dabei unterschiedliche wissenschaftliche Disziplinen (Chemie, Physik, Ingenieurswissenschaften, Statistik) sowie computergestützte Datenauswertungen.

Die Thesis wurde als Monographie in englischer Sprache verfasst.

Summary

The scope of this thesis is the manufacturing, analysis and two-dimensional assembly of ellipsoidal colloidal polystyrene particles. The synthesis of the ellipsoids is based on a since 1966 literature-known uniaxial stretching process of monodisperse polymer particle spheres. The process, including the synthesis of the origin particles, will be discussed and basic properties of the resulting particles are shown. The next step shown is the scaling up of the particle production to several gram. Therefore, an automated foil doctor blade device and a roll-stretching device are built up and adjusted. The parameters and solutions to some challenges that occurred are displayed in detail. For the upscaling of the particle analysis on microscope images, a python-programmed image evaluation tool (with GUI) is implemented. This is necessary as the analysis of the several manufactured particle samples with the given scientific image evaluation tool Fiji/ImageJ is very time-consuming. With the new image evaluation tool it is possible to evaluate images with more than thousand particles, e.g. from particle monolayers, within seconds. Additionally, the program is capable to do some statistical analysis. Using the serial block face imaging method for three-dimensional investigations on the particle shape, a method for imaging and evaluating polymer particles in a polymer matrix is developed. Thereby, the 3D image data stacks need to be corrected from distortion using a self-developed procedure and a python script. In the proceedings of the thesis, the ellipsoidal particles are used for assemblies into 2D particle layers. Two different methods of particle self-assembly on water surfaces are applied. The resulting floating dense-packed particle monolayers are then transferred utilizing a literature-known process to open-packed particle monolayers. Some of the anisotropic particles show crystalline lattices. By utilizing a mechanical rubbing process (forced assembly), the particles can be packed and ordered onto a very large area. The ordered dense-packed particle monolayers are transferred by an elastomer substrate stretching process to open-packed particle monolayers with well-defined lattice positions. Combining the previously used methods in a different order into a new process, it is possible to make open-packed crystalline monolayers from anisotropic particles of same orientation over large areas. In the end of the thesis, findings related to a cooperation project are shown. In the cooperation, the ellipsoidal particles are embedded into membranes and, after converting them to microsieves, they can be used in water filter applications in the future.

The here presented PhD thesis is covering different projects on the field of process and method development. It hereby combines different scientific disciplines, like chemistry, physics, engineering, statistics and computer driven data evaluation.

The thesis was written as a monography in English.

Table of Content

List of Conference Contributions

15th Zsigmondy Colloquium 2019,	Dresden, Germany	– poster contribution
GDCh Wissenschaftsforum Chemie 2019,	Aachen, Germany	– poster contribution
Particle Based Materials Symposium 2019,	Ulm, Germany	– poster contribution
Bayreuther Polymer Symposium 2019,	Bayreuth, Germany	– poster contribution
16th Zsigmondy Colloquium 2020,	Düsseldorf, Germany	– poster contribution

Further on-site contributions were not possible due to the COVID19 pandemic (in Germany 2020 – 2021).

Participation at funding aquisition

Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)

Project number 446333091: Float-Cast Microsieves with Elliptical Pores
Cooperation with N. Schwaar and Professor W. A. Goedel, Physical Chemistry,
TU Chemnitz, cooperation timespan 2020 – 2021,
funding for my position: 01.09.2020 – 31.05.2021 (9 months)

→ See Chapter VI

List of Contributing Students

The following students were supervised by me during the time of my PhD research at the chair of Physical Chemistry I. While the research assistants (undergraduate students) helped me on my main research projects, the interns had individual side projects contributing mostly to my research as indicated on the respective paragraphs.

Teresa Mauerer	– research assistant 2018 – 2019
Friederike Ganster	– research assistant 2019 – 2021 and Bachelor Thesis 2021
Antonia Moßbauer	– research assistant 2020 – 2021
Justus Sanders	– research assistant 2021 – 2022
Pascal Reiß, B.Sc.	– internship summer term 2021
Tobias Scheibel, B.Sc.	– internship winter term 2021/2022

List of Abbreviations

AA	Acrylic acid
AF4	Asymmetric-flow field-flow fractionation
AIBA	2,2'-azobis(isobutyramidine)hydrochloride; radical polymerization starter
AIBN	2,2'-azobis(isobutyronitril); radical polymerization starter
AR	Aspect ratio (longer axis/shorter axis)
AR_{waist}	Aspect ratio of particle waist (calculated by Amira's *Breadth3d*/*Thickness3d*)
AsB	Angle selective backscattered SEM detector
BCA	Butyl cyano acrylate
BSE	Back scattered electrons
cp	close-packed; used for particle lattice description
CsCl	Cesium chloride
CV	Coefficient of variation, particle size variation of particle batch
DLS	Dynamic light scattering
DSC	Differential scanning calorimetry
EsB	Energy selective backscattered SEM detector
ETD	*EVERHART-THORNLEY*-Detector
FFF	Field-flow fractionation
FFT	Fast *FOURIER* transform of an image by Fiji/ImageJ
$g(r)$	Pair correlation function
$\mathrm{g}_2^\theta(\mathrm{r})$	Angular correlation function
$HCl_{(aq)}$	Hydrochloric acid
hcp	Hexagonal close-packed; used for particle lattice description
iPrOH	Iso-propanol, 2-propanol
IR	Infrared (spectroscopy)
KPS	Potassium persulfate; radical polymerization starter
LSM	Optical laser-scanning microscope (Olympus LEXT)
MALS	Multi angle (laser) light scattering device
mDSC	Modulated differential scanning calorimetry
MTC	2-Methacryloxyethyl-trimethylammoniumchloride
$NaOH_{(aq)}$	Sodium hydroxide solution
PBCA	Polymer of butyl cyano acrylate
PDMS	Polydimethylsiloxane
PS	Polystyrene

PSS	Para styrene sulfate (4-Styrenesulfonic acid sodium salt hydrate)
PS+	Polystyrene particles with co-monomer MTC (positively charged)
PS-PSS	Polystyrene particles with co-monomer PSS (negatively charged)
PS-AA	Polystyrene particles with co-monomer AA (negatively charged)
PVA	Polyvinyl alcohol; water-soluble polymer
PVP	Polyvinyl pyrrolidone; steric stabilizer
pxl	Pixel
SBFI	Serial block face imaging method
SE	Secondary electrons
SEM	Scanning electron microscope
SiO_2	Silica, silicon dioxide
STA	Simultaneous thermal analysis
Stretch	Stretching method of particles in PVA
SLS	Static light scattering
TEM	Transmission electron microscope
T_g	Glass transition temperature of a polymer (see Appendix VIII.1.7.2)
TGA	Thermogravimetric analysis
THF	Tetrahydrofuran; solvent for PS
Transfer	Transfer method of particle monolayers as described in Chapter V.
v/v	Volume to volume
wt%	Weight percent
w/w	Weight to weight
XPS	X-ray photoelectron spectroscopy
1D	One-dimensional
2D	Two-dimensional
3D	Three-dimensional

This work is dedicated to my best friend and wife

Lisa

Chapter I. General Introduction

Since the famous speech of Richard Feynman ("There's plenty of room at the bottom", California Institute of Technology [3]) in 1959 addressing the basics of nanotechnology, many scientific disciplines evolved around the nano- and micrometer scale of materials. The nm/µm region is very fascinating as materials there show effects not seen at larger length scales or in bulk materials. Over the years, many of those effects were investigated in detail and resulted in products indispensable for our modern society. To name just a few of them:

- Micro-/nanostructure of all semiconductor products like transistors, computer chips or LEDs,
- liquid crystals for LCD screens,
- particles and micelles for paints and coatings,
- structures of surface finish to get hydrophilic/hydrophobic properties,
- optical structures and coatings (e.g. anti-reflex for glasses and telescopes)
- particles and micelles as drug carriers in medicine,
- channels in microfluidic systems (e.g. lab-on-a-chip),
- nanoparticles with anti-gen in test stripes for virus detection (e.g. COVID self-test),
- micro filters for air/water filter systems,
- electrodes for batteries and fuel cells,
- catalytic converters in cars, and many more.

The nano/microscale materials can hereby be split into two groups. The first group is bulk material with partial nano/micro scale (surface) structures, which origin mostly from top-down processes (structuring by removing material). A prominent example is the computer chip, where several different structures are etched into a silicon wafer.

The second group consists completely of meso/nanoscale building blocks. The individual building blocks can normally keep their size-effects like color e.g. from size quantification in quantum nanodots, like super-paramagnetism e.g. from nano-magnets, or like catalytic activity. The building blocks might be immobilized on a carrier or by assembly (e.g. aerogels from particles [4]).

Recent material research is focusing on materials from different building blocks with control over different length scales and properties deriving from that. Many ideas on the construction of such materials are taken from material found in nature (biomimetic), e.g. opals [5], nacre [6], teeth [7], and surface structures [8]. The materials are applied at e.g. optics [9], bone/tooth replacement, electrodes for batteries, fuel cells and coatings.

Building blocks of such materials can be isotropic or anisotropic particles. The anisotropic particles can be separated further in functionalization/material anisotropy, like patchy particles with different regions on one particle [10, 11], in shape anisotropy, like ellipsoids, rods, cubes, etc., and in particles with material and shape anisotropy, like patchy particles in ellipsoidal shapes [12-14]. Shape-anisotropic building blocks in nm and µm size range can be made by bottom up, top down or reshaping processes.

The classical bottom up (adding material) approach is the synthesis of particles. In nature, many inorganic materials are found in explicit anisotropic shapes resulting from their crystal growth directions. An example is the iron oxide hematite, which is crystallizing in needles [15, 16] showing similar spindle shapes as the prepared ellipsoidal particles in this thesis. Nowadays, the crystallization of such (nano) crystals can be controlled precisely. Particles of different shapes like spheres, cubes, plates [17, 18], rods (e.g. from hematite [19], silica [20-22] or gold [23-25]), or even unusual shapes like octopods are accessible. A very sound review of such crystal structures and their ordering can be found at Boles et al. [26] Other bottom up approaches for anisotropic building blocks are growing a second particle on an existing particle [27, 28] or melting/combining different smaller particles to a composite particle (e.g. dumbbell particles [29]). Another approach is to synthesize e.g. anisotropic polymer particles in shapes defined by a confined reaction zone, like by micro mold, microfluidic channel [30, 31], shear flow [32] or illumination [33-35]. An overview of anisotropic polymer particle synthesis can be found at Zhang et al. [36] Already present anisotropic particles could also be used as a substrate, e.g. when overgrowing hematite spindles by silica [37-40] or making core-shell and hollow ellipsoids of titania [41].

Top down approaches (removing material) for anisotropic particles come mostly from braking large structures into smaller particles, e.g. nanomilling of medicine [42] or removing parts of large particles or bulk material e.g. by etching [43] or dissolving.

The third proposed class for synthesizing shape-anisotropic particles, which is often forgotten to be an individual class as neither material is removed nor added, is the reshaping process of already present material. With deformable materials like polymers a variety of differently shaped polymer particles could be successfully manufactured: e.g. ellipsoids (details and literature see introduction at Chapter II), discs [44-49], worms [44], rods [44], barrels [44, 50], bullets [44], and several more.

In this thesis, the focus lies on ellipsoidal-like, shape-anisotropic polymer particles made by such a reshaping process. The starting material was hereby a dispersion of polystyrene particles, one of the standard polymers for colloidal particles. The particles used were in the range of 250 – 2000 nm.

I.1. Structure of the Thesis

In the beginning of this work, the literature-known process to create ellipsoidal polystyrene particles was investigated. First, spherical particles were synthesized, then deformed by the given method and the resulting ellipsoids were investigated on their properties. Additionally, a cleaning protocol was developed to recover the particles from the surrounding stretching matrix polymers (Chapter II).

By upscaling and enhancing the process, a continuous and reliable process for manufacturing the ellipsoidal particles could be developed. Additionally, the evaluation of particles on microscope images was semi-automated by programming an image evaluation software in Python (Chapter III).

The particle shape itself was then in the focus for further investigations. Therefore, the particles inside the surrounding matrix were investigated in 3D (Chapter IV).

By using the recovered particles as a structural element, close- and open-packed monolayers were formed. Several different assembly techniques were applied, resulting in large-area crystalline assemblies for some of the particles. The particle positions and the particle orientations within the monolayers were analyzed by pair correlation functions and the Python-based image evaluation software. (Chapter V).

Finally, the ellipsoidal particles could be applied in a DFG founded research project, aiming to make membranes for water filter applications (Chapter VI).

The thesis ends with a summary of the main findings (Chapter VII).

As the different chapters have individual scientific knowledge pre-requisites, the theoretical background for the respective chapter is implemented within the different chapters. Additionally, the chapters contain supporting information.

In the appendix (Chapter VIII), the used general analytic methods with their parameters are explained. The references to the literature are continuously numbered throughout the whole thesis and can be found combined in Appendix VIII.2. In the appendix, the tables of figures, tables and equations are included. The Python program code of the image evaluation tool of Chapter III and code snippets used in other chapters as indicated can be found in the appendix as well.

The digital pdf-version of the thesis and the image evaluation tool as ready to use .py files can additionally be found on the enclosed CD-ROM.

Chapter II. Ellipsoidal Polymer Particles

The preparation of anisotropic ellipsoidal polymer particles is mainly done in literature by utilizing a mechanical reshaping process. As this mechanical reshaping process, which is also known as uniaxial particle stretching process, was described more than 50 years ago by Felder in 1966 [51], and was further developed by Keville in 1990 [52], Ho in 1993 [53], and Champion in 2007 [44], ellipsoidal polymer particles in different size ranges were prepared and used in multiple disciplines by now. The method of Felder et al. is the main synthesis rout followed in this thesis. Thus, it is explicitly explained in the Chapter II.2.

In general, ellipsoidal polymer particles give a size-adjustable model system to investigate anisotropic particle systems. Therefore, in 1968, shortly after first mentioning the synthesis, the particles were used to investigate the influence of the elongation of particles on optical absorption properties [54].

Several studies used the polymeric ellipsoids as model systems for the assembly of anisotropic particles. The investigations on the 3D ordering of ellipsoids-stabilized emulsions [55], dispersion sediments [56] or dense dispersions [49, 57] might be a model system for liquid crystal assemblies. For the investigation with confocal fluorescence microscopes (3D investigations), particles with partial fluorescence labelling by curvature [58], fluorescence core-shell particles [59] or completely fluorescence labelled particles [56] were manufactured.

In biology, the (labelled) polymeric ellipsoids were employed to study cell uptake [60, 61], as novel drug carriers [62, 63], and as possibilities to rupture cell membranes [64].

Fundamental studies of the movement of individual ellipsoidal particles by themselves [65] or propelled by hydrogen peroxide (after attaching a reactive center of platinum) [66] were performed. Even some optical measurement methods were developed/updated using the ellipsoids as model systems, e.g. Micro-Flow Imaging and FlowCAM [67], Scattering Morphology Resolved Total Internal Reflection Microscopy [68], or improved light scattering [69].

A lot of research was also done for ellipsoids in 2D assemblies, as the individual ellipsoidal particle exhibit different capillary forces on tips or sides. Those different forces lead to different wetting behaviors on water surfaces with particle curvatures (tips different to sides) [70] influencing the assembly behavior (e.g. assembly into strings [71]). However, the assemblies of the polymeric ellipsoids in 2D do not just deliver model systems for other anisotropic particle systems, but are interesting in themselves. Due to the anisotropic nature of the particles, the assembled layers can exhibit anisotropic properties as well, which can be exploited e.g. in optical devices. The assembly of ellipsoids and spheres into particle monolayers is the main subject of Chapter V and will be introduced with more detail there.

Although not yet established, the ellipsoidal particles could be used as masks in colloidal lithography or microsieves (see Chapter VI).

Structure of the chapter

The chapter is divided in three parts, starting with the synthesis and definition of the used monodisperse polystyrene particles (Chapter II.1). The following sub-chapter explains the basic process for the preparation of the ellipsoidal polystyrene particles and the features of the used uniaxial stretching device (Chapter II.2). Brief overviews of the necessary theories behind the used processes in synthesis and stretching are displayed.

The spherical and ellipsoidal particles were investigated with several methods with the results depicted in the third part (Chapter II.3). Electron microscopy image evaluation gave access to the dependency of applied mechanical strain to the final particle AR, while light scattering of the ellipsoidal particle dispersions gave insights in their bulk behavior and monodispersity. As a mechanical force is applied during the process, the polymer chains inside the particles did gain internal energy. This internal energy was investigated with modulated DSC measurements.

Some effort was taken to develop a cleaning protocol for the particles to clean them from the surrounding matrix as this was seen to be crucial for later applications. The charge recovering and, therefore, cleaning off the PVA from the particles was checked by zeta potential and XPS measurements during different washing steps.

The main results of the Chapter II.3 are included in a scientific publication (published in the journal *Soft Matter* in November 2023 [1]). The main authors are myself, Dr. Tanja Feller (University Bayreuth), who performed the flow fractionation measurements (Chapter II.3.2, AF4) and thermal analysis measurements (Chapter II.5.4, STA), and Marcel Krüsmann, M.Sc. (University Düsseldorf), who performed light scattering measurements and theoretic calculations (not shown in this thesis). Further co-authors are from University Bayreuth: Dr. Anna Neuhöfer (mDSC measurements, Chapter II.3.3), Friederike Ganster (student assistance 2019 – 2021, helped with enhancing the process and development of the cleaning process), and Professor Dr. Markus Retsch, and from University Düsseldorf: Professor Dr. Matthias Karg. The particle synthesis and main measurements for the paper were done from mid-2018 to the end of 2019. For the development of the process, I got further help by my student assistance Teresa Mauerer (2018 – 2019). Several measurements were performed by or together with the other authors of the paper as indicated at the respective paragraphs. The analysis methods used are explained in separated paragraphs in the Appendix VIII.1.

The chapter closes with a summary (Chapter II.4).

II.1. The Starting Point – Monodisperse Polymer Particles

The anisotropic ellipsoidal polymer particles are prepared by deformation of already existing particles. Therefore, within this thesis, monodisperse polystyrene (PS) particles were used as starting points. The size of the particles does not matter for a successful deformation in the process and it is actually not even important to have monodisperse particles. Mixtures of different sizes are also processable, but for evaluation and later use, monodisperse particles were favored (Figure II-1).

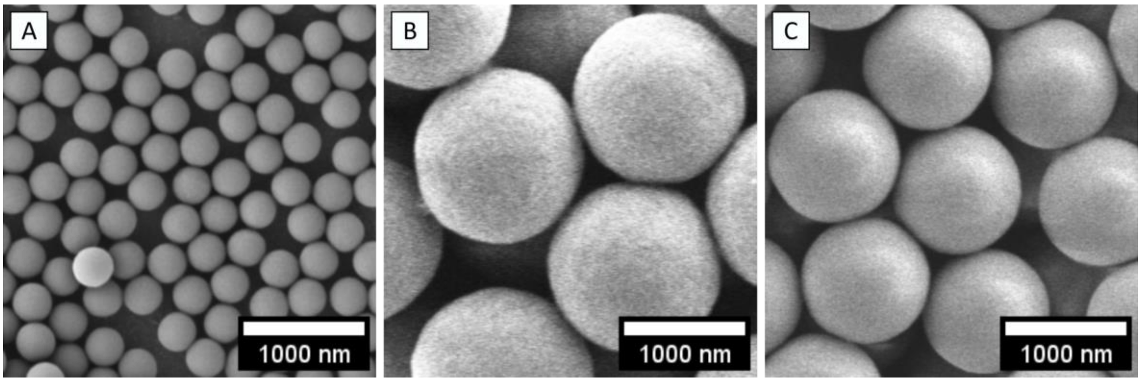


Figure II-1. SEM images of monodisperse spherical PS particles in different size-ranges. A: particle type 1 (326 ± 7 nm, FN269), B: type 2 (1409 ± 18 nm, AL166) and C: type 3 (1058 ± 27 nm, DB07).

The monodisperse polymer particles mainly used in this thesis were synthesized within the work group of Professor Dr. Markus Retsch, chair of Physical Chemistry I, University Bayreuth, either by myself (sample code DB) or by the former group members Dr. Fabian Nutz (sample code FN) and Dr. Anna Neuhöfer (sample code AL) as indicated in the respective paragraphs.

For the synthesis of monodisperse PS particles, two main routes were followed within this thesis.

II.1.1 Radical Emulsifier-Free Emulsion Polymerization

The emulsion polymerization which was patented in 1927 [72] is a type of polymerization where the monomer is dispersed in the continuous phase in droplets, as the continuous phase is a non-solvent for the monomer. The initiator and stabilizers are dissolved in the continuous phase. The normal emulsion polymerization uses emulsifiers above the critical micelle concentration (CMC), creating micelles and stabilizing the monomer droplets. The micelles are the reaction sites where free radicals meet the monomers, hence where the particle grows.

In contrary, the emulsifier- or surfactant-free emulsion polymerization (Figure II-2) has no emulsifier/surfactant added to the reaction vessel as a surfactant is created in-situ by the oligomer. A systematic study on the formation of colloidal polymer particles in an surfactant-free system is displayed in the paper of Goodwinn et al. [73]. Further detailed studies on the particle nucleation mechanism and reaction kinetics can be found at Hansen et al. [74].

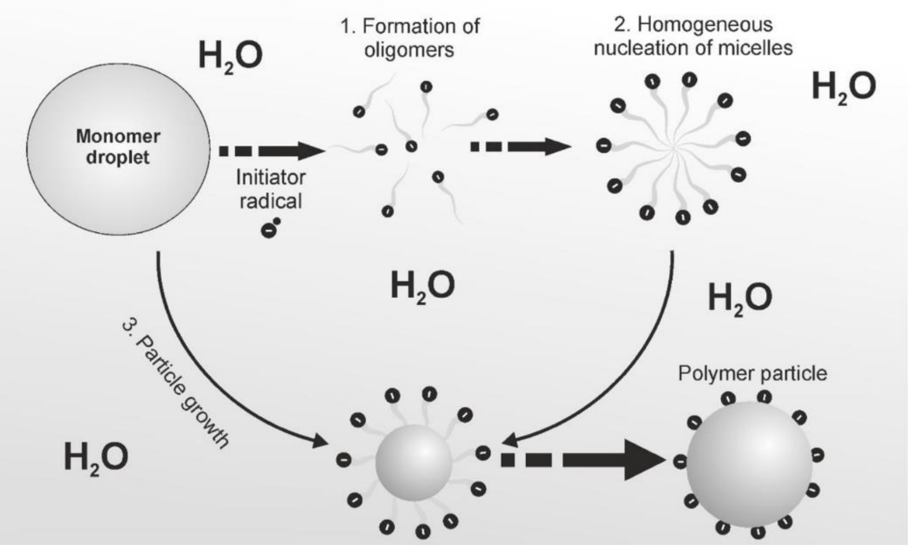


Figure II-2. Sketch of particle synthesis by emulsifier-free emulsion polymerization. Reprinted with permission from Nutz [75]. Copyright © 2017 Nutz.

The general principle of an emulsifier-free emulsion polymerization is explained shortly. The monomer (here styrene) is dispersed in a non-solvent, mostly water, and forms monomer droplets. A soluble initiator is added to the system (common radical starters are 2,2'-azobis-(isobutyramidine)-hydrochloride (AIBA) or potassium persulfate (KPS)). The initiator disintegrates after the start (here by temperature raise) into polar radical groups staying in the water-phase. The small portion of dissolved monomer polymerizes with the free radical inside the water phase and forms by itself an amphiphilic molecule with a polar head and a non-polar tail group, acting as an emulsifier/surfactant. The molecules grow to oligomers and associate to micelles/primary particles [73]. Additional monomers from the monomer-droplets (and free radicals) diffuse over the water phase to the micelles and polymerize at the radical sites inside the micelles. The resulting polymer chains form a stable particle. The particle grows while the charged head groups of the polymer chains stay at the surface, stabilizing the particle.

The charge of the particle surface can be increased by polar additives like co-monomers [76] to counter coalescence and stabilize the particles further. Additional steric additives like polyvinyl pyrrolidone (PVP) can be added to stabilize monomer droplets and help nucleation to occur [77].

Particle diameters from 50 – 300 nm are directly accessible with normal emulsion polymerization while emulsifier-free emulsion polymerization is mainly used for 100 nm – 1 µm particles [78].

II.1.2 Dispersion Polymerization

The other type of synthesis used in this thesis, which was also the main synthesis route used by myself, is the dispersion polymerization.

In dispersion polymerization (Figure II-3) the monomer, initiator and stabilizers are soluble in the continuous phase. The continuous phase, here a water-ethanol mixture, however, is a non-solvent for the polymerization product i.e. the polymer particles [79, 80]. The initiator e.g. 2,2'-azobisisobutyronitrile (AIBN) will not deploy charges and is soluble in the ethanol-water mixture [81]. The polymer grows from the radical starter until the polymer chain collapses on itself, forming a particle (Figure II-3 C). Particles grow by coalescent and by monomer diffusion into the particles.

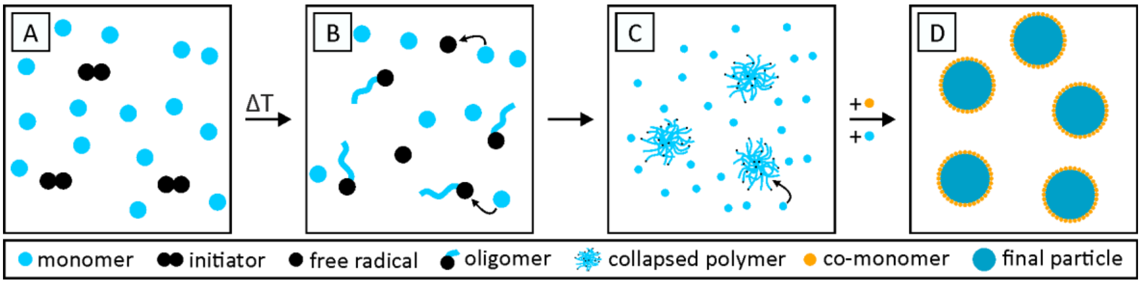


Figure II-3. Sketch of dispersion polymerization. A – C: Adapted from Arshady et al. [78]. A: The monomer and initiator molecules are dispersed in e.g. water-ethanol mixture. B: After increasing the temperature, the initiator disintegrates and the polymerization starts by forming oligomers. C: The oligomers grow and the polymer chains collapse randomly, forming particles. D: With an addition of further monomer and charged co-monomers, the particles grow and become charge-stabilized (surface charge from co-monomer).

Additional steric stabilizers like polyvinyl pyrrolidone (PVP) are used to prevent agglomeration [81]. To counter a later coalescence and to tune the charge on the particle surface, co-monomers with differently charged groups like e.g. sulphate or carboxyl groups for a negative particle charge or amine groups for a positive particle charge are added (after the primary particle nucleation, Figure II-3 C to D).

Because of the different mechanism, typical diameters for particles produced by dispersion polymerization are 500 nm – 10 µm [78], but dispersion polymerization can nowadays also be used for monodisperse particles with diameters of 100 nm – 15 µm [79].

II.1.3 Particles used in the Thesis

The different types of particles used for investigation, are listed in this paragraph.

For all the particle syntheses the supplier of styrene, acrylic acid (AA), potassium persulfate (KPS), 4-styrenesulfonic acid sodium salt hydrate (PSS), polyvinyl pyrrolidone (PVP, 40 000 g/mol) and 2,2'-azobisisobutyronitrile (AIBN) was Merck/Sigma-Aldrich. AIBN was recrystallized in methanol prior to use. Absolut ethanol was from VWR and iPrOH from Brenntag. Ultrapure water (MilliPore water system) was used throughout the synthesis.

II.1.3.1 Particle Type 1 (PS-AA) from Emulsion Polymerization

The particles (Figure II-1 A) used first were negatively charged monodisperse polystyrene (PS) particles with an average diameter of 326 nm (SEM image evaluation, dispersity < 5 %). They were synthesized by emulsifier-free emulsion polymerization (adapted from Goodwin et al. [73]) by former group member Dr. F. Nutz, University Bayreuth. The initiator was potassium persulfate (KPS) and the co-monomer acrylic acid (AA). A detailed recipe for the synthesis can be found in the supporting information at Nutz et al. [82].

*Table II-1. Particle samples of type 1. Additional particles used by my supervised students a) Friederike Ganster and b) Tobias Scheibel, B.Sc. * See supporting information of the respective chapter.*

Sample code	Particle size by SEM [nm]	Coefficient of variation [%]	Zeta potential [mV]	Chapter
FN269	326 ± 7	2.1	-53	II, III
FN270[a)]	337 ± 8	2.3	-56	II*
FN294[b)]	254 ± 6	2.4	N/A	II*, III

II.1.3.2 Particle Type 2 (PS+) from Dispersion Polymerization

The particles with µm-size and a positively charged surface (Figure II-1 B) were synthesized within the group by Dr. A. Neuhöfer (born Lechner) using a dispersion polymerization. For dispersion polymerization the initiator 2,2'-azobis(isobutyronitril) (AIBN) was used. The positively charged co-monomer 2-Methacryloxyethyl-trimethylammoniumchloride (MTC) and neutral polyvinyl pyrrolidone (PVP) were added to increase the steric stability. The recipe for the dispersion polymerization for positively charged particles can be found at Lechner et al. [83].

*Table II-2. Particle sample of type 2 displayed in the thesis. * see supporting information VI.7.1*

Sample code	Particle size by SEM [nm]	Coefficient of Variation [%]	Zeta potential [mV]	Chapter
AL166	1409 ± 18	1.3	N/A	II, VI*

II.1.3.3 Particle Type 3 (PS-PSS) from Dispersion Polymerization

The mainly used negatively charged particles of around 1 µm (Figure II-1 C) were synthesized by myself in dispersion polymerizations (Figure II-3) using the co-monomer 4-Styrene-sulfonic acid sodium salt hydrate (PSS). The initiator was AIBN and PVP was added as a stabilizer.

II.1.3.3.1 Synthesis Protocol

The following detailed recipe for the particles of type 3 (PS-PSS) with sizes around 1 µm was adapted from the recipe for dispersion polymerization used in the paper of Lechner et al. [83]. The main change thereby was the use of PSS as a negatively charged co-monomer.

A 250 ml 3-neck flask equipped with a large magnetic stirring egg (4 cm), a reflux condenser and a gas inlet was charged with 46 ml absolute ethanol, 10 ml water and 11 ml styrene monomer. Then 3 g PVP (40 000 g/mol), dissolved in 10 ml absolute ethanol via ultrasonication and vortexing, were added to the flask. Additionally, 300 mg recrystallized AIBN were added, too. The solution was degassed by argon while stirred with a speed of 150 rpm. After 30 min, the mixture was heated to the reaction temperature of 70 °C. The transparent mixture turned turbid, then white.

56 ml ethanol, 5 ml water, 11 ml styrene, and 300 mg of co-monomer PSS were premixed in an Erlenmeyer flask, and added to the reaction after 90 min.

The reaction was carried out overnight under a slight argon flow. The polymerization was stopped after 24 h by exposing the dispersion to ambient air and filtrated using a 125 µm nylon filter sieve (E-D-Schnellsieb Nylon super-fein). Monodispersity could already be seen by colorful reflexes on the flask.

To remove the PVP and unreacted species, the dispersion was washed by centrifugation (10 000 rcf) using two times pure ethanol, then by using an iPrOH mixture (3:7 v/v iPrOH:water) once and finally using 3 times pure water. The dispersions were stored in air-tight *SCHOTT* bottles at concentrations of 10 – 16 weight percent (wt%).

II.1.3.3.2 Synthesized Particles

The protocol was altered to synthesize differently sized particles by changing the amount of water and the amount of styrene. Less water and more styrene (reducing the water content as well) should result in larger particles [84]. A calculation of the yield after cleaning showed that the reaction was not completed after the given 24 h, as the yields were below 90 %. Therefore, the real influence of the different amounts of water or styrene on the particle size could not be evaluated. However, the trends proposed in literature were clearly visible, as the particle size increased with more styrene (DB10 to DB07 or DB13 to DB14) and less water (DB07 to DB14 or DB12 to DB11 and further to DB13).

Table II-3. Monodisperse particle batches prepared for and used in chapters of this thesis. The yield is calculated by the dry particle mass from wt% and mass determination of the latex, and by the total amount of styrene (mass by density) used.

Sample name	**Total styrene [ml]**	**Total water [ml]**	**Yield by mass [%]**	**Particle size by SEM [nm]**	**Coefficient of variation [%]**	**Zeta potential [mV]**	**Chapter**
DB07	22	15	87	1058 ± 27	2.6	-64	II, III, V, VI
DB10	17	15	67	1001 ± 28	2.8	N/A	--
DB11	16	15	72	979 ± 32	3.3	N/A	--
DB12	16	20	79	950 ± 14	1.5	-63	VI
DB13	16	10	77	1148 ± 20	1.7	-70*	V
DB14	22	10	79	1387 ± 19	1.4	-52*	III, IV

* Zeta potential of dispersion was measured after embedding in PVA, heating and recovering with iPrOH mix

To increase the size further, I tested a low amount of 7 ml water with 22 ml styrene. However, the reaction mixture became very viscous leading to polydisperse particles of 1406 ± 100 nm (CV 7.1 %). Hence, to increase the size further, a longer reaction time or an addition of further monomers to a later time is recommended.

II.1.3.4 Particle Type 4 (PS-AA) from Dispersion Polymerization

Similar to particle type 3, a batch of particles with acrylic acid as co-monomer was synthesized. The recipe was the same as described in Chapter II.1.3.3.1. The Initiator was AIBN. The 300 mg of co-monomer PSS was replaced with 109.4 ml acrylic acid.

Table II-4. Monodisperse particle batch of PS-AA prepared for and used in Chapter VI.4.

Sample name	Total styrene [ml]	Total water [ml]	Particle size by SEM [nm]	Coefficient of variation [%]	Zeta potential [mV]	Chapter
DB08	22	15	984 ± 33	3.4	-56	VI

II.2. The Stretching Process – from Spheres to Ellipsoids

To transform isotropic polymeric particles into anisotropic, ellipsoidal shapes, a uniaxial stretching process is widely used. The process was already mentioned by Felder in 1966 [51] and was then picked up and further developed by Keville in 1990 [52], Ho 1993 [85], and Champion 2007 [44]. The process requires the dispersion of polymer particles into a matrix, which can be mechanically stretched at temperatures exceeding the glass transition temperature of the polymer beads.

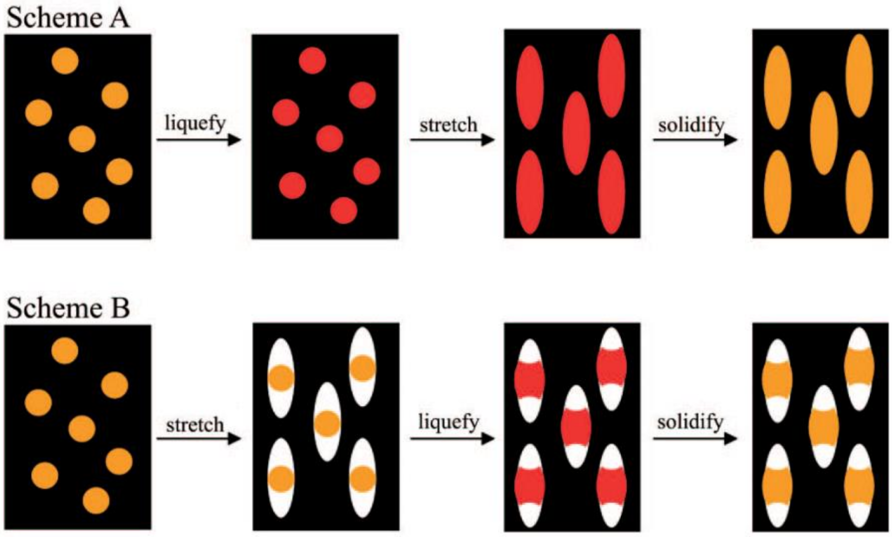


Figure II-4. The process of deforming particles inside a matrix. Sketch from Champion et al.[44]. Reprinted using the Standard PNAS License Terms. Copyright © 2007 National Academy of Sciences.

By heating the embedded polymer beads (here PS) above their glass transition temperature T_g (PS T_g = 107 ± 2 °C [86], see Appendix VIII.1.7.2), the polymer beads get soft and deformable. The surrounding matrix (mostly PVA or PDMS) can be seen as a flexible mold and defines the resulting particle shape. By consecutive stretching after heating, the polymer beads are deformed in a regular way by filling the emerging voids around them (Figure II-4 Scheme A). When applying an uniaxial stretching force, the resulting shape is ellipsoidal, while different other shapes, e.g. discs, can be manufactured by multiple stretching directions applied through mechanical stretching devices [44, 47] or through blow stretching [45, 87]. The shape of the resulting ellipsoids is highly predictable as the stretching is a simple mechanical process and can be simulated and calculated by the assumption of volume conservation.

The true resulting shape and dimensions of the beads are additionally defined by the point of time for liquefying/softening the polymer beads, which can be done either by a change of temperature or by using solvents. By changing this liquefying time to a short time period after already applying the strain, the polymer in the beads cannot fill the present void completely (Figure II-4 Scheme B, see also Chapter III.2). Therefore, the process gives access to even more different shapes of particles, e.g. barrel-, string-, or bullet-shapes [44, 50].

On the one hand, by applying monodisperse particles as beads, the resulting deformed particles exhibit narrow size distributions. On the other hand, one has to take into account, that every disturbance of the forming void around the particle, by e.g. other particles nearby, bubbles, or surfaces, leads to uneven deformations and, in the case of nearby particles, into particles partially melted together. Therefore, the amount of different particle shapes increases naturally. Additionally, as with higher stretching ratios it is more likely that the particles hinder each other by forming the void, it is expected that the shape and size distributions increase with increasing amount of strain.

The polymer particles used in this thesis were polystyrene particles, hence in literature, other polymer particles have been successfully used with the given stretching process as well: e.g. PS@PNIPAM [88], PCL, poly(beta-amino ester) PBAE, lactic acid to glycolic acid PLGA [89], polystyrene core/polyglycidol-rich shell microspheres [90].

II.2.1 Synthesis of Ellipsoidal Particles

For the basic results (Chapter II.3) of the present chapter, particles of type 1 & 2 (not self-synthesized) were used with the following preparation protocol, while foils from particle type 3 were mainly produced with the continuous foil doctor blade (see Chapter III.1).

II.2.1.1 Preparation of Foils

After extensive washing with water using a centrifuge, the spherical particles in dispersions were added to a polyvinyl alcohol solution (PVA, Mowiol® 8-88, Mw ~67000, 86.7 – 88.7 mol% hydrolysis from Merck/Sigma-Aldrich). The applied weight content was 20 wt% PVA and up to 4 wt% PS in the final dispersion, leading to up to 24 wt% solid content in the dispersion. The mixing was performed via "a SpeedMixer (Hauschild DAC 400.2 VAC-P). A gas-free suspension was obtained by applying vacuum (~ 50 mbar) during speed-mixing at 2500 rpm for 10 min" [1].

The resulting viscose slurry "was doctor bladed into films of up to 20 cm × 37 cm on a glass plate (wet film height 1 mm; blading speed 8 mm/s; film applicator Zehntner ZAA 2300 [; height-adjustable doctor blade aka 'Film Casting Knife' from BYK of 20 cm width]). After drying for 2 days at room temperature" [1], the foil was cut into pieces of up to 9 cm × 4 cm. The dry foils contained up to 16.7 wt% PS (from 20 : 4 dispersions).

The long foil ribbons, produced with the continuous foil doctor blade (see Chapter III.1) using up to 30 wt% PVA solutions and particles of type 3 (PS-PSS of around 1 µm), were also cut into pieces of 7 – 10 cm × 4 cm when stretched in the manual stretching device.

II.2.1.2 Stretching of the Foils

The foil pieces were mounted as a stack of multiple foil pieces in a manual uniaxial stretching device. The self-made stretching device was of stainless steel with a spindle of brass and manufactured by the mechanical workshop NW I at University Bayreuth. The final design of the normal stretching device (max stretching to 20 cm) can be seen in Figure II-5 with small brackets (width 6 cm) which were elevated from the ground plate to minimize its influence by heat reflection/storing.

The device with the particle-loaded foils was heated in a convection oven to 150 °C for 30 min. Note that for a successful particle stretching process, the temperature has to be above the glass transition temperature of the embedded polymer particles, here PS: $T_{g\_midpoint}$ = 107 ± 2 °C [86] (see Appendix VIII.1.7.2). After the equilibration time, the foils were stretched to the desired aspect ratio (AR = long axis/short axis) in multiple steps, depending on the targeted AR of the particles. Before and during the application of the strain, the screws of the device needed to be re-tightened outside the oven to avoid slipping. The strain was checked by cardboard templates, measuring the distance of the lining. Every time between the stretching steps or the tightening of the screws outside the oven, 2 min of equilibration time in the oven was given. The total process time was around 1.5 – 2 h.

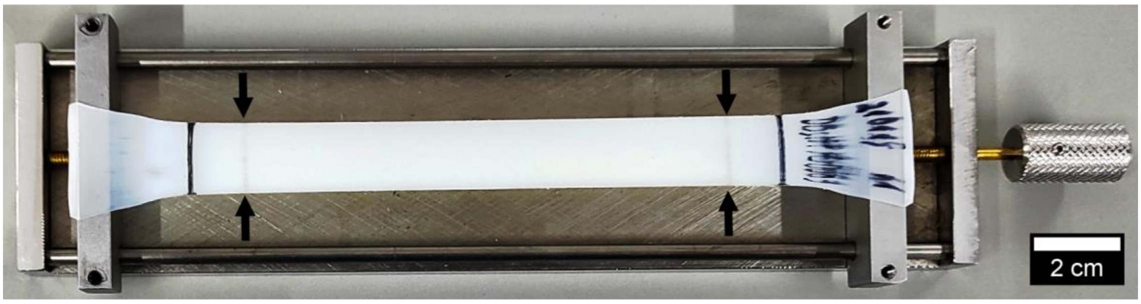


Figure II-5. Example of a foil after stretching in the stretching device. The former rectangular sample becomes bone shaped. The middle part is evenly stretched. The loss at every side due to clamping and shear thinning (bone shaping) was 1 cm. The pencil marks (black arrows) were at 1.5 cm from the sides before stretching. Total loss of foil was around 3 cm in length from original length.

II.2.1.3 Recovering the Particles

The homogeneous-stretched middle pieces of the stretched foils (Figure II-5, part in-between the pencil marks (black arrows)) were cut out with a scissor. Approximately 1 g of foil was dissolved in 30 ml of an iPrOH:water mixture (3:7 v/v) in a round-bottom flask at 60 °C stirring at 150 – 300 rpm for 30 min (hot wash). "Centrifugation and redispersion in hot iPrOH:water was repeated additional 2[– 3] times, followed by 3 times of centrifugation and redispersion in ultrapure water." [1]

The additional hot washes were first performed inside a round-bottom flask, while later on, the hot wash steps after the 1st dissolving "hot wash" were updated to washing inside the centrifuge tube (Falcon tube) with sonication (37 kHz) at 60 °C for 30 min. This update made the whole process less time consuming, as multiple samples could be cleaned in the tubes inside the same sonication bath. The cleaning inside the centrifuge tube worked as accurately as the cleaning in the flask (see zeta potential check, Chapter II.3.4.4).

All cleaned ellipsoids were stored in centrifuge tubes at various concentrations.

II.2.2 Advances of the Home-Build Stretching Device

In this short paragraph, the experience gained from tuning the simple stretching device is depicted. The paragraph should help scientists encountering problems with similar stretching device designs. It therefore shows my hands-on findings and solutions for countering some arising problems.

II.2.2.1 Dimensions of the Device

As around 3 cm from the original sample length have to be excluded due to uneven strain by clamping and necking (bone shape), the use of longer foils to increase the yield can be favored. Starting with a geometric basic yield of 40 % at 5 cm long foils, the yield was increased to 66 % for 9 cm long foils (later even 75 % for 12 cm foils).

The original stretching device with a maximum stretching size of 20 cm (Figure II-5) was not capable to stretch long foils. A stretching device of 37 cm maximal stretching size (outer length 38.5 cm + handle), fitting exactly into the convection oven, was built by following the same basic design (5 cm width ground plate with 6 cm width of brackets). In general, the scaling of the device gave no problems. The stretching of the long foils worked similar to the stretching of short foils, with an evenly stretched middle part.

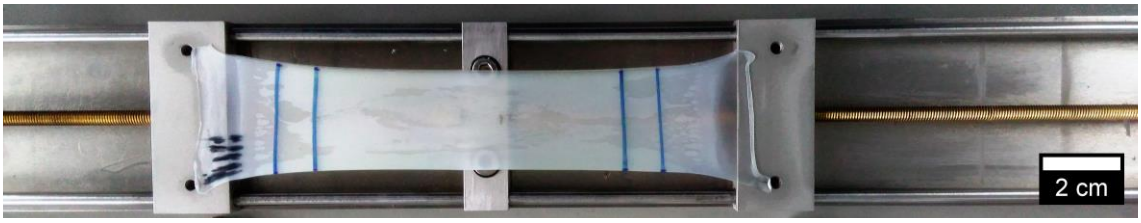


Figure II-6. Photograph of foil stack in long stretching device with steel block in middle and thick brackets. Scale bar estimated due to perspective view. Constant stretching area in-between the inner marks. Marks are tilted due to slipping on right bracket (upper part).

However, the design of the long device was slightly altered by a steel block in the center of the device, holding the guiding sticks and the brass spindle in place (Figure II-6). This block was later removed as the foil showed uneven deformation right above this additional steel area. The foils were wider at that area, leading to the assumption that the temperature at that spot was different and strain relaxation occurred. This assumption is in agreement with Ho et al. [91] who inferred that necking could be lower at higher temperatures.

II.2.2.2 Brackets

The design and dimensions of the brackets holding the foils can be important for a successful stretching process. The brackets of the first stretching device were 2 cm wide and the screws were just 3.5 cm apart (Figure II-6). If the foils would be clamped over the whole bracket (in stretching direction), the loss of foil length would be in total 4 – 5 cm and the width would be limited to 3.5 cm. To reduce the foil loss and enlarge the width, the foil was clamped half into the brackets, leaving the brackets tilted. Although accurately stretched foils could be produced by this configuration, a problem arose when multiple foils as a stack were used. The tilting of the brackets increased and the foils started to slip (e.g. Figure II-6 on right bracket). To balance this, some small waste foil pieces were mounted next to the foil stack into the brackets so the tilting decreased. However, multiple times the foils slipped, especially with large strain.

Therefore, new brackets with smaller width were produced, where the foil ends could be mounted over the full length and just 3 cm in total original foil length (1 cm per bracket + 1 cm by necking) were lost (Figure II-5). Later, these simple brackets got a wider foot (L-shape of brackets) to avoid jamming on the guiding rails. With these new brackets, the screws were additionally moved to the side, so foils of 4 cm width could be mounted.

The new brackets were additionally elevated, to reduce the influence of heat from the metal surface beneath.

II.2.2.3 Screws

To get enough pressure on the slim brackets and to avoid braking of the screws (happened multiple times with the tilted brackets), the screws got a wider diameter (M3 to M4) with the new bracket design. It was also found that hex bolts were the most sufficient way to tighten and especially loosen the screws after cool down, as all other tested screw heads got wasted during screwing (slot screw, cross screw, allen screw).

II.2.2.4 Counter Slipping

The brackets were sand blasted to enhance the grip and with the new slim brackets where the foil could be clamped over the whole bracket (Figure II-5), the slipping of foil stacks was eliminated completely. However, to stretch single foil pieces it was found to best use additional sand paper pieces clamped from top and bottom onto the single foil inside the brackets to eliminate slipping.

II.3. Stretched Particles – Basic Results

Most of the results and graphs in this chapter were used in my paper (Benke et al. [1]) describing ellipsoidal particles made from particle type 1 (PS-AA) and 2 (PS+). However, additional information on particle type 3 (PS-PSS) is added here (see Chapters II.3.1.3, II.3.4.3, and II.3.4.4), which is not included in the submitted paper.

Apart from Chapter II.3.2.3, further scattering investigations (depolarized dynamic light scattering and static light scattering) on the particle dispersions of type 1 (Figure II-7), were conducted for the paper by M. Krüsman, M.Sc. and Professor M. Karg at University Düsseldorf. The data and discussion can be found in the paper [1] and will not be displayed or discussed here.

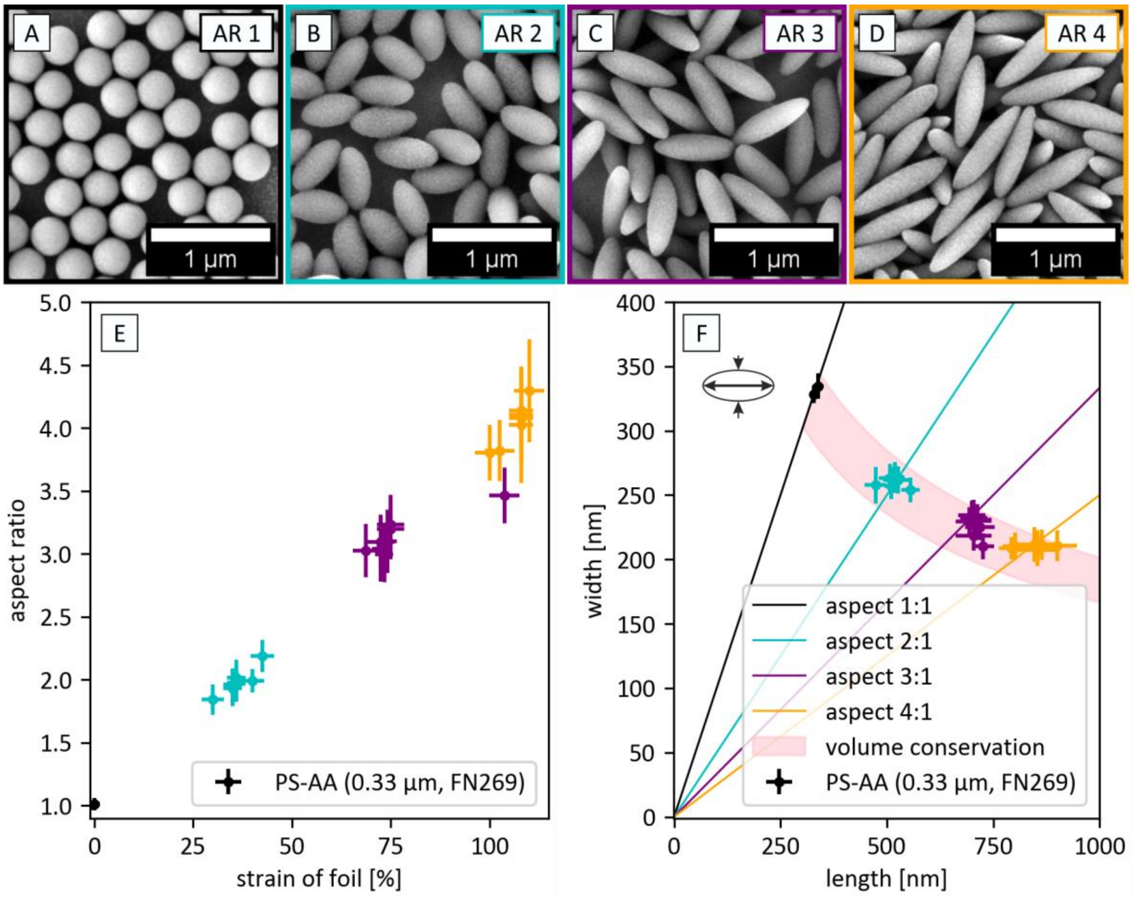


Figure II-7. Combination of data from particles of type 1 (PS-AA, FN269, 326 nm). A – D: SEM images of the particles at different ARs. E: AR plotted against the strain of the foil. Linear plots see Figure II-9. F: Width and length plot of different stretched particles. The volume conservation area was calculated by min/max size of the original particles. With large ARs, the simple ellipsoidal shape with two different axes might not be correct anymore, as the data points are not centered in the volume conservation area (hint to Chapter IV.6.4). Data points of individual batches colored black for particles AR < 1.5, cyan AR 1.5 – 2.5, purple AR 2.5 – 3.5 and orange AR 3.5 – 4.5. This figure is a replotted and altered version of figure 1 in my published paper [1] (just showing PS-AA particles here). Used with permission of Royal Society of Chemistry, from Benke et al. [1] ©2024; permission conveyed through Copyright Clearance Center, Inc.

II.3.1 Dependency of Foil Strain to Particle AR

The dependency of the applied strain onto the particle AR is one major research object and is, therefore, already shown in the early literature on the stretching method [85].

As explained in Chapter II.2, by stretching the foil, voids grow around the particles along the direction of stretching. The voids grow with applied strain and, therefore, the particles deform increasingly resulting in growing ARs (long axis/short axis) with applied strain (Figure II-7). For small particles of PS-AA (type 1, FN269, 326 nm) a strain of the foil of 36 % (elongation of the film to 1.36 of the original length) resulted in an AR of 2, while 108 % (2.08 times original length) gave an AR of 4. Consequently, a strain of around 72 % leads to an AR of 3. The dependency could, therefore, be fitted linear in the examined regime of AR 1 – 4.

With the linear dependency of strain to AR, the AR of the resulting ellipsoidal particles could be precisely tuned by applying a defined strain to the foil.

II.3.1.1 Inter-Sample Reproducibility

To produce a sufficient amount of particles for further use, multiple particle-loaded foils (and foil stacks) were stretched to the same strain and evaluated separately (Figure II-8).

The SEM image evaluation (by Fiji/ImageJ, version 1.53c) of the different samples show normal (*GAUSSIAN*) distributions in the AR and in their particle dimensions (length and width). The aspect distributions of similar stretched foils were nearly the same (Figure II-8 A – C). Therefore, the particle dispersions could be used together as one batch for the further evaluations.

The dispersity increased with higher strain as expected. Particles with an average AR of 4 (108 % strain, Figure II-8 F) had a dispersity twice as high as particles from AR 2 (36 % strain, Figure II-8 D).

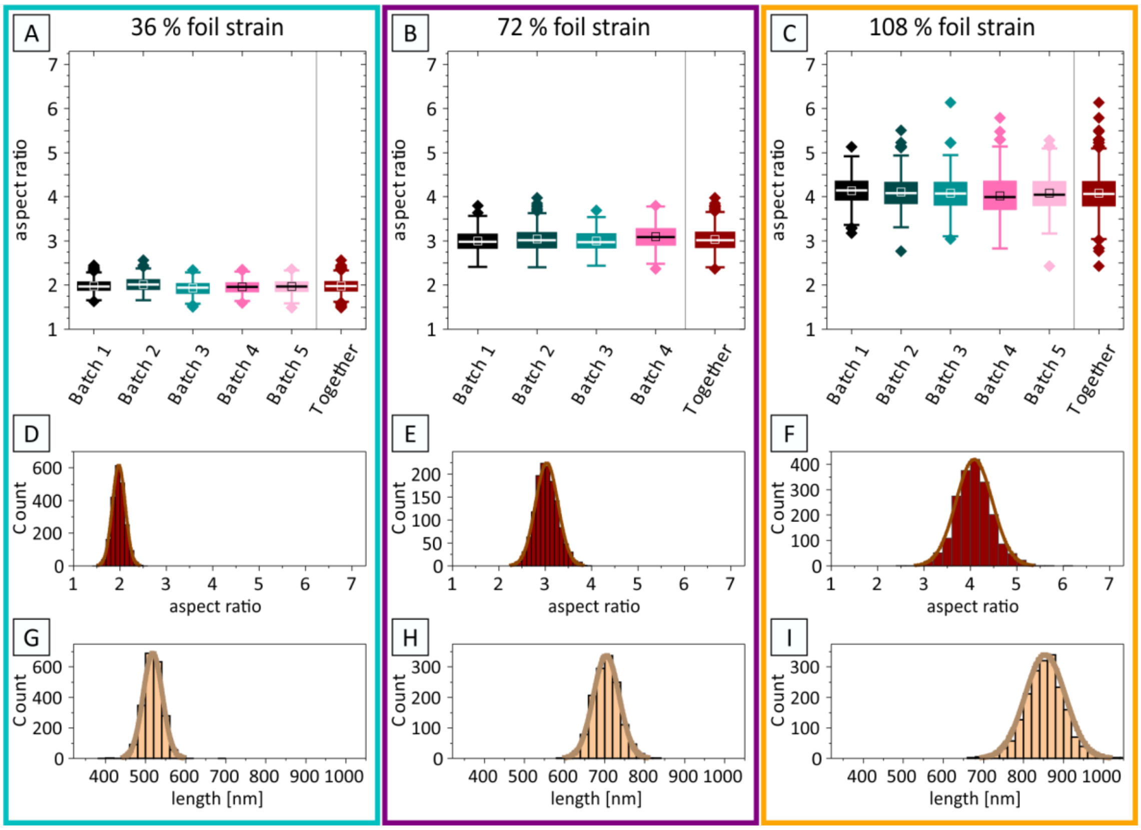


Figure II-8. Particle reproducibility shown for particle type 1 and different foil strain: A: 36 %, B: 72 %, and C: 108 %. The normal distribution of the AR (D – F) and length (G – I) of the combined batches (together) are displayed. This figure is a replotted version of figure S11 in the supporting information of my paper [1]. Used with permission of Royal Society of Chemistry, from Benke et al. [1] ©2024; permission conveyed through Copyright Clearance Center, Inc.

II.3.1.2 Different Foil Sizes

In terms of inter-sample reproducibility, it is worth mentioning that regardless of the foil size (pieces used were 5 – 9 cm in length and 3 – 4 cm in width), the same applied strain for the same particles resulted in the same AR of the ellipsoids. Therefore, extremely long foil pieces could be used to reduce the clamping loss. This finding was actually further exploited by making 8 m long foil ribbons with the continuous doctor blade device and stretching them by roll-stretching as implemented in Chapters III.1 and III.2.

II.3.1.3 Different Particle Sizes and Charges

The method of stretching polymer particles is applied on various particle sizes throughout the literature ranging from several 100 nm to multiple µm [85]. Within this thesis, two different size ranges were investigated. The one size range of particle type 1 (PS-AA, FN269) with around 300 nm was the main scope of this chapter. However, particles with sizes around 1 – 2 µm (second size range) were mainly used in the following chapters.

For the larger particles, type 2 (PS+) and type 3 (PS-PSS), the strain which was needed to get the respective AR was slightly lower compared to the small particles of type 1. For type 2 particles (PS+ 1.4 µm, AL166), a strain of 36 % resulted in AR 2.1 and not 2.0 as for the small particles. A strain of 108 % resulted in AR 4.2 and not 4.0. The freshly prepared self-synthesized PS-PSS particles (Type 3) in size of 1 µm (DB07) with strains of 0 %, 15 %, 25 %, 50 %, 75 %, and 110 % (measured percentage rounded to 5 %) were behaving similar to the large type 2 particles (PS+, AL166). Thus, an influence of the different surface charges was not visible.

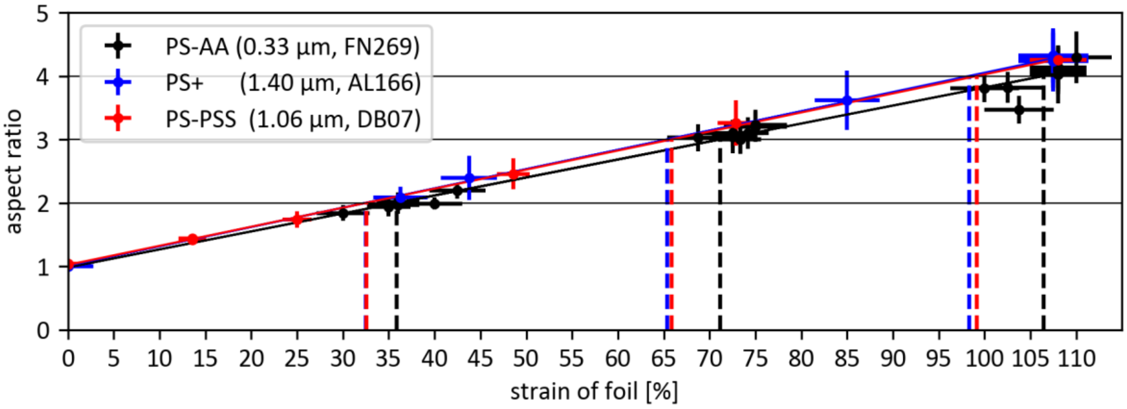


Figure II-9. AR compared to strain of foil. In the depicted range, the dependency is linear. Larger particles need less strain for a higher AR. The intersections of the linear fits with AR values 2, 3, and 4 are displayed as dashed lines. Note that for PS-AA and PS+ the amount of particles in foil was 16.7 wt% (PS : PVA 1 : 5 w/w) while for PS-PSS it was 10 wt% (PS : PVA 1 : 9 w/w).

When fitting the data linear (Figure II-9), a size dependency to smaller strains with bigger particles could be identified (further data of different particles can be found in the supporting information II.5.1). The finding that the slope of the linear dependency is getting steeper for big particles was described as well by Ho et al. in 1993 [85].

In general, by the simple linear dependencies, dependent on the size of the particles, it was possible to forecast the necessary strain to get the aimed AR.

II.3.1.4 Volume Conservation

The easiest mathematical description of the stretching process would be an elongation of one axis of a sphere, resulting in an ellipsoid with the perpendicular axis of the same length. The equation for an ellipsoid by basic math has three different axis-radii (x, y, z). If one dimension is altered (x; stretching direction), the other two radii should stay the same $(y = z)$. From the SEM images, x is identified as half of the length l and y as half of the width w. The volume of a uniaxial stretched ellipsoid is then calculated by Equation II.3-1.

$$V_{el} = \frac{4}{3}\pi(x \cdot y \cdot \mathrm{z}) = \frac{4}{3}\pi(x \cdot y^2) = \frac{4}{3}\pi\left(\left(\frac{l}{2}\right) \cdot \left(\frac{w}{2}\right)^2\right) \qquad \text{(Equation II.3-1)}$$

The corresponding sphere was calculated by a basic math equation for a sphere with radius r or diameter d (Equation II.3-2).

$$V_{sp} = \frac{4}{3}\pi r^3 = \frac{4}{3}\pi\left(\frac{d}{2}\right)^3 \qquad \text{(Equation II.3-2)}$$

For a much easier handling and the discussion of the corresponding sphere volume, the diameter of the corresponding sphere was calculated by combining the above-mentioned equations (Equations II.3-1 and II.3-2). The error was calculated by error propagation law (see supporting information II.5.2).

$$V_{el} = V_{sp} \qquad \text{(Equation II.3-3)}$$

$$d = (l \cdot w^2)^{\frac{1}{3}} \qquad \text{(Equation II.3-4)}$$

However, the real particle itself was not just elongated in one direction but was also compressed in the other two directions perpendicular to the stretching axis as the volume of the particle cannot change, only when two particles melt together. This decrease in width as well as the length increases can be seen in the experimental data (Table II-5).

Table II-5. Individual examples for particle dimensions and the diameter of their volume corresponding sphere for FN269 particles and some of their stretched analogous. Diameters, calculated with Equation II.3-4, and error propagation (Equation SI II-2) rounded to integer. This table can be found in my paper [1] as table 1. Used with permission of Royal Society of Chemistry, from Benke et al. [1] ©2024; permission conveyed through Copyright Clearance Center, Inc.

AR	length [nm]	width [nm]	calculated diameter of volume corresponding sphere [nm]
1	327 ± 7	325 ± 7	326 ± 5
2	556 ± 22	254 ± 10	330 ± 9
3	711 ± 35	230 ± 11	335 ± 12
4	901 ± 47	211 ± 12	342 ± 14

The diameters of the corresponding spheres of the stretched particles were larger than the diameter of the original sphere and showed a continuously growing deviation from the original diameter with increasing AR. Therefore, the true shape of the stretched ellipsoids might not be a regular ellipsoid as defined by the basic equation (similar to Ho et al. [85]). Indeed, investigations of the SEM images did show multiple shape deviations from the optimal ellipses with which the particles were fitted.

The differences in shape not being a regular ellipsoid, especially when the tips were pointier, lead to an overestimation in volume and, therefore, resulted in a higher corresponding sphere diameter. This can be seen as well in Figure II-7 F, where the data points of AR 4 started to migrate to the upper inner border of the range for the corresponding sphere min/max (pink area). However, as the difference of the corresponding sphere diameter was low and still within the error, while the error is getting larger with higher AR, the volume was counted to be conserved.

II.3.2 Particles in Dispersion

The actual size distributions of the given particle samples are hardly accessible based on local image measurements. Therefore, bulk techniques are normally used to investigate particle dispersions.

For small particles (type 1, FN269), the field flow fractionation technique (FFF) coupled with an UV detector, a multi-angle light scattering detector (MALS, here the UV detection signal was used) and an in-line dynamic light scattering (DLS) could be used by my colleague Dr. T. Feller to first separate particle dispersions and then characterize the size distribution of the different fractions. Additionally to the investigation on ellipsoidal particles, we used this method to analyze the influence of the fundamental processing conditions like embedding and heating. Therefore, the original spherical particles (Figure II-10, sample code “PS Spheres”) were treated in two different ways. Firstly, the particles were embedded into a PVA foil and were then immediately recovered from the dried foil without heating (sample code “Embedded”). Secondly, they were embedded, clamped into the stretching device (no stretching), and heated for 1 h at 150 °C with subsequent recovering of the particles (sample code “Embedded & Heated”).

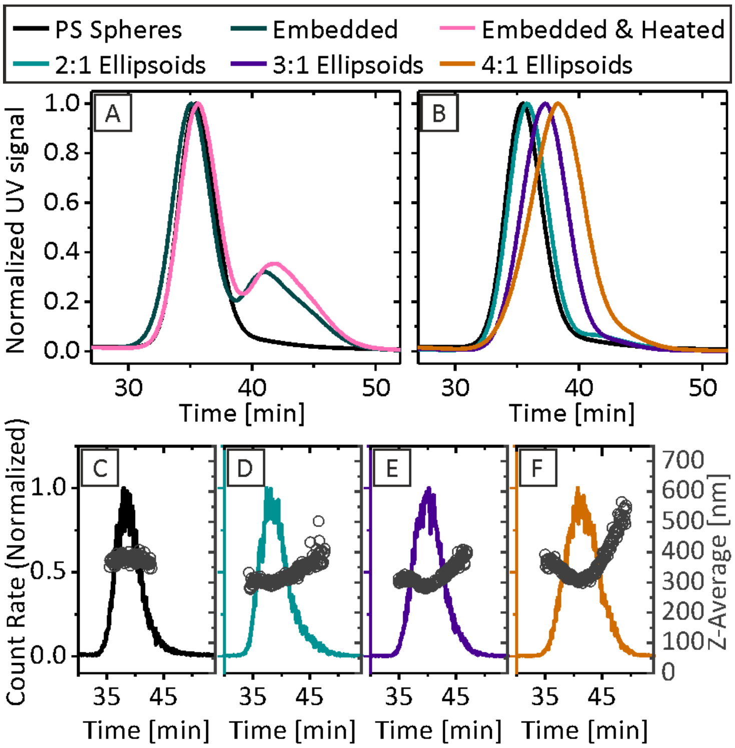


Figure II-10. Elution times of particle samples measured by the UV detector after FFF for (A) spherical and (B) stretched particles. C – F: UV detector signal of MALS and in-line DLS measurements of the respective size (circles) of (C) untreated PS spheres, (D) 2:1, (E) 3:1 and (F) 4:1 ellipsoidal particles; for embedded and heated spherical particles see next Figure II-11. Data measured by my colleague Dr. T. Feller at University Bayreuth. The figure can be found in my paper [1] as figure 3. Used with permission of Royal Society of Chemistry, from Benke et al. [1] ©2024; permission conveyed through Copyright Clearance Center, Inc.

II.3.2.1 Retained Monodispersity

The spherical particles (Figure II-10 A "PS spheres") showed the expected normal distribution of a monodisperse particle sample on the UV detector. Surprisingly, an additional distinct peak at a higher elution time with the treated spheres was present (see Chapter II.3.2.2).

By evaluating the elution times of the ellipsoids, all samples showed normal distributions, displaying the retained monodispersity. In good accordance with the SEM image evaluation, the particles with the highest AR exhibited the broadest peak i.e. highest degree of variability of size homogeneity. Furthermore, the elution time increased with higher ARs (Figure II-10 B), which could be explained by a higher interaction of the ellipsoids with the FFF membrane or/and a possible alignment along the channel with a small diameter not reaching far in the channel resulting in longer elution times.

II.3.2.2 Cluster Formation

The UV detector results (Figure II-10 A) for the small particle samples showed that embedding already led to a second peak at higher elution times and heating supported this (Figure II-10 A pink) even more. In-line DLS measurements proved that the later eluted particles were larger than the first eluted (Figure II-11 Z-Average data points as grey circles above the curves). Therefore, we concluded that there were clustered particles in the dispersion.

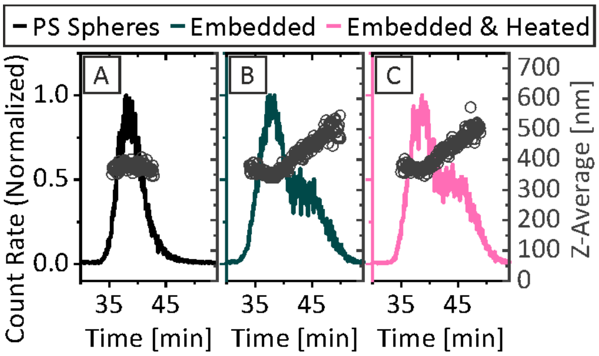


Figure II-11. In-line UV data of MALS detector combined with in-line DLS measurements (circles) of A: original PS spheres and B, C: treated (embedded and heated) particles. The second peak (shoulder) shows distinct larger particles, which were identified as clusters. Data measured by my colleague Dr. T. Feller at University Bayreuth. This figure can be found in the supporting information of my paper [1] as figure S12. Used with permission of Royal Society of Chemistry, from Benke et al. [1] ©2024; permission conveyed through Copyright Clearance Center, Inc.

Although a lower weight fraction of particles in the PVA film might reduce the cluster formation, the yield of the particle fabrication has to be considered. I chose 4 wt% particles in a 20 wt% PVA solution (PS : PVA 1 : 5 w/w, 16.7 wt% particles in dry foil), to have a sufficient stretching process with a good yield in the tested range for the small particles.

Additionally, when investigating stretched particle samples with different ARs prepared from the same original particles (FN269), we found that the peak for present clusters in the spherical particle samples was almost not visible in low stretched samples and cannot be seen in the higher stretched samples (Figure II-10 B). We concluded that the stretched particles could be separated again during recovering from foil, but they were agglomerated when not stretched at a particles concentration of 16.7 wt% in dry foil (PS : PVA 1 : 5 w/w).

Therefore, we do not advice to go to higher weight ratios of PS to PVA (particle wt%) as the cluster formation might increase and not all clusters can be broken up by recovering (see SEM image at Chapter IV.8, Figure IV-44). Note that to my knowledge other publications, where the same stretching process is used, show results for much lower weight ratios of PS to PVA (maximum 1:10 w/w, 9 wt% PS in dry foil).

Thus, as soon as the yield was not that much important anymore, due to the scaling up of the foil manufacturing and roll-stretching process (see Chapter III), a lower particle content of 10 wt% in dry foil (PS : PVA 1 : 9 w/w, for big particles) was used to avoid cluster formation.

II.3.2.3 Hydrodynamic Diameters (Z-Average)

As expected, the hydrodynamic diameter of the original particles (Figure II-10 C grey circles) was a bit higher than the diameter by SEM evaluation. The size of the embedded and the embedded & heated samples (points over peak, Figure II-11) were close to the original particles, indicating that probable PVA residues on the particle surface did not change the hydrodynamic radius. However, the visible second peak points clearly to either bigger particles or particle clusters with the higher hydrodynamic diameter.

The in-line DLS measurements of the ellipsoids showed a minimum in their hydrodynamic diameter at the maximum of the particle count (Figure II-10 D – F grey circles). While the later raise in size could be referred to clusters (although not seen in UV Vis), the first raise could not be explained in that way. We suggest that the particles aligned inside the channel with higher concentration (at the peak of the curve), leading to smaller hydrodynamic radii and, therefore, smaller sizes at the center of the peak. However, further investigations have to be done to prove this suggestion.

II.3.3 Internal Energy by mDSC Measurements

Measurements of stretched polymer particles inside a modulated DSC (see Appendix VIII.1.7) revealed not just the glass transition temperature T_g but also energy contributions related to the shape of those particles. While bulk polymers would only show an endothermal increase at their T_g (Figure II-12 orange reverse heat flow curves), polymers in particle shapes additionally loose surface areas above T_g and, therefore, exothermal energy is set free (Figure II-12 blue areas at non-reverse heat flow). Furthermore, internal stresses, which might occur from stretching, could contribute to this exothermal release (thermomechanic energy release).

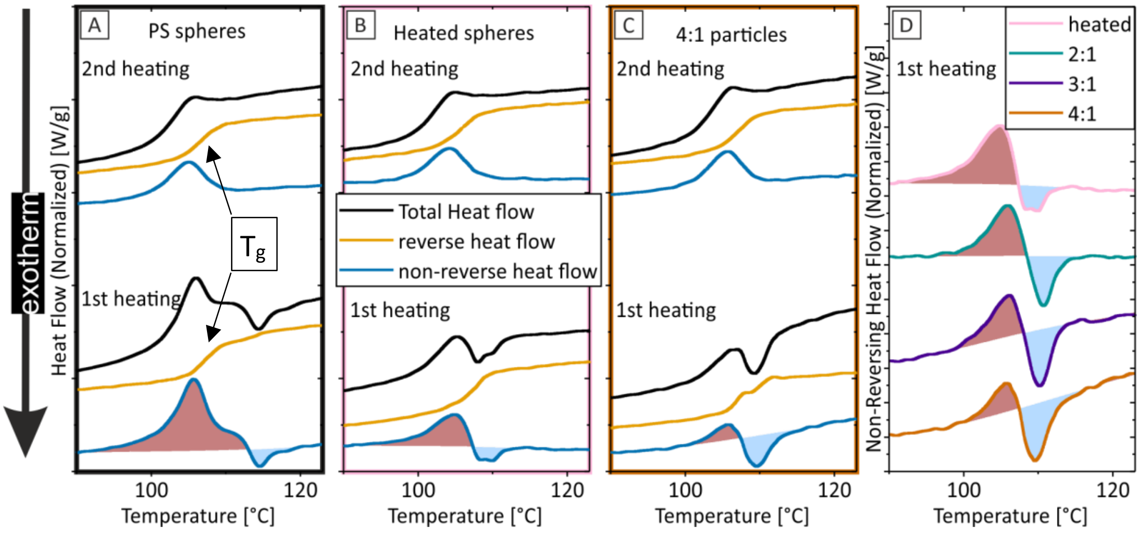


Figure II-12. "Modulated DSC measurements. A, B, C show the first and second heating cycles of three different samples with the total (black), reverse (orange [, T_g of PS visible in graphs as midpoint of step, approx. 107 °C (lit. values for PS: 107 ± 2 °C [86], see Appendix VIII.1.7.2]) and non-reverse heat flow (dark blue) signals. Diagrams of individual signals are y-shifted and ordered for better visualization. The first heating cycle contains contributions from the polymer conformation relaxation and the loss of surface area. The second heating cycle resembles bulk PS for all specimen since the particulate structure has been erased. D: The non-reverse heat flow signals of the first heating cycles display the systematic increase of the exothermal (downwards, blue) relative to the endothermal (upwards, red) contribution." The figure here is an enhanced version of figure 2 of my paper [1]. Used with permission of Royal Society of Chemistry, from Benke et al. [1] ©2024; permission conveyed through Copyright Clearance Center, Inc.

Original particles prior and after heating showed different non-reverse heat flow signals at their 1st heating, as the signal of the 1st heating depends crucially on the thermal history of the investigated particles. For the ellipsoidal particle samples, the ratio of the exothermal to the endothermal peak area of the 1st heating (Figure II-12 D red to blue area) increased with the AR. However, as higher stretched particles have also more surface area and, therefore, more energy to be set free when particles sinter/melt together, the contribution of the exothermal energy release by the relaxation of polymer chains in higher entropy conformation could not be separated from the contribution of surface area loss.

The second heating curves show no difference when compared to each other. That means that the thermal history was erased from the individual samples. The samples became the same prior to the second heating cycle. The different surface areas and internal stresses are not present anymore and the samples behave like bulk polystyrene with a T_g at around 107 °C (lit. value for PS: 107 ± 2 °C [86], see Appendix VIII.1.7.2).

II.3.4 Recover Surface Charge – Cleaning Particles

After dissolving the foil, the particles had residuals of PVA on their surface. As the surface properties of particles are important for self-assembly, it was necessary to get as much PVA off the particles as possible. The cleanliness of the particles was monitored by zeta potential and XPS measurements (methods explained in Appendix VIII.1.4 and VIII.1.5).

II.3.4.1 Developing the Standard Cleaning Protocol

With every washing step, the zeta potential of the particles recovered more (Figure II-13).

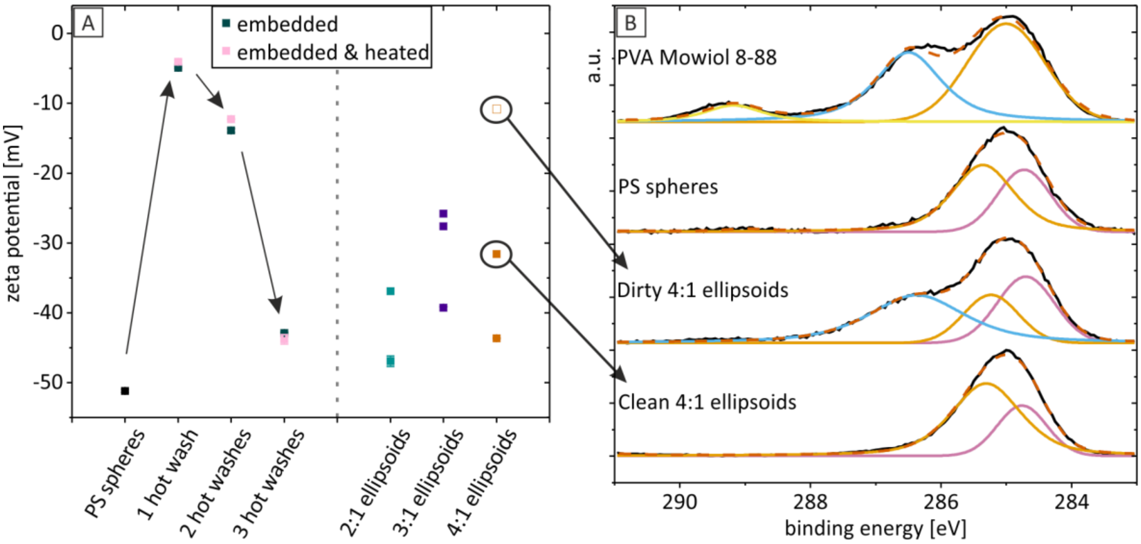


Figure II-13. Combination of zeta potential and XPS measurements. A: Zeta potential measurements of different samples origin from particles FN269. The here discussed zeta potential values were taken after x times of hot washing with an iPrOH:water mix followed by 3 times rinsing with water to avoid errors by residual iPrOH (details of measurement see Appendix VIII.1.4.1). "B: Deconvoluted C 1s spectra of PVA, Polystyrene original particles, heated and stretched particles right after recovering from foil (dirty 4:1 ellipsoids), and the respective cleaned particles (clean 4:1 ellipsoids). Deconvolution of peaks showing saturated hydrocarbon groups (C–H; purple and orange), alcohol groups (C–OH; blue) and acetyl groups (COOH; yellow). The spectra were normalized to the height of the C 1s maximum and shifted so that C 1s maximum was at 285 eV" [1]. The same figure can be found in my paper [1] as figure 5. Used with permission of Royal Society of Chemistry, from Benke et al. [1] ©2024; permission conveyed through Copyright Clearance Center, Inc.

The zeta potential recovery is much more effective at the hot washing steps with iPrOH:water mix than at plain water rinsing steps. This did not change even when the particle solution was heated during water washing steps as well (see supporting information II.5.3). It was, therefore, important to use iPrOH in the washing procedure. In general, after embedding the PS particles in PVA matrix, the zeta potential weakens in comparison to the original particles, leading to the assumption that some of the PVA was incorporated into the particle surface. However, the PVA does not seem to be permanently incorporated during the step "heating of the foil to 150 °C" as with sufficient cleaning, the zeta potential of just "embedded" and the "embedded & heated" particles can become the same. Nevertheless, the charge stayed weaker (around -44 mV) as the one of the original particles (-53 mV) and could not be recovered to the fullest; not even after multiple additional washing cycles with iPrOH mix or water. The zeta potential measurements of the stretched particles could be compared to the spherical ones. Although the sample-to-sample variation for the cleaned ellipsoidal particles (type 1, FN269) were quite strong (examples for zeta potential of differently stretched particles see Figure II-13), stretched particles with a zeta potential of up to -47 mV could be recovered from the foil with a 3-hot-washes cleaning protocol.

The values received after the 3rd hot washing step could be changed neither with additional washing steps, nor with washing at higher temperature (also not at samples with low zeta potential). We, therefore, claim that three hot washing steps were sufficient for cleaning the particles for further use.

In literature, Rey et al. [92] showed that particle dispersions are getting dirty with time. Therefore, the thorough cleaning of the particles after preparation of the ellipsoids is important, but might never be completed. For using the particles within assembly studies (Chapter V), it is recommended to clean the dispersion prior to use again.

II.3.4.2 Residual PVA Checked by X-Ray Photo Spectroscopy and STA

To investigate the particle surface during cleaning, x-ray photo spectroscopy measurements of the used PVA and some particle samples were performed without sputtering prior measurements to investigate the actual surface (Figure II-13). At the deconvoluted PVA C1s peak I identified the aliphatic C atoms at 285 eV and the C atoms with hydroxyl groups at 286 eV (literature value 286.52 eV [93]). As the manufacturer of the PVA (Mowiol 8-88) explains that the used PVA is just 88 % hydrolyzed with impurities of 10.0 – 11.6 % residual content of acetyl, the presence of a weaker peak for O–C=O and C=O around 289 eV can be explained. The pronounced C-OH band (binding energy in the area of 286 eV) does act as an identifier for the PVA. The important peaks for polystyrene are the aromatic and aliphatic C atoms at 285 eV and 286.52 eV [94].

To investigate leftover PVA on the particles, a sample of stretched particles was measured right after recovering the particles from the foil (Figure II-13 dirty, one hot wash, transferred to water 3 times) and after cleaning with the standard protocol (3 hot washes and then transfer to water). By the deconvolution of the respective C1s peaks, the just recovered sample with a zeta potential of -11 mV showed the pronounced C-OH band of PVA on the particles. However, after cleaning, the XPS did show no residual PVA on the particles' surface (C-OH peak vanished), although the zeta potential of that sample with -32 mV was not the maximum recovered zeta potential for particles after cleaning. The cleaned particles showed nearly the same XPS signal as the original particles, while the original particles had a zeta potential of -53 mV. The difference in the zeta potential showed that the proposed residual layer of PVA from zeta potential estimations on cleaned particles could not be resolved by XPS and should, therefore, be rather thin.

Simultaneous thermal analysis (STA) measurements could not show any attached PVA on the cleaned particle surfaces as well. This could result from the fact that the pronounced small layer of PVA on the cleaned samples was presumably not in the detection range of the instrument (supporting information II.5.4).

II.3.4.3 Cleaning of Big Particles

The cleaning protocol in this chapter was developed with type 1, PS-AA, particles (FN269). Therefore, the cleaning protocol was later re-checked with type 3 particles (PS-PSS, DB particles). As those particles had a different surface functionalization (sulphate instead of carboxyl groups), the cleaning protocol worked even better. The surface charge was normally less than -60 mV for cleaned spherical PS-PSS particles (original -63 mV) and less than -55 mV for ellipsoids. The sample-to-sample variations were low.

II.3.4.4 Cleaning Protocol used in Centrifuge Tubes

Later on, the hot washes of the particles were changed from washing inside a flask, to washing inside the centrifuge tube with ongoing sonication. Additionally, a 4th hot washing step was introduced. This change in the treatment did not affect the zeta potential recovery (Table II-6). However, the time necessary for cleaning the particles decreased drastically. The cleaning protocol was later further adapted into washing at different pH values (Chapter III.1.3).

Table II-6. Comparison of cleaning in a round-bottom flask (glass) and a centrifuge tube (Falcon tubes, plastic). All hot washes were 3 times rinsed with water prior to the zeta potential measurements. Data was gathered from FN270 (PS-AA, type 1) particles by Friederike Ganster during her bachelor thesis 2021.

step \ zeta potential [mV]	heating in flask	heating in tube
1st hot wash	-13	-12
2nd hot wash	-14	-12
3rd hot wash	-31	-37
4th hot wash	-43	-43

II.4. Summary of Chapter II

In this chapter, firstly (Chapter II.1) the particle synthesis of spherical polystyrene particles is explained. All synthesized particle batches (type 1 – 4) used within this thesis are depicted with their major properties (size, coefficient of variation and zeta potential). For successfully self-synthesized particles of type 3 (PS-PSS), a detailed synthesis protocol is shown and the dependency of different parameters on the resulting particle size was investigated briefly.

Secondly (Chapter II.2), the spherical particles were transformed with the uniaxial stretching method to ellipsoidal particles. For applying the method, a homebuilt stretching device was built up and the major properties and advances of the process and the device are depicted.

Finally (Chapter II.3), the basic results for the stretching process are shown. The applied strain did show a linear dependency within the investigated region of ARs (AR 1 – 4). Therefore, it was possible to forecast the AR at an applied strain or to target a specific AR by applying a certain strain. The reproducibility of the process was very high and the samples dimensions did not show an effect on the resulting AR despite the same strain.

The AR-strain dependency was individual to differently sized particles, while large particles (type 2 – 4) needed a lower strain for the same AR as small particles (type 1).

The stretched particles showed volume conservation. Nevertheless, they might not be described perfectly as mathematical ellipsoids with two different axes, as the diameter of the volume corresponding sphere increases with AR. Compared to mathematical ellipsoids, the real particles might be pointier or deformed in shape (see Chapters IV.6.4 and IV.8).

The particles of type 1 were investigated with light scattering techniques, revealing clusters in the embedded spherical particle samples. The stretched samples did not (yet) show the clusters. Thus, the maximum to-be-used amount of particles of type 1 inside the foil was defined to 16.7 wt% PS in dry foil (PS : PVA 1 : 5 w/w).

The internal energy of the particles showed an AR dependency, though it was not possible to separate internal energy contributions of the stretched polymer chains from the contribution of surface area loss.

A sufficient cleaning protocol for recovering the particles from foil was developed and checked via zeta potential and XPS measurements. After three hot washing steps, no residual PVA was found by XPS and the zeta potential recovered almost completely. The cleaning protocol was further simplified by applying it to samples inside centrifuge tubes.

II.5. Supporting Information to Chapter II

II.5.1 SI for Chapter II.3.1.3 – Further Strain Values

The gained ARs with their applied strain for differently sized oven-stretched particles is shown in the following Table SI II-1. Further data of particles stretched with the roll-stretching device can be found at Chapter III.2.3.

Table SI II-1. AR to strain from oven stretched foils. Additional datasets collected by me and my supervised students a) Friederike Ganster, during her bachelor thesis, and b) Tobias Scheibel, B.Sc., during his internship within his master studies. The foils of b) (FN294 and DB14) were thinner than the other oven stretched foils (from 400 µm doctor blade, normally 0.6 – 1 mm) and contained CsCl. They showed low ARs around 100 % strain, while the small particles (FN294) remain lower in AR than the big particles (DB14). The particles b) were additionally stretched via the roll-stretching device (Chapter III.2.3).

particle type		**name**	**SEM size [nm]**	**selected ARs (with applied strain)**				
Type 1	PS-AA	FN269	326 ± 7		2.0 ± 0.1 (36 %)	2.2 ± 0.1 (43 %)	3.2 ± 0.2 (75 %)	4.1 ± 0.3 (108 %)
		FN270[a)]	337 ± 8	1.5 ± 0.1 (19 %)	2.0 ± 0.1 (35 %)	2.2 ± 0.1 (45 %)	3.1 ± 0.2 (75 %)	4.0 ± 0.3 (106 %)
		FN294[b)]	254 ± 6	1.4 ± 0.1 (15 %)		2.3 ± 0.2 (51 %)	3.2 ± 0.3 (80 %)	3.5 ± 0.3 (99 %)
Type 2	PS+	AL166	1409 ± 18		2.1 ± 0.2 (36 %)	2.4 ± 0.3 (44 %)	3.6 ± 0.5 (85 %)	4.2 ± 0.5 (108 %)
Type 3	PS-PSS	DB07	1060 ± 27	1.4 ± 0.1 (14 %)	1.7 ± 0.1 (25 %)	2.5 ± 0.2 (49 %)	3.3 ± 0.4 (73 %)	4.3 ± 0.1 (108 %)
		DB14[b)]	1387 ± 19	1.4 ± 0.1 (14 %)	2.1 ± 0.2 (39 %)			3.7 ± 0.4 (99 %)
bought	(PS-PSS)	INVI2000	1899 ± 57		1.7 ± 0.1 (25 %)	2.6 ± 0.2 (51 %)		4.6 ± 0.3 (110 %)

II.5.2 SI for Chapter II.3.1.4 – Corresponding Sphere Error

Equations to calculate the corresponding sphere diameter error:

$$\Delta d = \sqrt{\left(\frac{\partial d}{\partial l}\cdot \Delta l\right)^2 + \left(\frac{\partial d}{\partial w}\cdot \Delta w\right)^2} \qquad \text{(Equation SI II-1)}$$

$$\Delta d = \sqrt{\left(\left(\frac{1}{3}\cdot\left(\frac{w}{l}\right)^{\frac{2}{3}}\right)\cdot \Delta l\right)^2 + \left(\left(\frac{2}{3}\cdot\left(\frac{l}{w}\right)^{\frac{1}{3}}\right)\cdot \Delta w\right)^2} \qquad \text{(Equation SI II-2)}$$

II.5.3 SI for Chapter II.3.4.1 – Zeta Potential during Washing

The zeta potential was recorded after every washing step (Figure SI II-1). The values for Figure II-13 were the values of “in water stored” for the 1 – 3 hot washes (normal protocol). The displayed steps (Figure SI II-1) included heating in a sonication bath while getting the particles back in dispersion after centrifugation. The data without water rinsing steps has to be discussed carefully as residual PVA and iPrOH might be floating freely in the solution.

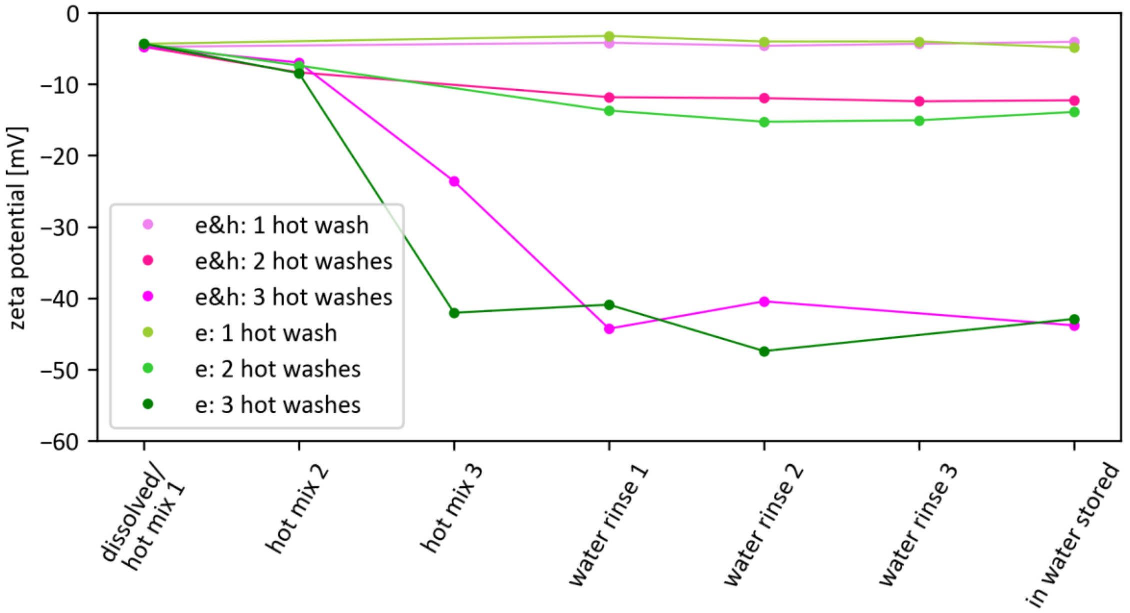


Figure SI II-1. Zeta potentials at different steps of particle cleaning. Embedded and heated samples (e&h) and just embedded samples (e) showed that rinsing with water did not alter the zeta potential much. For a significant change, the particles had to be washed with iPrOH:water-mix.

II.5.4 SI for Chapter II.3.4.2 – STA Measurement

The simultaneous thermal analysis (Figure SI II-2) was performed by Dr. T. Feller and is displayed in terms of completeness. The STA coupled a TGA with IR and showed that no PVA signal could be detected on the cleaned “Embedded & Heated” particles.

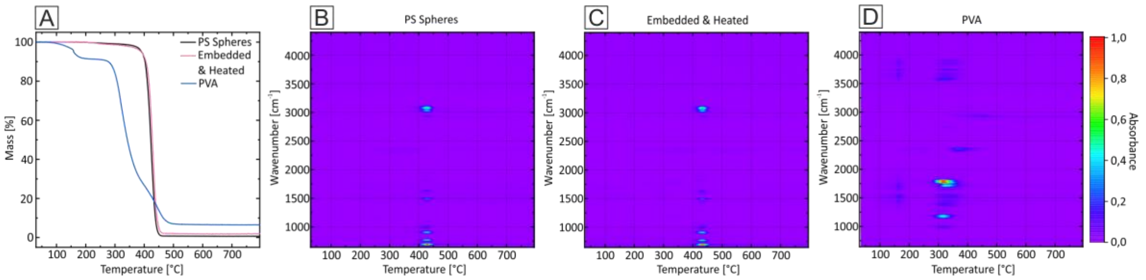


Figure SI II-2. STA analysis with A: mass loss diagram of the TGA and B – D temperature dependent detector signal of the coupled IR spectrometer. The “Embedded & Heated” sample was similar to “PS Spheres” as it did not show PVA bands. Diagrams and data by Dr. T. Feller.

Chapter III. Advances and Scale Up

The ellipsoidal polymer particle manufacturing method introduced in Chapter II.2, making foils batch-wise on glass plate, cutting them by hand and process them by hand-stretching in the convection oven is certainly not feasible for industrial applications. The process is time consuming and produces many waste particles in foil pieces with unregularly stretching or dimensions (cutting losses). As seen in Chapter II.2.1, the whole process of preparing ellipsoidal polymer particles can be split into three different steps. The first step is the embedding of particles in foils, the second step is the stretching of the foils and the third step is the recovering of the stretched particles. In this chapter, the upscaling of the first two steps is displayed.

Furthermore, the evaluation of multiple particle samples needs an "upscaling" as well as the particle images from SEM are evaluated by hand (in Fiji/ImageJ) in a time consuming effort.

Structure of the chapter

For the upscaling of the preparation process, long particle-loaded foils (Chapter III.1) are made that can then be stretched (Chapter III.2) in an up-scaled, semi-continuous mode, similar to Ferrar et al. [95]. Therefore, two devices are employed. Firstly, a semi-automated semi-continuous foil doctor blade is implemented to produce m-long foil ribbons. The device and the foil manufacturing with the device are explained. Secondly, a home-built roll-stretching device is recommissioned to stretch the long foil ribbons. The setup and the resulting particles at different processing parameters are shown. The two devices are adjusted to have the possibility to combine the processes to a one-step liquid-to-roll continuous process.

For upscaling the particle image analysis, I developed a particle evaluation program in Python3 (Chapter III.3). With this program, it is now possible to evaluate SEM images of multiple 100 spherical and ellipsoidal particles within a minute. As I extracted the particles' dimensions as well as their orientations, the program was indispensable for the evaluation of the monolayer assemblies of Chapter V.

The chapter closes with a summary and outlook (Chapter III.4).

III.1. Automated Foil Doctor Blade Device

The primary idea for scaling up the process was to make foil ribbons of infinite length. Fortunately, a decommissioned continuous doctor blade device could be lent from the chair of Macromolecular Chemistry I, University Bayreuth (see supporting information III.5.1). This device was then recommissioned and enhanced, with the help of the mechanical and electronic workshops of University Bayreuth, to the final version of a semi-automated semi-continuous foil doctor blade.

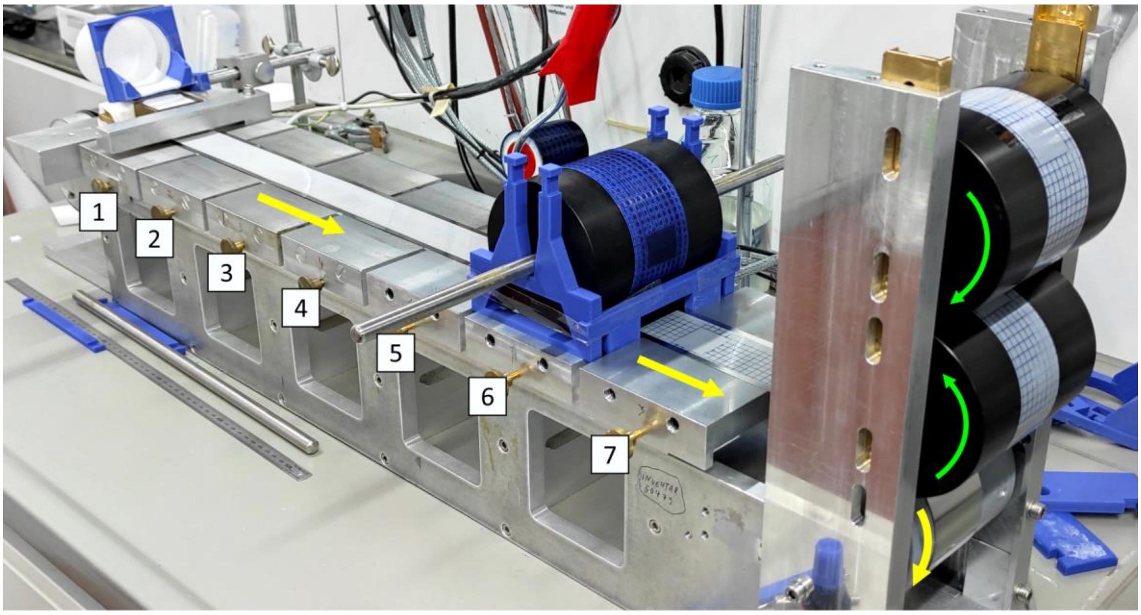


Figure III-1. Photograph of the latest expansion stage of the semi-continuous foil doctor blade while producing a particle loaded PVA foil. The slurry was filled directly from the mixing beaker over a 3D printed ramp/holder into the brass-frame doctor blade. The steel strip was moved (yellow arrows) under the fixed brass-frame (plate 1), over the heating plates (in this configuration only plates 2 – 7 were in use), and rolled up on the motorized cylinder (lowest on the right). The slurry dried until the 4th heating plate and was stamped by the rolling-stamp-cylinder at plate 6. Plate 7 dried the stamp ink and the foil completely so that it could be detached. The foil was then rolled onto the upper black cylinder automatically (green arrows indicate their direction of movement). Between the heating plates are gaps filled with magnets to flatten the steel strip.

The final design of the device (Figure III-1) consists of a supporting steel strip, which is pulled from an un-motorized cylinder to a motorized cylinder passing heating plates. The PVA dispersion is doctor bladed on the steel strip to equal height, then dried on the heating plates, and recovered as a long foil ribbon before the steel strip rolls up again.

The robustness of the particle-in-PVA-foil embedding process with the semi-automated foil doctor blade device and its easy use could be proven as several students worked with the device and produced meters-long, even-looking, bubble free foil ribbons.

In the following sub-chapters, I will explain the preparation process and the different parts of the device. I included my solutions to different challenges that arose by using the continuous foil doctor blade.

The foil ribbons made by the doctor blade were further processed with the hand-stretching device (Chapter II.2.1.2) or the roll-stretching device (Chapter III.2).

III.1.1 Preparation of Meters-Long Particle Loaded Foils

The main type of particles used in the chapter were self-made monodisperse PS-PSS particles from dispersion polymerization in the range of 1 µm (Type 3, see Chapter II.1.3.3). However, the foil doctor blade device was as well successfully used with other type of particles (Type 1, 2 or bought ones).

For preparing a foil, a bubble free dispersion of PVA (Mowiol 8-88) and PS particles was prepared, similar to before, via speed-mixing (see Chapter II.2.1). The PVA/PS dispersion contained around 30 wt% of PVA (adjustable due to water evaporation by speed mixing under vacuum) prior to pouring it onto the supporting 5-cm-wide precision-thickness-gauge strip (h+s Präzisionsfolien GmbH, Chapter III.1.2.1) in front of a brass doctor blade or into a frame doctor blade with 4 cm slit width (Chapter III.1.2.2).

The supporting strip moves the dispersion under the doctor blade with adjustable speeds (Chapter III.1.2.3) and the wet film height is set by the slit height of the doctor blade. The PVA dispersion is dried by pulling it on the supporting strip over several heating plates. The plates were exhibiting an increasing temperature profile (Chapter III.1.2.4). The temperature of the last plate was set in a way to finalize the drying of the foil.

The resulting foil was then peeled off around 10 cm after the last heating plate (temperature of foil back to room temperature) by a rotating cylinder (diameter 10 cm), and rolled immediately on another cylinder of the same size (Figure III-1 black cylinders on the right). The cylinders were designed to contact the supporting strip only by weight on top of the foil. Therefore, the speed of the cylinders was the same as the foil, even when the foil is rolled up and the cylinder diameter increases by the foil. With this design, no additional pulling forces on the foil were applied.

The film was then transferred and stored on a cylinder spool (Häfner & Krullmann GmbH).

It was noted that the foil reacted to the humidity of the storing drawer. When the humidity was up, the foil was more flexible.

III.1.2 Parts of the Device

The different parts of the device are explained to give other scientists, who build up a semi-automated semi-continuous foil doctor blade device, some more information on the developed device and on the process.

III.1.2.1 Supporting Strip

The previous process of making the PVA foils in Chapter II.2.1 was using glass plates as a substrate and long drying times (1 – 2 days). This worked suboptimally as sometimes the PVA film got too dry and then it was not possible to detach it from the glass anymore. Especially the difference in air humidity and temperature between summer and winter was challenging and could not be compensated completely, not even when drying the PVA film in a closed fume hood without ventilation. The continuous doctor blade was, therefore, designed that the drying of the film is rapidly done by heating the foil from beneath. Thus, the supporting strip had to be heat conducting and the PVA foil should not stick to it. Additionally, the drying PVA deals with shrinking forces. The used support needed to be stiff enough to counter such forces.

The easiest way to have a stable but also still flexible support, which could be easily bought at a supplier, was a steel strip.

For a real continuous foil preparation, the supporting strip would need to form a loop and the device was actually equipped with such a steel loop/belt in its standard configuration (Figure III-2). However, in the end I avoided using the steel belt configuration as even the carefully LASER-welded and polished connection of the steel strip ends (to form a belt) gave an imprint on the prepared foil. Used steel stripes for making new belts: stainless steel, 1.4310, 5 m × 10 cm × 0.1 mm from Hasberg-Schneider GmbH; laser-welded at: Huber-tec GmbH; further polishing at the mechanical workshop NW I at University Bayreuth.

The eventually used steel strip is pressed to the heating plate by magnets, which was a solution to reduce buckling of the steel strip due to uneven heating (see Chapter III.1.2.1.1). Additionally, the width of the steel strip was reduced to the width needed for preparing foils that could be used within the stretching device of Chapter II.2.1.2.

For the final design, I chose a 10 m × 5 cm × 0.1 mm spring band steel strip of stainless steel 1.4310 from h+s Präzisionsfolien GmbH (1500-1800 N/mm^2). To avoid the sticking of PVA on the steel, I rubbed the steel with Ballistol (Ballistol GmbH) to form a separating oil layer in the final process. The steel strip of 10 m length gave access to 8.5 m long foil ribbons.

III.1.2.1.1 Buckling and Sticking

Using metal as a support gave the challenge that metal expands when heated up. Unfortunately, the heat expansion was different in the part where the to-be-dried PVA dispersion was upon, and in the part where no PVA was on top. This was especially a problem when using the 10 cm width steel belt together with lower width doctor blades (4 – 6 cm). The sides of the steel belt were expanding more and with the combination of pulling forces, the steel belt started to get wavy over time. Unfortunately, the buckling stayed also when the steel belt cooled down. To suppress the buckling, I tested applying some pressure on the rim next to the foil during drying (Figure III-2). However, this was not successful as tension forces lifted some middle parts of the supporting belt above the heating plate.

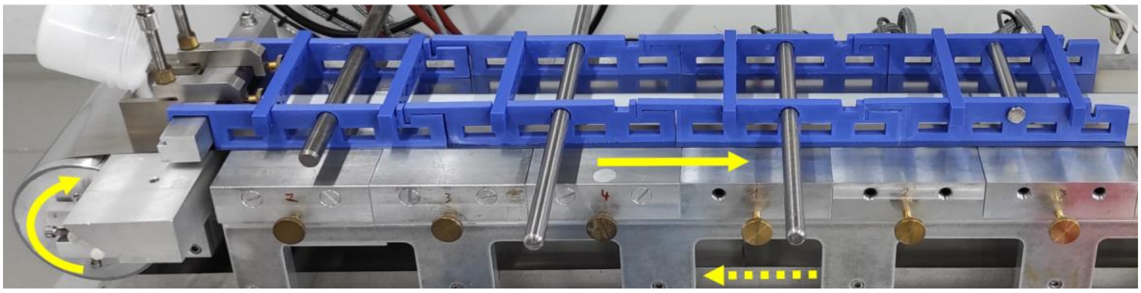


Figure III-2. The continuous foil doctor blade during foil preparation. 3D printed frames (additional weight by steel rods) pressed the continuous steel belt on the heating plates. Arrows indicate the belt's moving directions.

A solution was found by putting cuboid neodymium magnets (N35 5 cm × 1.4 cm × 6 mm, magnets4you GmbH) between the heating plates (Figure III-1). They flattened the strip at least where the foil dried. Additionally, the width of the steel strip was adjusted to fit the slit width of the doctor blade so most of the steel strip was covered with drying PVA (new 4 cm width doctor blade on 5 cm steel strip).

Furthermore, a hardened C-steel strip (material 1.1274, h+s Präzisionsfolien GmbH, 10 m × 5 cm × 0.05 mm) with a stronger magnetic interaction than stainless steel was tested. Due to the stronger magnetic interaction, the thickness of the strip could be reduced to 0.05 mm. The lower thickness reduces tension forces from heating.

With this C-steel strip, my supervised student Friederike Ganster produced good-detachable foil ribbons during her bachelor thesis (using a 600 µm frame doctor blade, around 30 wt% PVA dispersion, and 50 mm/min drying speed). The ribbons showed homogenous opacity (resembling same thickness) over the whole length (3 – 4 m) and no buckling of the strip occurred. However, with the repeatable use, the C-steel strip surface changed and started to rust (starting after the 4th foil). The spots of rust let the PVA foil stick very tightly to the steel, causing rupture. Sometimes the PVA foils, which were still detachable, got yellow from the rust (see Chapter III.1.3).

Due to the described rust problem with the C-steel strip, I finally used a stainless steel strip with 0.1 mm thickness (10 m × 5 cm × 0.1 mm) again as rust was no problem there. However, the magnetic interaction was not so strong. Thus the strip was lifted at some areas slightly (0 – 1 mm) from the heating plate due to thermal tension forces. Nevertheless, with this setup, it was as well possible to produce foil ribbons with homogeneous opacity that could be used in the following. To avoid the sometimes still occurring sticking of the foil, the strip was rubbed with oil (Ballistol).

III.1.2.2 Doctor Blade

The doctor blade was specially designed to fit on the 5 cm wide steel strip. Requirement was that the dispersion did not spill out on the sides nor in the back and additionally, the doctor blade should not scratch the steel strip. The firstly used steel-frame doctor blades, which were held in place by spring screws (Figure III-3 A), worked well in countering the leaking of the dispersion. However, the steel scratched the supporting strip surface. Thus, multiple frame doctor blades with slit heights from 100 µm to 1 mm were manufactured from brass (Figure III-3 B), a softer than steel material. The brass-frame doctor blades were mostly used without additional pressure by spring screws. The slit width was set to 4 cm (outer width of frame 5 cm) fitting the stretching device of Chapter II.2.1.2. The frames had a height of 2.5 cm and length of 2 – 3 cm.

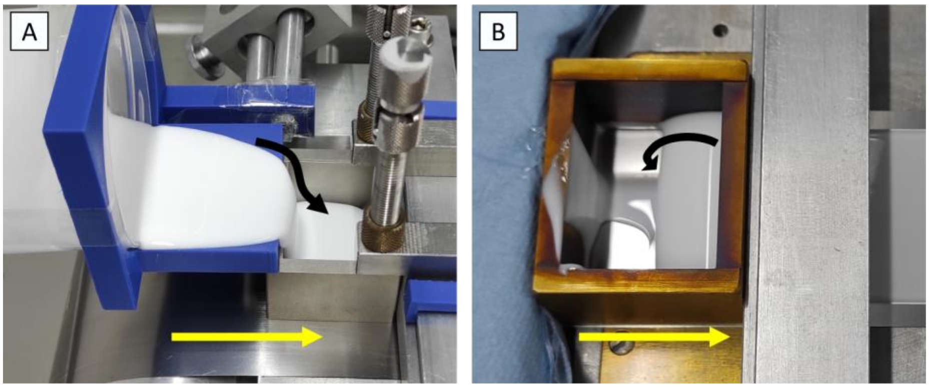


Figure III-3. Photographs of different frame doctor blades. A: Steel-frame doctor blade during filling with PVA/PS dispersion over one side of the frame (here before foil preparation, where strip was not yet moving in direction of yellow arrow). B: Brass-frame doctor blade at the end of foil preparation where the liquid roller turbulence was visible. The yellow arrow indicates the movement of the supporting strip. A, B: Black arrows indicate the movement of the dispersion.

Frame doctor blades were chosen since they have the advantage that the dispersion can be filled inside the frame prior starting the strip movement and the dispersion could not spill to the sides nor the back (compared to bar type doctor blades). However, inside the frames a liquid roller turbulence formed (Figure III-3 B) when the strip was pulled under the frame, leaving it necessary to fill in the dispersion without bubbles. This could be achieved by slowly pouring it over one side of the frame to the ground (Figure III-3 A).

The dispersion in the frame exhibit a certain squeezing pressure on the slit opening, making the final wet film height dependent on the filling height inside the frame doctor blade. Therefore, to have the same thickness in one foil ribbon sample, care was taken that the dispersion was filled to the same level (mostly until the top) of the frame and steadily re-filled by some dispersion while foil making was in progress. The filling and refilling of the frames was done with a 3D printed adapter to the mixing beaker (Figure III-3 A, final design see supporting information III.5.2).

III.1.2.3 Wet Film Height – Drying Speed, wt% and Resulting Foil Thickness

The different wet film heights from different doctor blade slit heights resulted in different drying speeds of the foils. As the heating area was limited to around 60 cm, the speed of the strip needed to be reduced drastically for large slit heights. This, however, led to leakage, when the PVA content (wt%) was too low (low viscosity).

An optimal wt% found for the mostly used doctor blade slit heights of 400 µm or 600 µm was around 30 wt% of PVA (Mowiol® 8-88, Mw ~67000, 86.7 – 88.7 mol% hydrolysis) in the total mixture. The 30 wt% PVA was easily reached by using the speedmixer under elongated vacuum conditions decreasing the water content of the mixture (starting from > 20 wt% PVA in the mix). However, when using the speedmixer for water content reduction the pump was protected from the water by a freezing trap or/and a membrane pump was used. Additionally, I have to mention that the PVA solution was getting warm when speedmixing with high wt% as the viscosity increased. Note here that the former manufacturer of Mowiol declares the viscosity for a 25 % mixture of pure PVA 8-88 in water at 20 °C with 10 Pa·s [96]. The viscosities of the doctor bladed dispersions, however, were not investigated, but could be a subject for future research.

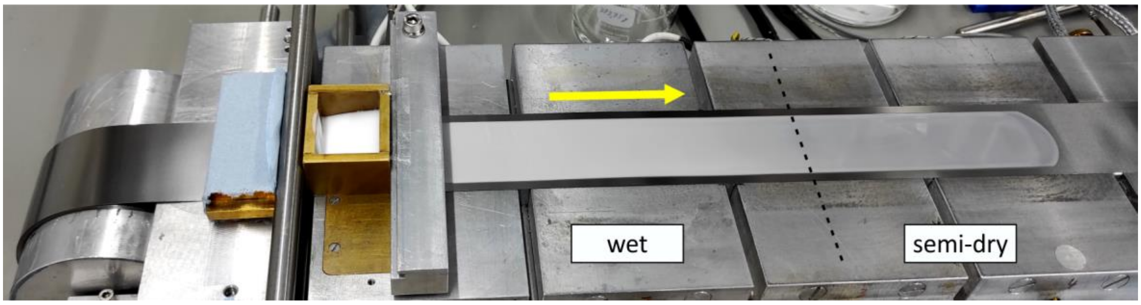


Figure III-4. Photograph of the film doctor blade with 30 wt% dispersion. The stainless steel strip is cleaned from excess Ballistol right in front of the frame doctor blade. The wet foil exhibited a brighter white reflection as the semi-dry foil; magnets are visible between the heating plates. The brass sheet beneath the doctor blade was required for helping the steel strip pass the area without scratching. The doctor blade could be pressed on the supporting steel strip by spring screws but was normally used just sitting on top of the strip without additional pressure.

The best drying speed found for the dispersion with 30 wt% PVA was up to 100 mm/min for the 400 µm or 50 mm/min for the 600 µm slit height. With higher speeds, the foil was not completely dry at the point of peeling off and either stuck to the strip or deformed by pulling it off the steel strip.

For the 600 µm wet film height the resulting thickness with 30 wt% of total PVA (at mixture 9:1, 10 wt% PS in dry foil) was around 110 µm, while for 400 µm it was around 70 µm. Unsurprisingly, the foil thickness differed not just by using different slit heights, but also by different total wt% of PS particles in the dispersion (see supporting information III.5.3).

The foils from 600 µm wet film height were mostly cut into pieces and used for the normal hand-driven stretching device (process like Chapter II.2.1.2), while the foils from 400 µm slit height were used for the roll-stretching device (Chapter III.2) as the lower thickness ensured a better heat infusion.

III.1.2.4 Heating Plates and Final Drying

The heating plates of the semi-automated foil doctor blade device can be adjusted to individual heating temperatures. The definitively used temperatures of the six heating plates in use during the final setup (Figure III-1) were 75 °C | 78 °C | 80 °C | 83 °C | 85 °C | 90 °C. The final drying temperature of 90 °C was below the onset temperature of the glass transition of PS (see Appendix VIII.1.7.2) to avoid any deformation of the particles during drying or during the stripping of the foil. Note that when immediately applying 90 °C to dry the dispersion, the dispersion or wet film started to boil and bubbles appeared. With the given gradient heating, the foils were homogenous and did just show bubbles when they were already present in the dispersion. Note further that when the film was not dry at the site of the automated square-pattern stamp (Figure III-1 above plate 6) the foil was squeezed by the stamping roll and the pattern was carved into the foil, which led to uneven stretching results.

After drying, the foils were rolled up on a spool and stored in a lab drawer at room temperature (and uncontrolled humidity). This led to the bending of some stored foil ribbons to one side or another. An explanation for this behavior was not found, but it might origin from variations in end-drying while spooled on the cylinders. Especially no explanation was found as it did not occur consistently and the bending did not always appear on the same side (e.g. wall side of the device).

Note: As PVA is sensitive to the air humidity, a very dry foil can be brittle and stiff and might not be the best to use for (roll) stretching.

III.1.2.5 Stretch Markers

Especially for the use in the roll-stretching part, squares were stamped onto the foil. This was automated by 3D printing a square-patterned stamp (on an Ultimaker 3 with TPU as polymer) which was mounted on a cylinder. The rolling stamp cylinder applied the pattern on the freshly dried foil before the last heating plate so that the blue stamp ink (Kores Europe s.r.o.) could dry fast (Figure III-1 above plate 6). To apply the ink onto the stamp a paint roller was soaked with stamp ink and pressed onto the stamping cylinder. For normal stretching in the hand-driven stretching device, the ink was mostly not used as even after excessive washing of the recovered particles some blue residuals were left.

III.1.3 Final Cleaning Protocol – Yellow Foils and Blue Particles

During the bachelor thesis of my supervised student Friederike Ganster, the C-steel strip used at that time started to rust (see Chapter III.1.2.1.1). The resulting foils were yellowish. After stretching the foils in the oven and recovering the particles (type 1, PS-AA, FN270) with the standard cleaning protocol (Chapter II.3.4.1), the particles flocculated at the rinsing steps with water. This was very surprising, as the particle dispersions were not flocculating while being in the hot wash iPrOH:water mixture (3:7 v/v).

As it is generally known, ions can shield surface charges of particles. Therefore, I concluded that the iron ions of the rust influenced the colloidal stability leading to flocculation. To remove the iron ions from the solutions I altered the pH value drastically. Additional washing steps with 0.1 M $HCl_{(aq)}$ (pH 1) or 0.1 M $NaOH_{(aq)}$ (pH 13) did bring the particles back in dispersion. While the supernatant of the centrifuge with $HCl_{(aq)}$ was clear, it was yellow-brown with $NaOH_{(aq)}$. This might be explained by nanoparticles of iron (hydr)oxide forming in the alkaline solution [97, 98]. Although not performed here, further investigations on this proposed iron oxide particle formation might be done by using ultra centrifugation on the supernatant to separate the nanoparticles from the residues of PVA.

The washed particle dispersions (in water) of both ways were white again in the end, but zeta potential measurements by Friederike Ganster showed that the $HCl_{(aq)}$ cleaned particles did not recover their zeta potential. This might be a result of an additional flocculation of the negatively charged particles at pH < 2 (protonated surface groups) and thus insufficient removal of the PVA during cleaning. In contrast, the $NaOH_{(aq)}$ cleaned particles did recover their zeta potential due to a better electrostatic stabilization of negatively charged particles at high pH values (deprotonated surface groups) resulting in spacing of individual particles and sufficient cleaning. The zeta potential values from the particles after three additional pH washing steps were: HCl: -10 ± 0 mV and NaOH: -47 ± 1 mV. Therefore, I incorporated alkaline pH washing steps directly in my cleaning protocol.

Thus, the final cleaning protocol for the big particles (PS-PSS) in Chapter V (made by foil doctor blade and roll-stretching) was defined to be a centrifugal clean up with 4 times hot wash in iPrOH:$NaOH_{(aq)}$ mixture (iPrOH:1 M $NaOH_{(aq)}$ 3:7 v/v) inside a centrifuge tube standing at 60 °C in a sonication bath (37 kHz) for 30 min. The hot washes are followed by 3 times rinsing with pure water (again 30 min, 60 °C, 37 kHz) and a final filling up with pure water after centrifugation (values after rinsing, Figure III-5).

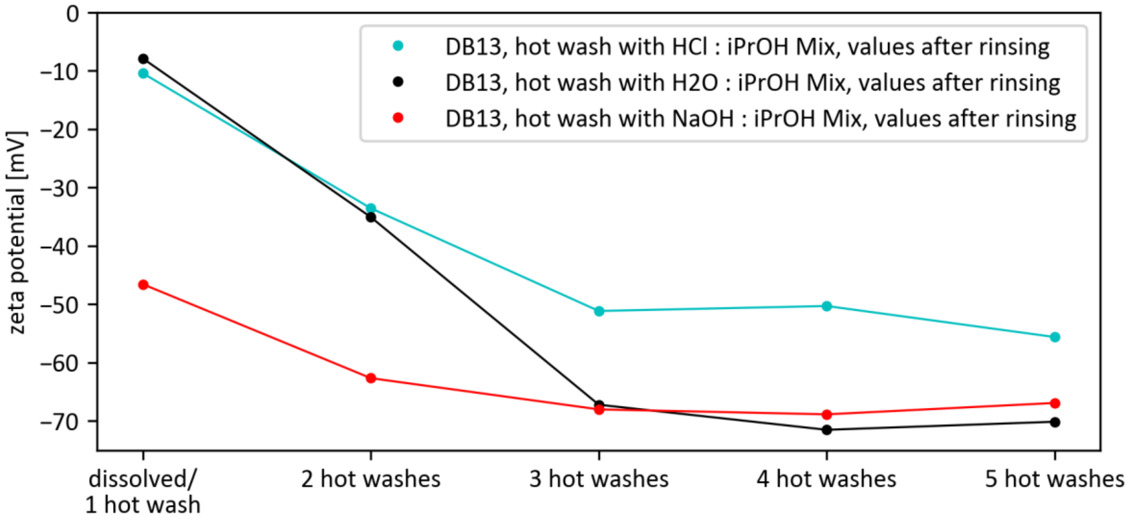


Figure III-5. Zeta potential plot of a study for the final cleaning protocol. Zeta potential was recorded for multiple hot washes with different pH dependent mixes. The NaOH-Mix showed an impressive zeta potential restoring already after dissolving the foil (1st hot wash). The standard mixture showed the previous explained zeta potential restoring after 3rd hot wash (standard protocol). Particles used in this study were spherical PS-PSS type 3 (DB13) with 1148 ± 20 nm.

As discovered later on, cleaning with $NaOH_{(aq)}$ also got rid of the majority of the blue stamp color of the automated stretch marker. In contrast, washing with $HCl_{(aq)}$ led to flocculation of the PS-PSS particles (Figure III-6). Therefore, all roll-stretched particles were recovered using the new cleaning protocol with hot washes of $NaOH_{(aq)}$. However, the first hot wash, so the dissolving step, was sometimes done with the standard mixture (iPrOH:water 3:7 v/v) as the foil dissolved quicker then.

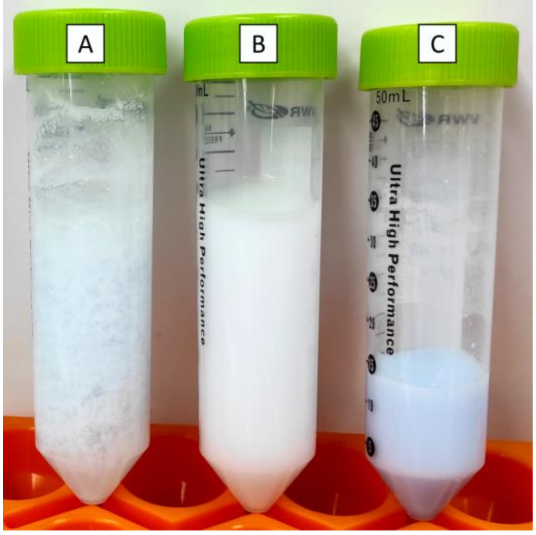


Figure III-6. Washing the particles from stamped foils with different solvents. A: 1 M $HCl_{(aq)}$ or B: 1 M $NaOH_{(aq)}$, C: dispersion before the special pH washing procedure (after cleaning with standard iPrOH mix). The blue ink of the stamp is colorizing the dispersion.

III.2. Roll-Stretching Device

The roll-stretching device was set up to speed up the stretched particle production. The basic features of the design are two cylinders turning at different speeds in the same direction, while the part between the cylinders could be heated by a heating plate/oven. In industry a similar process is used to produce films with better properties by film or strain hardening [99]. The device used here is based on two spool rollers with speed control, which was borrowed from the chair of Macromolecular Chemistry I, University Bayreuth.

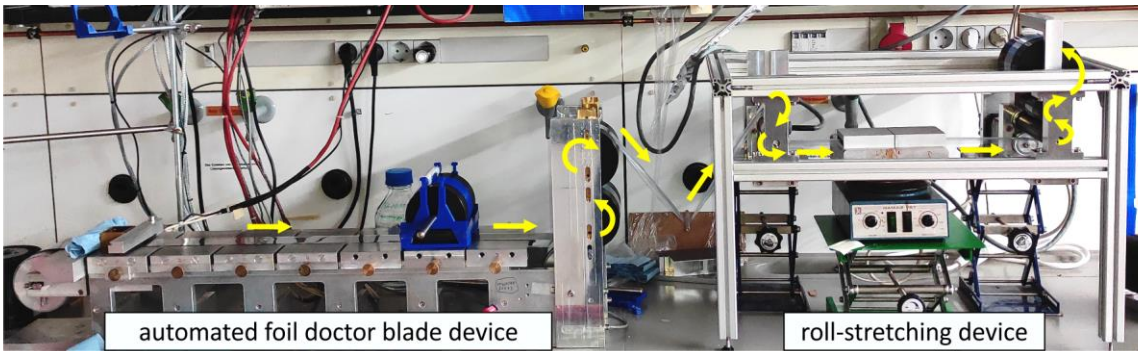


Figure III-7. Final setup of the roll-stretching device (with aluminium stand) directly connected to the automated foil doctor blade device. Here, a PVA foil without particles was produced from 30 wt% PVA solution and immediately stretched. Foil movement along yellow arrows. In between the two devices a metal cylinder is pushing the foil down (rolling along) to give the possibility to adjust the position of the foil on the onset of the stretching device.

Although the roll-stretching device could be used right after the continuous foil doctor blade without any delay (Figure III-7), the roll-stretching was normally done separately after foil manufacturing. Therefore, it was possible to use a variety of speeds for the foil movement in the roll-stretching device (not dependent on the drying speed of the PVA dispersion) and to solve errors independently. The setup of the roll-stretching device (final design Figure III-7 on the right) is explained in Chapter III.2.1. The parameters and challenges for using the device are discussed in Chapter III.2.2, while the resulting stretched particles made with different parameters are discussed in Chapter III.2.3.

III.2.1 Setup

The roll-stretching device consists of six rolls in total and a heating zone in between (Figure III-8). For better explanation, the movement of the foil and, therefore, the stretching direction is defined from left to right. In the final design, the rolls were mounted inside an aluminium stand for a higher stability (see Figure III-7 on the right and Figure III-12 B).

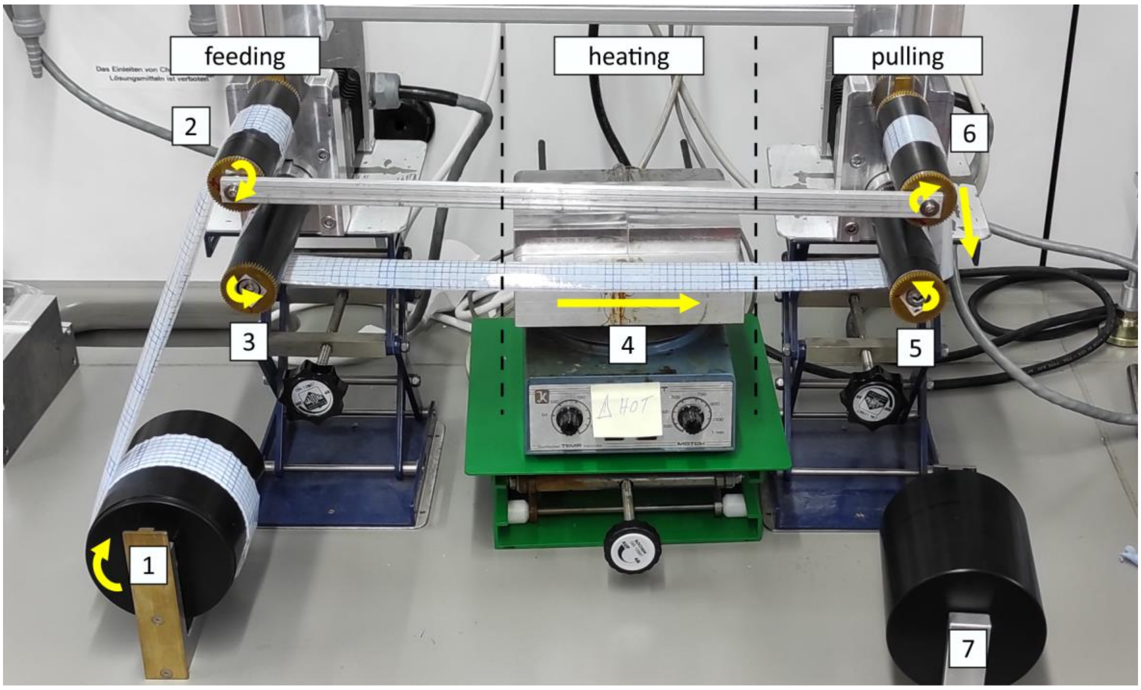


Figure III-8. The roll-stretching device before equipping it with the final aluminium stand (for stability) showing the foil path. In the picture the foil is halfway mounted. The upper rolls (2 and 6) needs to be pushed down to get the gearwheels in contact. The lower rolls (3 and 5) are motorized. Before starting, the heating plate (4) is pulled in the front so the middle part of the foil is heated. The big black cylinders (1 and 7) can either be standing on the table or be mounted on top of the gearwheel-equipped rolls. Yellow arrows show the future movement of the foil.

The three rolls on the left (Figure III-8 #1 – #3) can be defined as the feeding parts of the foil to the heating area. The first one is the storage roll (Figure III-8 #1) where the foil can be mounted prior to roll-stretching. The roll is not motorized and can also be dismounted for a better foil alignment. It can even be replaced by the storing spools where the foils might be rolled onto after foil processing or by the foil coming directly from the continuous foil doctor blade (in connected configuration). The foil rolls off just by pulling force onto the 2nd roll (Figure III-8 #2). The 2nd roll is the upper pulling roll. It is tightly connected by gearwheels and roll-to-roll rubber contact to the 3rd roll (Figure III-8 #3) which is the left driving roll. The foil is deployed in a mirrored S-shape around the 2nd and 3rd rolls to have maximum contact and to avoid slipping. To enhance sticking of the foil, the rolls surface consists of rubber (part of a bike tube mounted on a plastic roll) and can be even more sticky when cleaned by ethanol prior usage. The left driving roll is normally put to the device's lowest possible speed (1.3 rpm). The foil is fed from the top left on to the upper roll and by leaving the feeding part of the device at the lowest position in the right direction.

The foil is then pulled through a heating zone (Figure III-8 #4). The heating zone is a heating plate with an aluminium block mounted on top and two individual retractable aluminium covers (each 8 cm × 8 cm × 2 cm), creating a heating tunnel of approximately 1 cm height and 16 cm length (Figure III-9 A). The aluminium covers were sprayed inside with black paint (oven black) to maximize the heat radiation. During the movement through the heating zone, the foil is stretched (Figure III-9 B). The foil is not in contact with any part of the aluminium and is, therefore, just heated by radiation. In-contact experiments were also attempted but failed as the foil stuck and ruptured or got unevenly deformed.

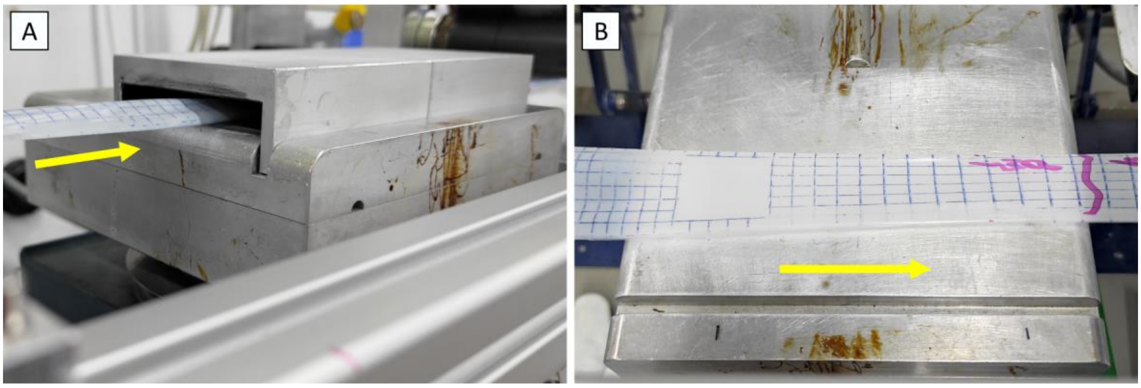


Figure III-9. Heating zone of roll-stretching device. A: The two aluminium covers create a heating tunnel. B: Part of the heating tunnel without the covers. Here, the foil is stretched while moving along (compare stamped pattern on left side squares to rectangulars on right). Foil movement indicated by yellow arrows.

The foil moves further to the right and is taken up by three rolls (Figure III-8 #5 – #7). The rolls on the right produce the pulling force on the foil while their arrangement mirrors the feeding part (left side). The foil is deployed in an S-shape around the 4th (lower, Figure III-8 #5) and 5th (upper, Figure III-8 #6) roll, which are gearwheel-equipped as well and have roll-to-roll rubber contact. The 6th roll is the final roll (Figure III-8 #7) and can be stacked on top of the 5th. As the 6th roll has just contact by weight and the foil is increasing the radius of the roll, the speed of rolling will be constant to the foil feed (same principle as with the rolls of the continuous foil doctor-blade).

In the final setup with the aluminium stand, both lower rolls, 3rd for feeding and 4th for pulling, are connected with timing belts (same toothed pulley sizes on engine and drive roll) to their respective engine to counter precession of rolls (Chapter III.2.2.4).

The engines driving the drive rolls are capable of high speeds (> 100 rpm) but the foil stretching was just made with a maximum speed of 3 rpm for a better control.

III.2.2 Parameters, Pitfalls and Solutions

The roll-stretching device was adjusted with the help of two students. First to mentioned is my long-time student assistant Friederike Ganster who prepared the first foils on the first setup of the roll-stretching device. Secondly, the parameters to make reproducible stretched foils with the final design of the roll-stretching device were investigated together with my internship student Tobias Scheibel, B.Sc.

III.2.2.1 Speed of Foil through the Heating Zone

It is favorable that the foil and the particles have the longest possible time in the heating zone to heat up the embedded PS particles above their T_g and to get fully stretched particles. Therefore, the maximum possible oven time was defined by the largest targeted strain. The lowest speed of the feeding roll (M_1) was 1.3 rpm, leading to a total maximal oven time of 42.8 s for targeted 100 % strain (pulling roll M_2 speed 2.6 rpm).

Table III-1. Speed test with heat zone times for different motor speeds and different strains. The table is adapted from the protocol of the internship of Tobias Scheibel, B.Sc. Note that the slowest speed of the engine is a motor speed of 1.3 rpm (set at M_1 with 100 % strain). The times are hand-stopped with the help of markers on the foil (start by entering and stop by leaving the covered area of the heating plate/heating zone, 16 cm). The displayed time is the average of 3 measurements. Further speeds that were close to 42 s oven times can be found in the supporting information III.5.4 at Table SI III-2.

Strain	**0 %**		**15 %**		**50 %**		**100 %**	
	M_1	M_2	M_1	M_2	M_1	M_2	M_1	M_2
Motor speed [rpm]	2.05	2.05	2.0	2.3	1.6	2.4	1.3	2.6
Heat time	42.9 ± 0.1 s		42.0 ± 0.6 s		42.5 ± 0.0 s		42.8 ± 0.2 s	
Foil speed	3.73 mm/s		3.81 mm/s		3.77 mm/s		3.74 mm/s	

For the other applied strains, the motor speed of the feeding (M_1) and the pulling (M_2) rolls had to be adjusted to have a similar oven time of 42 – 43 s (Table III-1) of the foil. This was necessary to have the prepared particles comparable to each other. The foil speed was calculated to around 3.76 mm/s (13.5 m/h). As faster speeds were not used in this chapter, the influence of changing the speed on the resulting particles could be interesting for further investigations.

III.2.2.2 Bending of Foil Ribbons

The produced foil ribbons that showed extreme bending in the xy-plane (see Chapter III.1.2.4, x-direction is the length and y-direction is the width of the foil ribbon) could not be used in the roll-stretching device, as such foils would move to the sides of the rolls and get tangled during stretching. However, with minor bended foils it was possible by constantly reallocating the foil to the same spot on the upper feeding roll (2nd roll) during the stretching process. Hence, one has to be careful to maintain the tension of the foil ribbon.

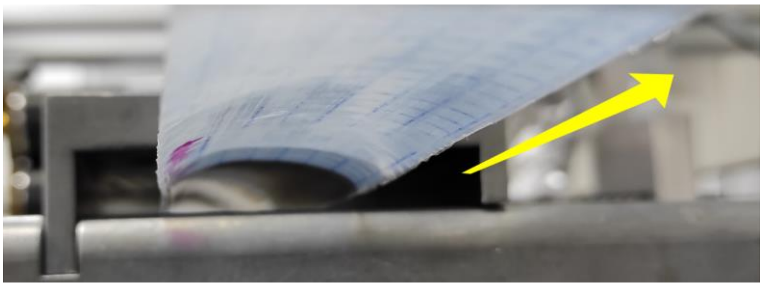


Figure III-10. The foil bends in the heating zone, but still has no physical contact with the 150 °C hot plate. Foil movement indicated by yellow arrow.

Another type of bending always occurred in the part of the heating zone. The foil started to bend in the yz-plane (width/thickness), forming an arc (Figure III-10). The yz-bending was a problem sometimes, when parts of the foil got in physical contact with the aluminium of the heating tunnel, leading to sticking and rupture. However, the printed stretching markers showed that the foil was still evenly stretched in x-direction.

III.2.2.3 Maximum Temperature

The maximum temperature was found to be 170 °C for the contact-less mode. With higher temperatures, the foil got too soft, lost its tension and lowered itself down to the heating plate where it got stuck and decomposed. However, using the 170 °C as processing temperature proofed to be difficult as well since the foil got stuck immediately when encountering the plate. Therefore, the temperature of 150 °C was chosen as the standard processing temperature since loosening the foils that touched the plate was still practicable then. This was actually the same temperature that was applied in the convection oven at the manual stretching process (Chapter II.2.1).

A continuous temperature measurement inside the heating tunnel could not be done with this device. However, with a glass thermometer reaching from one side into the heating tunnel without having direct contact, the temperature of 150 °C was found in the center of the heating tunnel when 150 °C was applied by the temperature control unit.

III.2.2.4 Counter Precession of Rolls

With the first designs of the roll-stretching device (Figure III-8), a precession of the rolls occurred when they were directly stacked on the motor. The precession resulted in foils with different stretched areas (wavy stretching grid, Figure III-11).

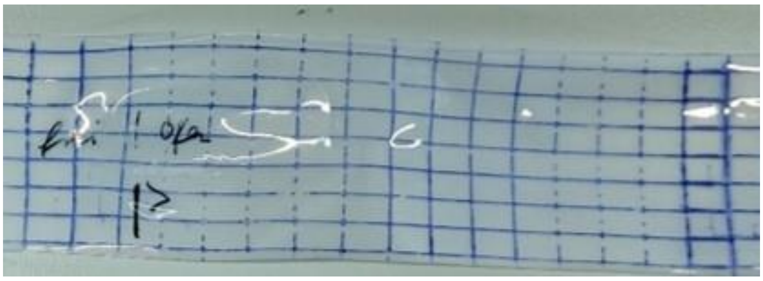


Figure III-11. Example of a foil made with the first design of the roll-stretching device, showing precession and therefore uneven stretching.

Therefore, the design was adjusted to counter this precession. For the final design (Figure III-7 and Figure III-12), the rolls were mounted in a rigid frame (top rolls still movable upwards and downwards) and the motors were connected by a timing belt. On top, the whole device was stabilized with an aluminium stand having additional feet.

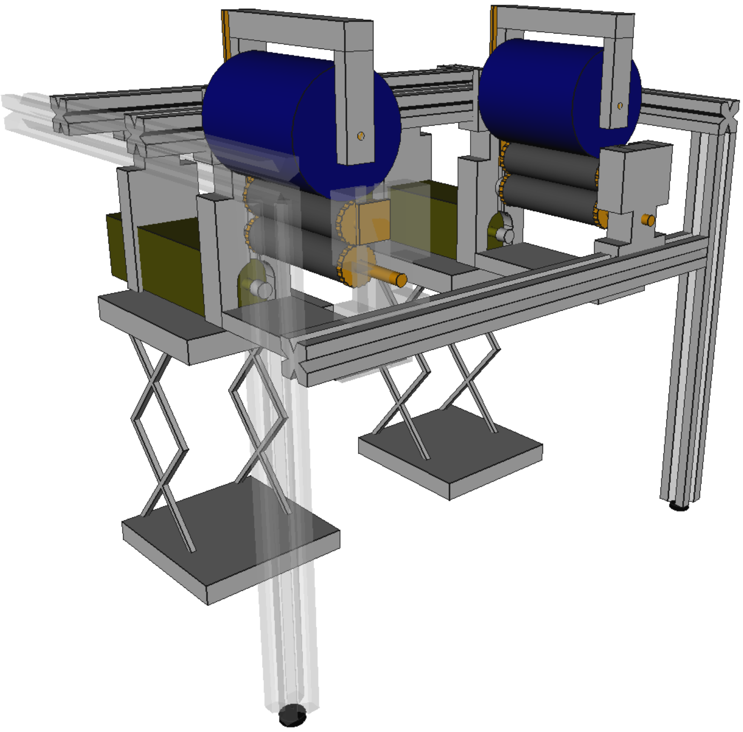


Figure III-12. Perspective view of a 3D reconstruction (by FreeCAD 0.18) of the final design. The precession of the rolls was countered by having the rolls guided in a rigid frame and the motors (green) use a timing belt (not shown) to transfer their power to the lower fixed-in-place rolls.

III.2.3 Results of Stretched Particles by Roll-Stretching

With the roll-stretching device some interesting-looking batches of particles were prepared. Unfortunately, the roll-stretching device was finished late in the lab work time of this thesis (end of 2021) and could, therefore, not be used to its full potential. The results in this chapter are extracted by myself, but are based on the foils prepared during the internship of Tobias Scheibel, B.Sc.

The investigated particles were small type 1 (254 ± 6 nm PS-AA, FN294) and big type 3 (1387 ± 19 nm PS-PSS, DB14) particles. The particles were investigated on different stretching ratios at a constant temperature of 150 °C and at two different concentrations of PS in PVA (16.7 wt% (5:1) and 10 wt% (9:1) PS in dry foil) (Figure III-13). The 10 wt% foils were additionally investigated with the oven-stretching method at 150 °C and with different temperatures on the roll-stretching device at 100 % strain. All foils used here were prepared with the automated foil doctor blade (Chapter III.1) and a frame slit height of 400 µm. Due to another idea, the foils discussed were made with an addition of Cesium ions (from 1 M aqueous CsCl salt solution, see Chapter IV.2.2) similar to the samples used in Chapter IV.

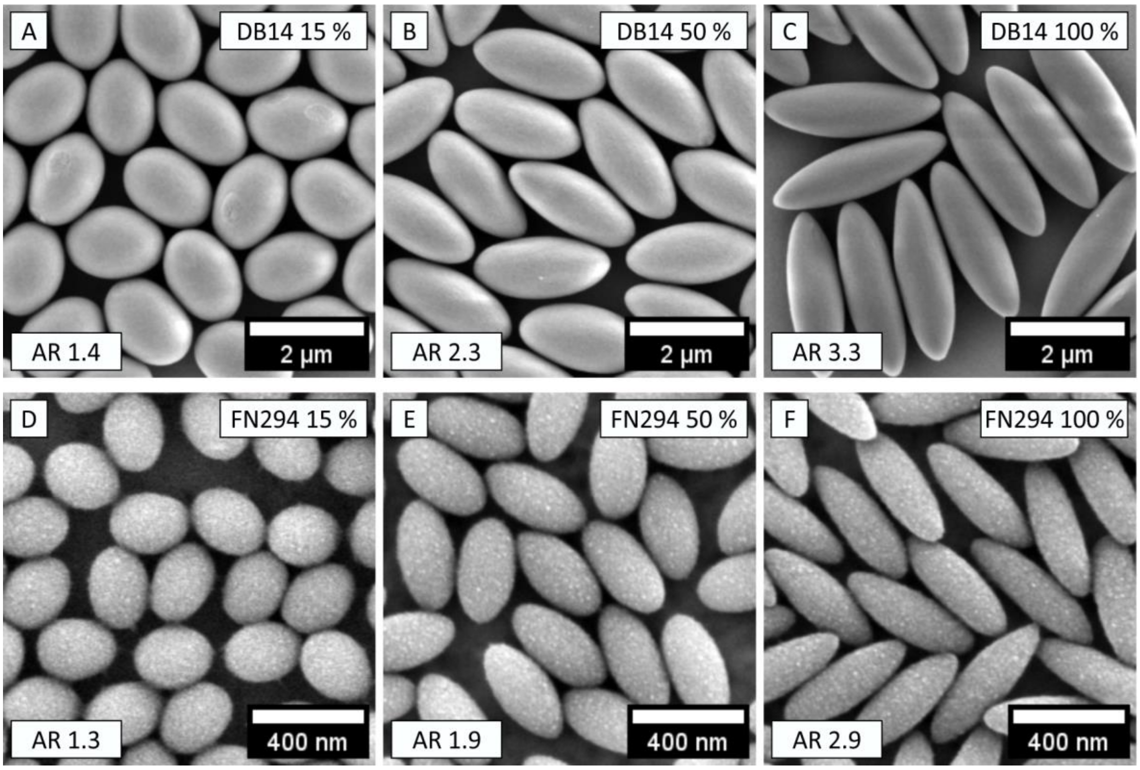


Figure III-13. Particles of different sizes stretched with the roll-stretching device using different strains; A, D: 15 %, B, E: 50 %, and C, F: 100 %. The big particles A – C (DB14) had the same shape as the small particles D – F (FN294). The small particles showed a rough surface, which was already present in the original particles. Displayed images are from SEM SE2 detector, while evaluation was mainly done using InLense for small and EsB for big particles.

III.2.3.1 ARs at Different Strains

The targeted strains were 0 %, 15 %, 50 %, and 100 %. The foils were stretched at 150 °C using the motor speeds depicted in Table III-1.

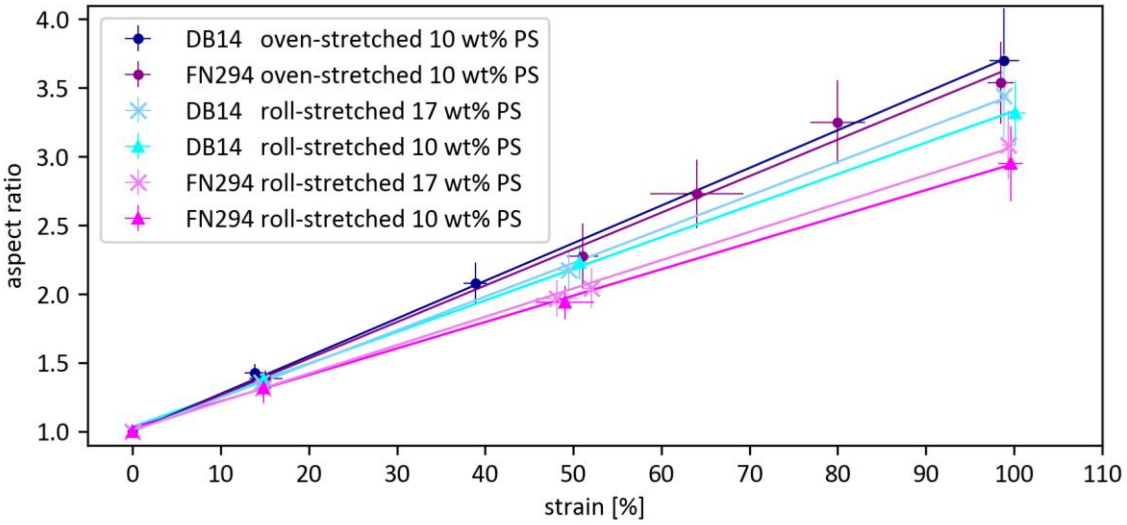


Figure III-14. ARs of different particle samples after stretching at 150 °C in the oven or with the roll-stretching device with linear plots. Data table can be found in the supporting information III.5.5.

The particles were stretched evenly and retained their monodispersity (Figure III-13). The resulting AR showed linear dependencies with strain within the investigated AR regime (Figure III-14).

III.2.3.2 ARs of Different Stretching Methods

The roll-stretched particles had a significantly lower AR compared to the oven-stretched particles at the same strain (Figure III-14 e.g. compare dark blue with light blue). I attribute this to the short time inside the heating tunnel and, therefore, incomplete stretching as explained later.

Although the ARs of the oven-stretched samples were higher than the roll-stretched ones, they were significantly lower than expected. As discussed in Chapter II.3.1.3 for oven-stretched small particles (FN269, 326 nm) at 100 % strain an AR of 4 can be expected. This AR was not met with the investigated foils here (AR 3.5 for small FN294, 254 nm, particles and AR 3.7 for big DB14, 1387 nm, particles). An explanation to this lower AR within this study could not be found. Therefore, the ARs of the roll-stretched samples should just be compared to the here shown results of the oven-stretched samples of the same foils.

III.2.3.3 AR Dependency on Particle Concentrations

The investigation of different concentrations of particles resulted in slightly lower AR with lower concentrations (Figure III-14). Although the trend is not clearly visible (close to the error range), I hypothesized that an increased next neighbor particle interaction might increase the shear field leading to higher deformation (sometimes uneven deformations).

III.2.3.4 AR Dependency on Particle Size

The trend that the larger particles will show a higher AR at the same strain as small particles (compare to Chapter II.3.1.3) could be seen in the datasets of Chapter III.2.3.1 and III.2.3.5 as well. Interestingly, the oven-stretched particles showed just a low deviation of $\Delta\mathrm{AR} = 0.16$ (10 wt% PS) between big and small particles, while the roll-stretched particles had a higher difference of $\Delta\mathrm{AR} = 0.37$ (10 wt% PS) and 0.35 (17 wt% PS) at 100 % strain. Thus, the big roll-stretched particles showed a significantly higher deformation with lower strains when compared to the small roll-stretched particles.

III.2.3.5 Roll-Stretching at Different Temperatures

Within the internship of Tobias Scheibel, B.Sc., we concentrated on investigating different parameters of the roll-stretching device and their influence on the particle inside the foil. Therefore, the processing temperature in the heating zone was altered from 110 °C up to 170 °C for foils with the two differently sized particles (note: T_g of PS: 107 ± 2 °C [86], Appendix VIII.1.7.2).

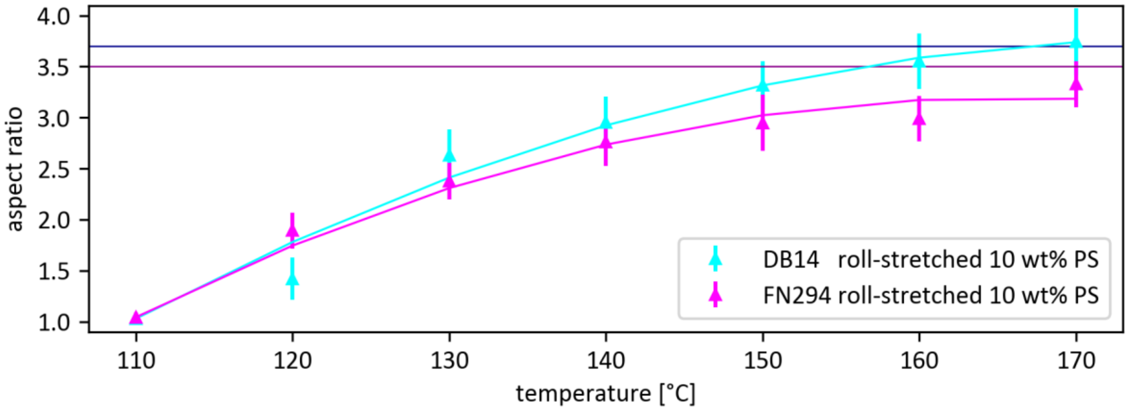


Figure III-15. AR to different temperatures applied on foils with 10 wt% PS. Horizontal lines show the value for 100 % strain of the oven-stretched particles at 150 °C (compare Figure III-14). While DB14 particles managed to come to the value between 160 – 170 °C, it was not possible for the small particles to reach the oven-stretched value. The values seem to show a cubic dependency (points fitted with 110 °C as starting point). Note that the value of DB14 at 120 °C resulted from lemon like shapes, which could not be evaluated well, as one of their tips always pointed upwards.

The resulting stretched foil ribbons were sorted by evenly stretched areas and particles were recovered. All foils for the temperature dependent investigation were stretched to 100 % strain.

As expected, the particles deformed more with higher temperatures as the polymer chains could move better (Figure III-15). The shape of the particles changed from spheres over lemons, bullet and small grain rice to the expected long grain rice shapes (Figure III-16).

This deformation behavior was the same for different particle sizes. However, the small particles exhibited no bullet shapes (Figure III-16 H – N). Although the PVA matrix to particle contact was different between the two particle types (small particles had AA surface groups while big particles had PSS, both particle types negatively charged), I hypothesize that the size of the particles had the most influence on the different shape evolutions. Especially at temperatures close to the T_g of PS ($T_{g\_midpoint} = 107 \pm 2$ °C [86], see T_g definition in Appendix VIII.1.7.2), it seems like just a fraction of the big particles was heated far enough above T_g where the polymer chains can move freely. Thus, the chains in the particles were still partially hindered and the particles could not deform completely within the given time. In other words, the mass transfer of the polymer was not fast enough to fill out the surrounding voids completely while moving through the heating zone (compare to Champion et al. [44]). Thus, the big particles developed the present lemon and bullet shapes (Figure III-16 A – D).

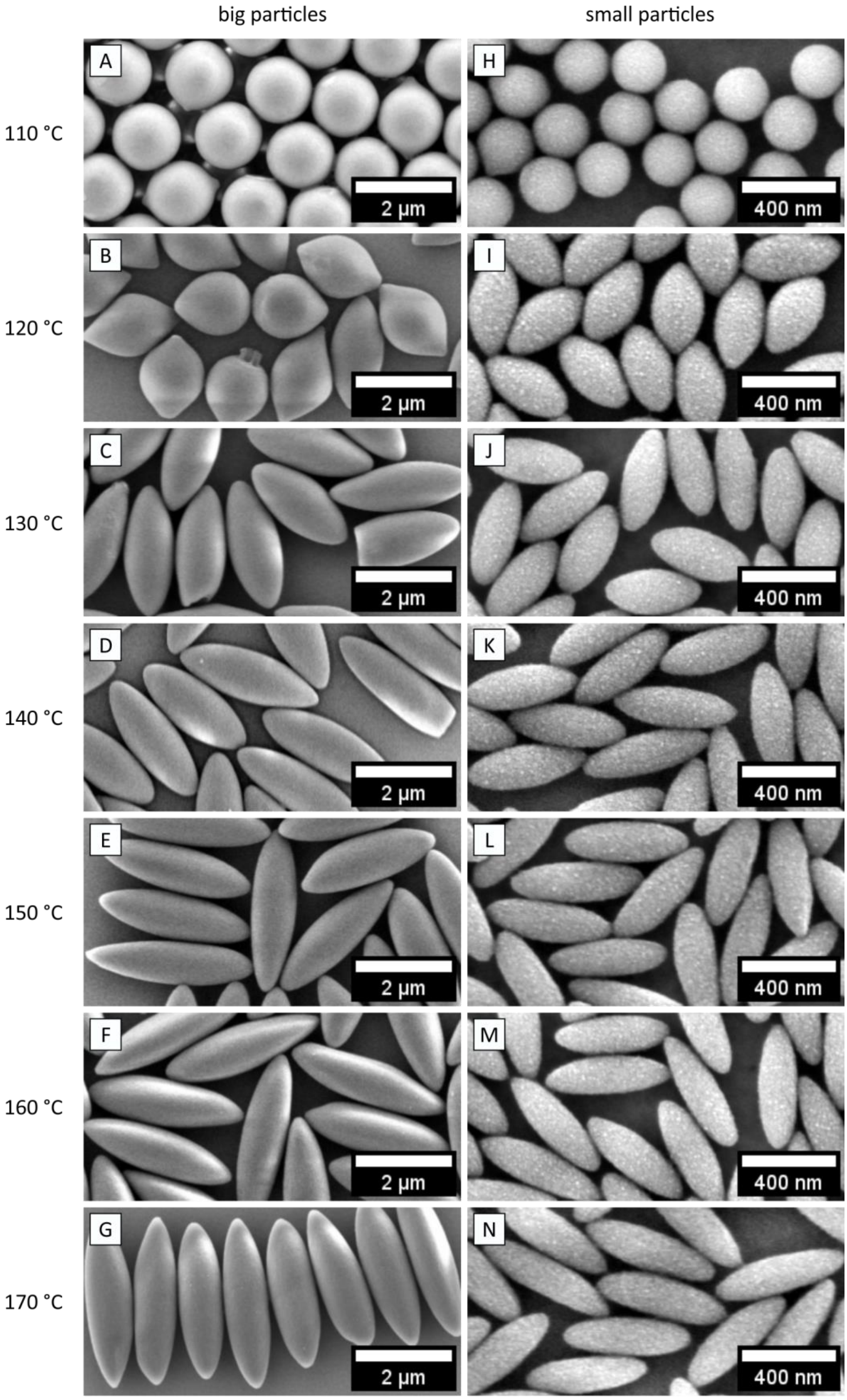


Figure III-16. SEM images of roll-stretched particles. A – G: Big particles (DB14) and H – N: small particles (FN294). The particles exhibit different shapes with different temperatures at roll-stretching. The ellipsoidal shape development in big particles needed higher temperatures. All particles from foils with 10 wt% PS.

III.3. Automated Particle Sizing and Positioning by Python

With hundreds of particles in just one microscopy image to be evaluated with respect to dimensions and orientation of the particles, the common method of sizing particles by hand via a scientific image-processing program like Fiji/ImageJ would have been very time consuming. Therefore, I developed an image evaluation program in the programming language Python3 to determine the sizes, ARs, positions and orientation of the particles in the microscopy images rapidly in an automated fashion.

The program version discussed in this paragraph is version 0.48. The final version, however, is version 0.54 where additional statistic evaluations were implemented. Those additions have no influence on the core of the program. The used python distribution was version 3.9.1 with sublime text (Sublime HQ Pty Ltd, Build 4143) as a coding environment.

The filters used for the thresholding and the evaluation are from the python package *scikit-image* (version 0.18.1, import *skimage*). “Scikit-image is a collection of algorithms for image processing” [100]. The free-of-charge package and a documentation can be found on the webpage (https://scikit-image.org/, last access on 29.10.2023).

Other packages used in the program: PIL [101] version 8.0.1, Numpy [102] version 1.19.3, matplotlib [103] version 3.4.3, scipy [104] version 1.6.0.

The general construct for evaluating images was not altered after version 0.39 and follows this linear process:

1. load the image file into python (program tab 1)
2. identify separated areas and get their parameters (tab 2)
3. filter the areas roughly and create ellipses with area dimensions (tab 3)
4. choose by clicking on the image the well detected areas/particles (tab 4)
5. export the data for further use.

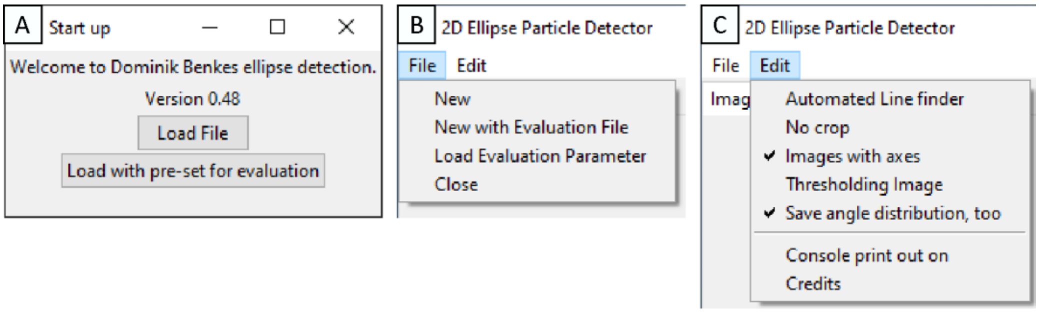
The basic concept of loading the picture and separating the particle areas (points 1 and 2) was developed and implemented by myself together with my former colleague Dr. Bernd A.F. Kopera (member of the group until 2019), while the whole graphic user interface (GUI) like implementing choices for filter and further visual evaluation was programmed by myself.

In the following, the individual parts of the image evaluation program are explained.

III.3.1 Step 1 – Starting the Program and Loading the Image File

The python program can be started by building the start file with python 3 (python files on enclosed CD ROM or via download link).

The first frame (Figure III-17), which pops up, asks to choose whether a new file should be loaded or whether a file should be loaded using former evaluation parameters. In both ways, the file dialog (from the internal python package *tkinter*) opens up and one can choose an image file (jpg, bmp, tiff).



Welcome to Dominik Benkes ellipse detection.

Version 0.48

Figure III-17. A: The starting frame of the ellipse detection program lets one choose how to load the file. B: A new file can be loaded as well by clicking on the file menu "New" or faster by clicking the name of the file that appears next to the edit menu after firstly opening a file. A set of evaluation parameters can be opened even after loading the image file with "Load Evaluation Parameter". C: The edit menu shows different options as explained in the given paragraphs.

The image file is loaded as a 2D array to python (via *io.imread*) and is transferred to a grey-value image. The standard settings at every start up crop the lower part (10 %) of the loaded image, as the SEM images of the SEM microscopy department of University Bayreuth had the imaging parameters branded on the image there. To switch off this behavior, one can go to the "Edit" menu and choose "No crop". Another possibility to crop the image is to use the "Automated Line finder" where the image is searched for horizontal black lines. The height of the image is then reduced to one pixel above the black line (in y-direction).

These original and reduced, grey-value images are displayed on the right side of the first tab in the program (Figure III-18). On the tab, the user must insert a pixel ratio of the image. If the pixel ratio is not displayed in the information on the SEM image, it is recommended to use Fiji/ImageJ to measure scale bars normally displayed on microscopy images. The pixel ratio has to be inserted as nm/pxl with English notation (use a dot, not a comma for decimal).

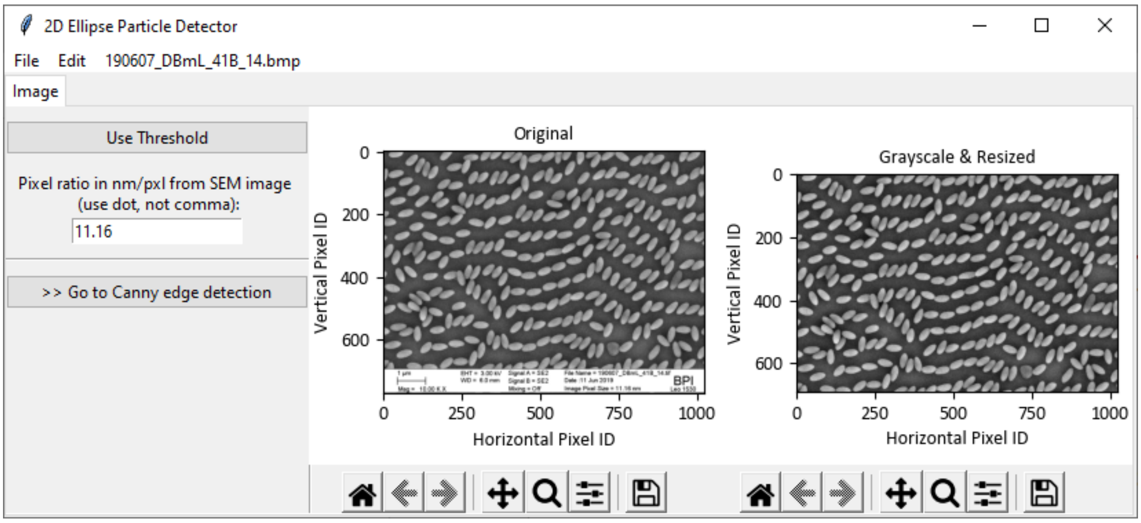


Figure III-18. Tab 1 of the program. The pixel ratio must be inserted. Additionally, one can open up the threshold window. The original image and the cropped greyscale image are displayed. The displayed images can be zoomed, saved and altered with standard image menus below the respective image, as known from using python's matplotlib. Note that in the figure here, the inset images in the tab are extra minimized since normally the tab has full screen size.

For some images (e.g. optical laser-scanning microscope images of Chapter V), it is advisable to choose the "use threshold" method on tab 1. With this, the image is thresholded to a black and white image with two different methods (*threshold_local* and *threshold_otsu* (global)). On the pop-up window (Figure III-19) one can choose which threshold should be used for the further evaluation (important to close via X if not used). Note here that sometimes thresholding gives not the exact dimensions of the particles. If a threshold is chosen, the "Thresholding Image" option in the "Edit" menu is set active. The threshold window will then be automatically generated when opening a new image (the option can be deselected again in the "Edit" menu).

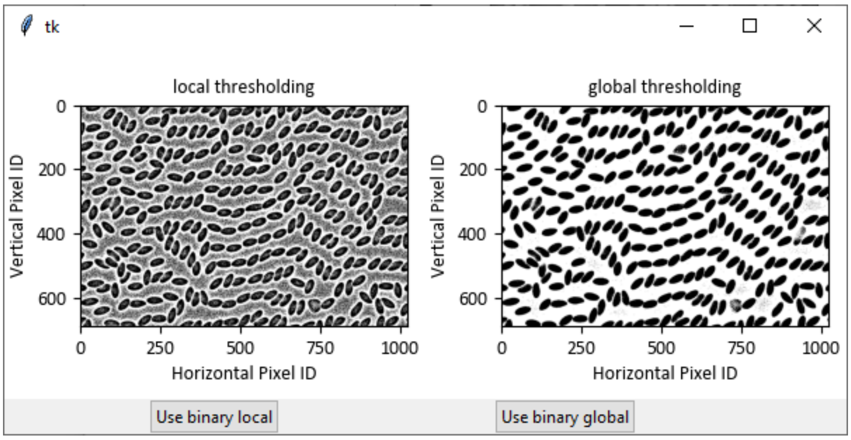


Figure III-19. Threshold window of the program (insets of images minimized). On this window, one can choose which thresholded image is used. Binary local uses the method "threshold_local" while binary global uses "threshold_otsu" from "skimage" package.

Proceed to the next step by clicking on ">> Go to Canny edge detection".

III.3.2 Step 2 – Get the Area Information

Tab 2 is used for splitting the image in areas and getting the size and other information from these areas. To do so, the canny edge detection algorithm from the *skimage* package is used to define the edges of the particles on the image. Parameters needed for the edge detection algorithm are sigma and the high and low threshold. The parameters can be defined with the range sliders on the left side of the tab. With the button ”> Edge Detection“ the canny edge detection is performed on the image. The result is displayed as 1st image and as an overlay with the original image as the 2nd image on the right side of the tab (Figure III-20). The plotting of the edges can be done simultaneously with sliding the values after clicking the checkbox “Simultaneous Plotting“. This is very helpful when having no previous evaluation done on similar microscope images to get the values into the right range. For later evaluation, the checkbox “Add Particles at Borders” can be switched on to get regions touching the image borders. As these regions are normally deformed, it is standard to be switched off.

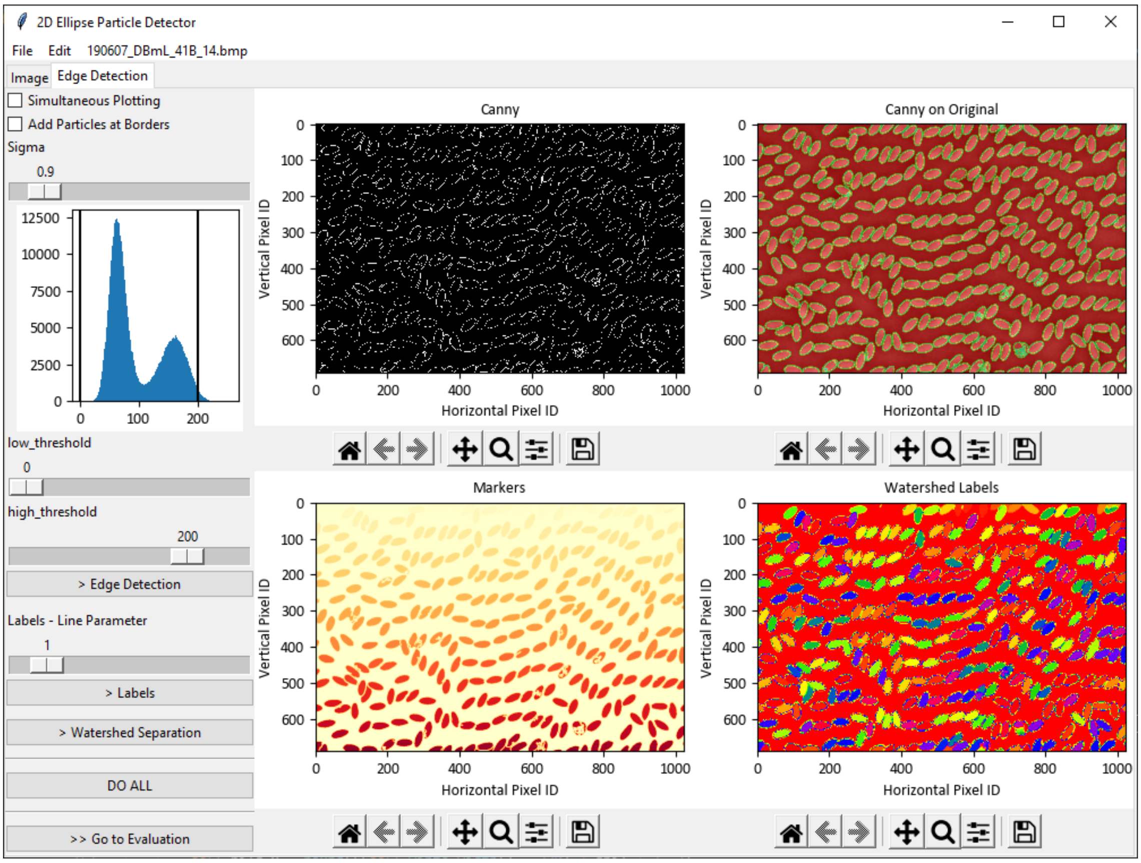


Figure III-20. Tab 2 of the program (insets of images minimized). The loaded image is investigated by canny edge detection and watershed separation. The parameters can be set on the left side of the tab with sliders. The individual steps can be done at once with “DO ALL“. The plotting can be switched on to “Simultaneous plotting” to find the best “Sigma”, threshold, and “Labels - Line Parameter” on the respective slider more easily.

After defining the edges of the ellipsoids with canny, the marker image is made by adjusting the size of the lines from the canny image (slider of “Labels - Line Parameter“) and separating enclosed areas using *rank.gradient* filter of *skimage*. The so defined areas are the starting points for the watershed algorithm (button “> Watershed Separation“) used to plot the watershed labels image. With the watershed method, it is possible to separate touching particles where the detected edges are not completely surrounding the single particles, thus leaving a gap.

The areas defined by the watershed separation are evaluated using *regionprops* of *skimage.measure.* The resulting property list is then handed to the next part of the program.

To ease batch evaluation of images with similar illumination parameters, the three steps can be done all together by clicking on “DO ALL“.

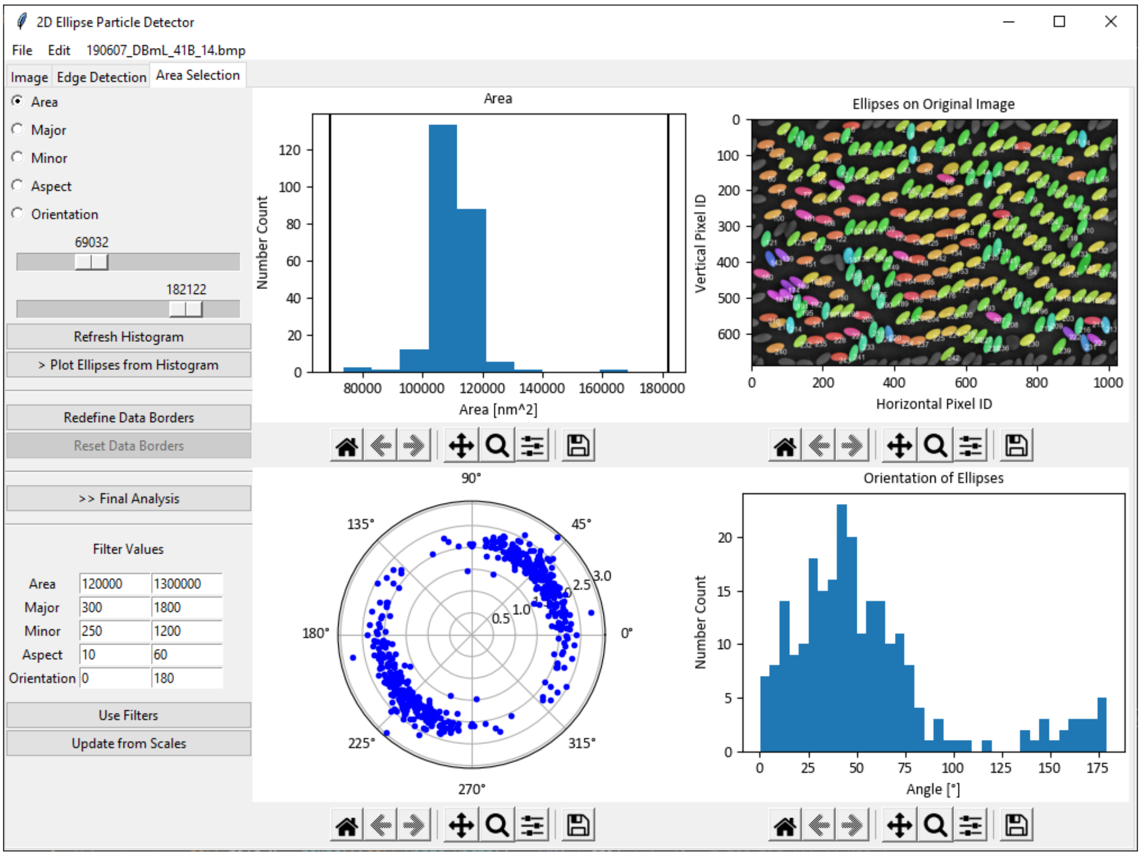
The evaluation is then continued by clicking on “>> Go to Evaluation“.

III.3.3 Step 3 – Filter the Data

The region property list is filtered on its different properties in tab 3. While there are more than 40 different properties in the list (full list and definitions see webpage of scikit-image.org), the ellipse particle detector is just using five of them to filter the data.

The regions in the list can be filtered by area, major length (*major_axis_length*), minor length (*minor_axis_length*), calculated AR (*major_axis_length/minor_axis_length*), or orientation. By clicking the respective radio button on tab 3 (Figure III-21), the histogram in the middle shows the values of the list. The filter is applied using the sliders and then clicking “Refresh Histogram“. By using the button “> Plot Ellipses from Histogram“ the region properties are used to construct an image with ellipses of the same length, width, orientation and location as the regions they originated from. The ellipses will have a coloration dependent on their orientation (horizontal red, vertical cyan, used cyclic color map: *cm.nipy_spectral*). The ellipse image is then displayed on the right as an overlay on the original image. For a first quick check, the orientation distribution of the ellipses is displayed on a polar plot and as a histogram.

The filter used on the particle list is not yet permanent and can be altered by moving the sliders to new positions, followed by clicking on “Refresh Histogram” and then clicking on “>Plot Ellipses from Histogram”. To redo the filter with other properties (e.g. first area, then aspect filter) the dataset needs to be made semi-permanent by redefining the particle list clicking “Redefine Data Borders“. The histogram will be rescaled to the chosen part of the particle list. By then choosing another parameter with the radio button, the particle list can be further filtered. To come back to the full particle list the button “Reset Data Borders“ is defined.



● Area
○ Major
○ Minor
○ Aspect
○ Orientation

69032

182122

Filter Values

Area	120000	1300000
Major	300	1800
Minor	250	1200
Aspect	10	60
Orientation	0	180

Figure III-21. Tab 3 of the program (insets of images minimized). The region properties are filtered and the ellipses image is made.

When using images of the same data source one might know already which filters in size need to be applied. Therefore, a data field for the filter values was implemented where one can insert the minimum and maximum values applied as filters on the dataset (click “Use Filters“). These data fields can also be used to view the values of the applied filters (click “Update from Scales“).

The program continues by clicking „>> Final Analysis“.

III.3.4 Step 4 – Confine the Data Precisely by Image Clicking

The filtered areas/particles are then displayed on a large overlay image (on original), giving the user the possibility to deselect individual ellipses by mouse click. This is necessary as some ellipses do not fit the particle surroundings or the particles, where the ellipse is based on, were tilted (Figure III-22 ellipse 191, lower left on image). For a better view, the numbers on the image can be deleted when deselecting the checkbox “Switch ON Numbers“.

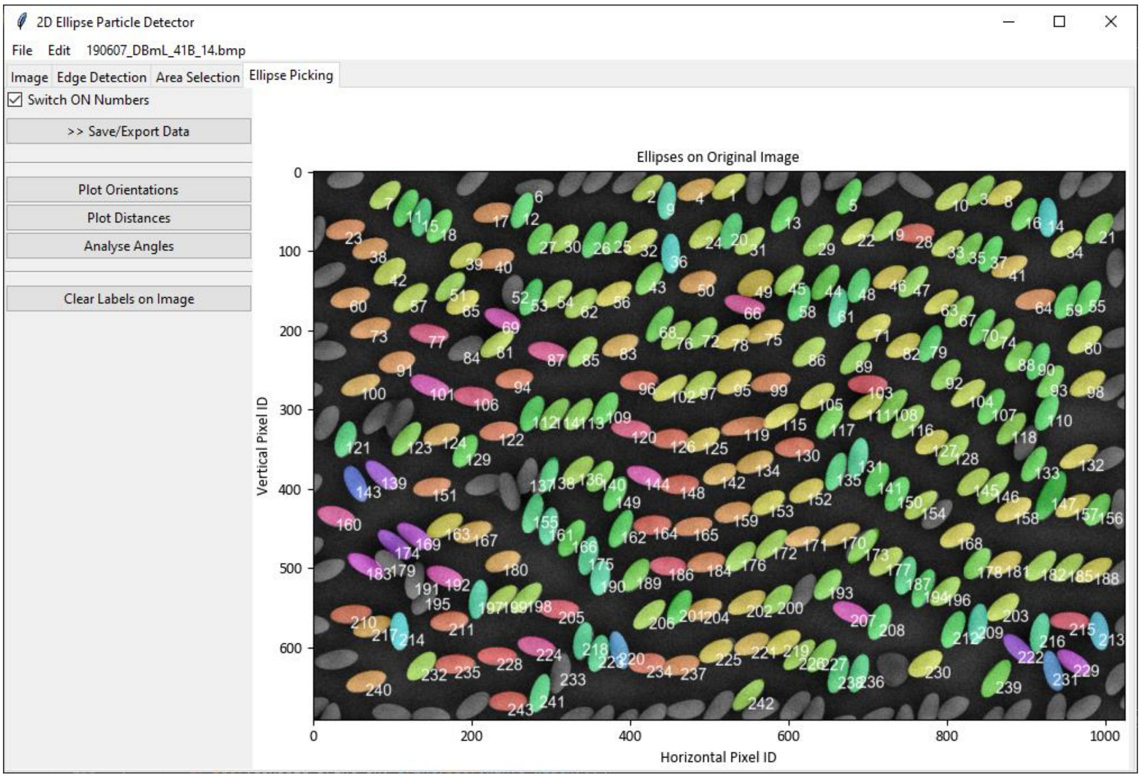


Figure III-22. Tab 4 of the program gives the user the opportunity to deselect individual particles by clicking on the overlay image (inset of image minimized).

As one can select and deselect different particles, the angular distribution of the shown ellipses can be re-plotted with the button “Plot Orientations“ (Figure III-23 A).

The button “Plot Distances“ creates a histogram and polar plot of the distances between the displayed particles (Figure III-23 B). These plots change drastically when having a crystalline pattern.

The button “Analyse Angles“ creates a plot of the mean-value angular correlation function (Figure III-23 C) describing the difference in the ellipse orientation θ [105]. This evaluation will be implemented and further discussed in Chapter V.1.7.

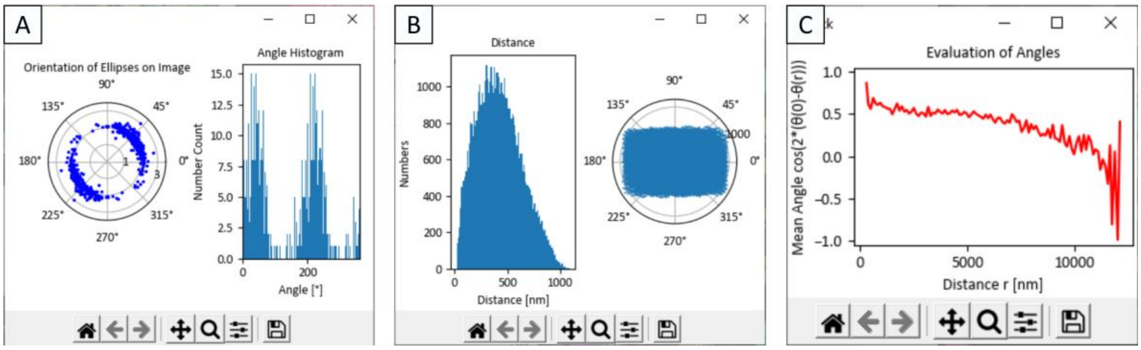


Figure III-23. Analysis windows of tab 4. A: Orientations of the ellipses on image in tab 4 as polar plot and histogram. B: Distance plot, mostly used for just a few particles or highly ordered particle layers (the polar plot is normally larger with more space between data points as displayed here). C: Mean-value angular correlation function of the displayed particles (discussed in Chapter V.1.7).

The last button on tab 4, "Clear Label on Image", can be used to inverse the deselection process. All particles on the image will be deselected, so one can select the individual particle regions again by clicking on the image.

III.3.5 Export the Data and Parameters for the Whole Sample

When clicking ">> Save/Export Data", the data of the displayed ellipses, the program parameters, the ellipse image (colored ellipses on black ground), the overlay image (as displayed in Figure III-22 with axes and, additionally, in cropped original dimensions), the graph of "Analyse Angles" (if clicked), and, by default, the orientation plots are saved. However, the default saving of the orientation can be switched off in "Edit" menu (Figure III-17 C).

The images and plots are exported as png files and the ellipses data and program parameters are exported as plain txt files. The program parameters txt-file contains e.g. the parameters sigma, threshold, and line width, which are parameters of tab 2 (Chapter III.3.2). The program parameters txt-file is used for loading an image with pre-set values (explained in Chapter III.3.1).

The evaluation txt-file contains a table with comma-separated values of the following particle/ellipse data: number of particle reference on picture, coordinates of the particle in the picture, major area length, minor area length, orientation angle in degree (0° is horizontal), and the AR.

The evaluation data files of different picture evaluations of the same sample were then further processed in Microsoft Excel 2016 for average, median and deviation calculations of the ellipsoids samples.

III.4. Summary and Future Perspectives of Chapter III

The chapter focuses on the scale up of the three-step process to make ellipsoidal particles and on the automatization of the particle evaluation.

First, the foil making was successfully speeded up by implementing a semi-continuous foil doctor blade (Chapter III.1) with a 10 m long steel strip. With this, it was possible to make up to 8 m long, particle-loaded, already dry, ready-to-use foil ribbons within 1 – 2 hours. The foil ribbons were used successfully in the oven-stretching device of Chapter II.2.1.2. However, a continuous doctor blade for endlessly long foil ribbons could not be implemented due to difficulties with the steel belt, as described in the chapter. For future work, it would be nevertheless beneficial to implement a belt, as no back and forth spooling of the support would be needed and even longer foil ribbons could be manufactured.

The next step for scale up was successfully done, with building up a roll-stretching device for the foil ribbons (Chapter III.2). It could be shown that the particles made with this roll-stretching process exhibited the same trends as the oven-stretched foils. Small particles exhibited lower ARs than big particles at the same strain. With low temperatures on the roll-stretching device, the big particles showed different shapes (lemon, bullet), especially when the temperature was getting closer to the glass transition temperature of PS. For small particles, the AR was low at low temperatures but they did not exhibit different shapes. For future work it would be beneficial to enlarge the heating tunnel or to slow down the speed (e.g. by adding a gear reduction to the device), thus increasing the heating time of the foils which should result in more even shapes of the particles.

The last step for scale up the three-step process would be to recover the particles in large quantities. Although this step might be very important in terms of industrialization, it could not be investigated during this research. For future work it would therefore be favorable to dissolve the foil and recover the particles by using less solvent (here 30 ml per g foil) or by reducing the number of washing steps (here 7 with dissolving step), and maybe even recover the used PVA (here 83 – 90 wt% of dry foil was PVA).

To scale up the particle evaluation, an image evaluation program was developed and described in Chapter III.3. This 2D-ellipse-particle-detector program was frequently used in all chapters in this thesis to evaluate the particles. However, the particle detection might be improved as with some low contrast microscope images, the used canny edge detection algorithm was not sufficient and gaps on the surrounding lines left the particles undetected. Here, it would be either beneficial to implement other edge detection algorithms also provided by the *skimage* package or/and to give the user the ability to draw lines in the edge detection image to close the particle border or to separate the particles.

III.5. Supporting Information to Chapter III

III.5.1 SI to Chapter III.1 – Decommissioned Doctor Blade Device

The recommissioning of an old doctor blade device from the chair of Macromolecular Chemistry 1 (MC I), University Bayreuth, was done with the help of Dr. Reiner Giesa (MC I).

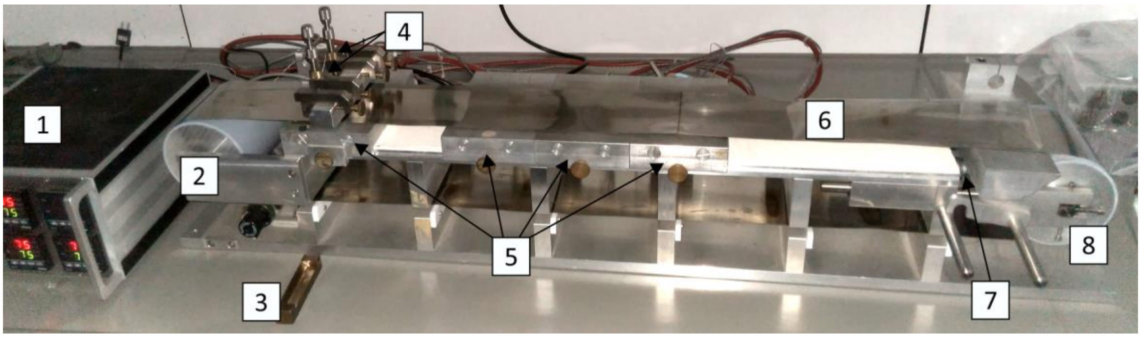


Figure SI III-1. Continuous doctor blade device before recommission. The three water-bath-connectable heating/cooling plates are not installed. 1: controller for speed and temperature of the four electronic heating plates. 2: motorized cylinder. 3: bar type doctor blade. 4: holder with screws for bar type doctor blade. 5: electronic heating plates sitting on the aluminium frame on top of an isolating sheet (white). 6: steel belt with dents. 7: gap with spring, to give tension on the steel belt. 8: un-motorized cylinder with adjusting screws.

The device already had its major features. It consisted of an aluminium frame with two cylinders on either side (one motorized, Figure SI III-1 #2). Seven heating plates were sitting on the aluminium frame (spaced by an isolating sheet). Four heating plates (Figure SI III-1 #5) had an individual temperature control (Figure SI III-1 #1) while the other three heating plates could be linked to a water cooling/heating bath (not in the picture). A steel belt (Figure SI III-1 #6) of 10 cm width was spanned around the rolls and over the heating plates. The steel belt had dents. The tension for spanning was achieved by the un-motorized roll (Figure SI III-1 #8) that could be moved outside by a spring (Figure SI III-1 #7). The un-motorized roll could be individually adjusted by spacing screws within its holder to counter side movements of the steel belt. Doctor blades (bar type, Figure SI III-1 #3) were held in place and pressed down by screws with springs (Figure SI III-1 #4).

At the MC I department, the doctor blade device was used in the process of making foils from polymer melt. At the time I took control over the doctor blade (mid 2020), the device was already unused for several years.

As described in Chapter III.1.2.1, I equipped the doctor blade device with a new supporting belt/strip. To use the device with a steel strip, I turned the device, so the motorized cylinder marked the end of the foil making process. With the help of the mechanical and electronic workshops of the University Bayreuth, the three heating bath controlled heating plates could be replaced by electronically heated plates with an individual temperature control unit. Then I constructed an automated foil pick up system (Figure III-1 black rolls), changed the doctor blade from bar type to frame type, the supporting steel strip width to 5 cm, and included magnets to flatten the steel strip.

III.5.2 SI to Chapter III.1.2.2 – Adapter for Pouring the Dispersion

The adapter was constructed in FreeCAD (version 0.18) by myself and printed on an Ultimaker3-3D-printer with ABS polymer.

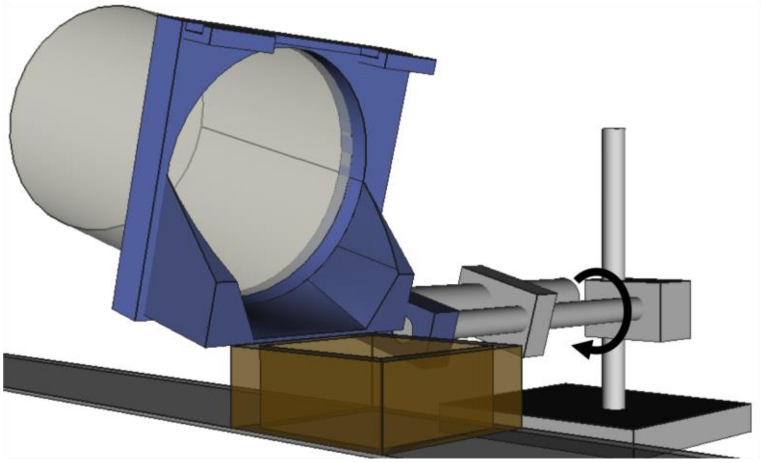


Figure SI III-2. 3D image from FreeCAD showing the final version of the adapter of the beaker (blue) for filling the dispersion into the frame doctor blade.

The adapter (Figure SI III-2 in blue) was held in position and stabilized by two metal rods. As just one of the rods is connected to the stand, the whole beaker-adapter-combination could be rotated. The outlet was positioned on the rotation axis. Additional hooks on the top were designed for a possible lid to partially cover the opening.

III.5.3 SI to Chapter III.1.2.3 – Foil Thickness at Different wt% PS

*Table SI III-1. Thickness of dried foils from the doctor blade device for different wt% particles in foil. All foils made from around 30 wt% of PVA (Mowiol® 8-88, Mw ~67000) in the total mixture. Measurements and foil preparation by Tobias Scheibel, B.Sc. during his internship. All foils contained additional CsCl, similar to the foils used in Chapter IV. *Data from heated unstretched foil.*

Particle : PVA (w/w)	**FN294 (254 ± 6 nm)**	**DB14 (1387 ± 19 nm)**
1 : 5 (16.7 wt% PS in foil)	74 ± 1 µm*	75 ± 2 µm
1 : 9 (10 wt% PS in foil)	71 ± 3 µm	70 ± 2 µm
1 : 25 (3.8 wt% PS in foil)	64 ± 2 µm	64 ± 4 µm
PVA without PS (just +CsCl)	61 ± 1 µm	

III.5.4 SI to Chapter III.2.2.1 – Speeds of Roll-Stretching Device

Table SI III-2. Further speeds of the roll-stretching device that are close to target 42 – 43 s, but were not used for the roll-stretching process. Data from internship of Tobias Scheibel, B.Sc.

Strain	**0 %**		**15 %**		**50 %**		**100 %**	
	M_1	M_2	M_1	M_2	M_1	M_2	M_1	M_2
Motor speed [rpm]	2.00	2.00	2.2	2.53	1.7	2.55	1.4	2.8
Heat time	44.0 ± 0.1 s		37.8 ± 0.3 s		41.9 ± 0.5 s		39.9 ± 0.5 s	
Foil speed	3.64 mm/s		4.23 mm/s		3.82 mm/s		4.01 mm/s	

III.5.5 SI to Chapter III.2.3.1 – AR at Different Strains

Table SI III-3. Values of the AR of different particles after applying strain by the roll-stretching device or the manual device in the oven. The foils were made by Tobias Scheibel, B.Sc., with a 400 µm frame doctor blade from 30 wt% PVA and contained CsCl (same as in Chapter IV). Particles evaluated by myself via Python program of Chapter III.3. The AR for all samples was lower as the previous found parameter for normal oven-stretched particles (normally 100 % ≈ AR of 4).

		PS in foil [wt%]	**15 %**	**50 %**	**100 %**
FN294	oven	10	1.4 ± 0.1	2.3 ± 0.2	3.5 ± 0.3
	roll	10	1.3 ± 0.1	1.9 ± 0.1	2.9 ± 0.3
	roll	16.7	1.3 ± 0.1	2.0 ± 0.1	3.1 ± 0.3
DB14	oven	10	1.4 ± 0.1	N/A	3.7 ± 0.4
	roll	10	1.4 ± 0.1	2.2 ± 0.1	3.3 ± 0.2
	roll	16.7	1.4 ± 0.1	2.2 ± 0.1	3.4 ± 0.3

Chapter IV. 3D Reconstruction of Ellipsoidal Particles in Foil

The particle dimensions can be controlled in terms of AR as shown in Chapter II. The process is consistent enough for scale up as seen in Chapter III. In this Chapter IV, the focus is drawn on the 3D shape of the resulting particles, which is important for the assembly processes of the following Chapter V and Chapter VI.

One major scope for 3D investigations is the question of what the true 3D shape of the stretched particles is. Are the particles depicting a mathematical ellipsoid with one long axis and two identical short axes, or are they flattened in one of the short axis directions?

The quest of depicting the true 3D shape was first tried to be solved by TEM tomography for individual particles. In TEM tomography, the recovered stretched particle sample (small particles in 300 nm region of Chapter II.1.3.1, FN269) was spread on a copper grid and mounted into a tiltable sample holder. Then, individual particles were imaged by TEM from various angles (-65° – 65°) and the angle-dependent images were reconstructed by the given software (IMOD) to a 3D representation of the particles. The major drawback of TEM tomography, however, is that not the whole 3D reconstructed volume is based on the taken image data since some angles cannot be measured because of tilting restrictions and shadowing of the sample (from sample holder and copper grid). Thus, the missing volume regions of the particle need to be interpolated, leading to artefacts like a pointy upper part and a deformed lower part, where the surface of the grid was (see supporting information IV.10.1, Figure SI IV-1). Moreover, only few individual particles were reconstructed and statistic methods could not be applied.

In literature, stretched particles still being inside the stretching matrix or already recovered from the matrix were investigated by fluorescence labelling and optical confocal laser microscope measurements [56, 57]. However, such approaches investigate the ensemble of the particles and not individual particles. Thus, a quote for the true 3D shape cannot be made.

Structure of the chapter

To resolve the true shape of the particles, the method of Serial Block Face Imaging Scanning Electron Microscopy (SBFI SEM, Chapter IV.1) is applied in this chapter on the polymer-particles of type 3 (DB14, 1387 ± 19 nm, CV 1.4 %, II.1.3.3) while they are still inside the stretched PVA foil. However, such a polymer in polymer observation is tricky. Therefore, a general procedure to obtain well-defined reconstructed 3D images of the particles is developed, starting with the sample preparation and imaging in a microtome SEM (Chapter IV.2), followed by a reconstruction in AMIRA (3D reconstruction software from Thermo Fisher Scientific Inc., Chapter IV.3). After the reconstruction, a Python-based distortion correction is implemented (Chapter IV.4). The correction and results on incorporated fiducials (Chapter IV.5) and on the particles (Chapter IV.6 and Chapter IV.7) are discussed. The results of the particle dimensions are compared to the particles recovered from the foil. Additionally, the 3D analysis of the particle ensembles in the foils do reveal answers to questions like: How are the particles oriented within the foil? Are they all oriented with their major axis in one direction, as expected from one-dimensional stretching? If there is a flattening in the 2nd or 3rd axis, is this flattening oriented within the foil? Are there areas of different orientations inside the foil?

The image stacks created by the SBFI SEM method also provide answers to the questions: Why do the shapes sometimes differ drastically from mathematical ellipsoids? How do the particles change their shape when hindered by each other? (Chapter IV.8)

The chapter closes with a summary and further perspectives (Chapter IV.9).

The SBFI measurements and first evaluations were done with the help of Dr. Hendrik Bargel, chair of Biomaterials, University of Bayreuth.

IV.1. The SBFI SEM Technique

For the reconstruction of the stretched particles inside the PVA foil, the so-called Serial Block Face Imaging method (SBFI) was used. With SBFI, a (resin-embedded) sample is cut by a diamond blade of an in-situ microtome inside the vacuum chamber of an SEM. After each cut, the exposed surface is imaged perpendicularly by the electron beam. The images of the different z-positions (image stacks) are then used to reconstruct sample features inside the 3D volume.

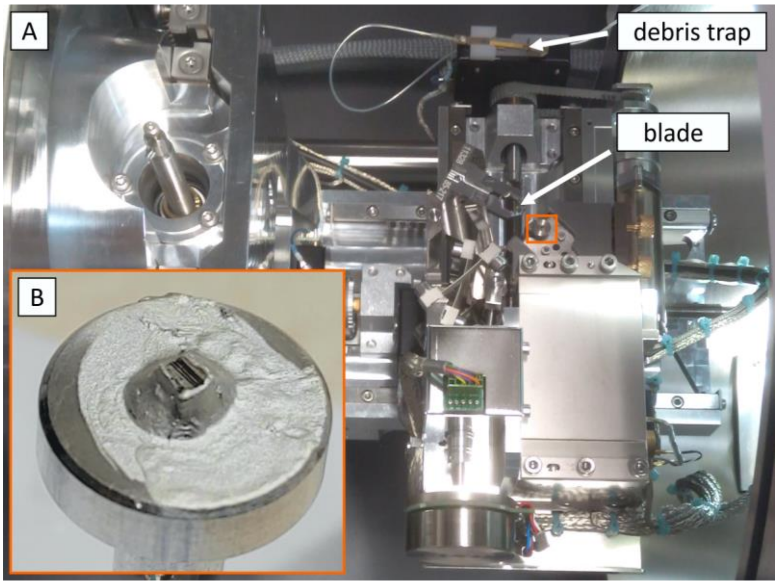


Figure IV-1. A: Photograph of the setup for serial block face imaging. The sample (orange frame + inset) was fixed on a xyz movable stage. The debris trap was mounted on top of the blade before imaging. The trap collects the cut debris by electro static forces. The pulley, where the special stage of the microtome was mounted on, was moved into the SEM chamber on the right. B: Close up of the sample with 4 foils (black stripes) in a conductive resin block (grey) mounted on a sample stub (8 mm) and contacted by conductive silver liquid prior final polishing. The sample surface was approximately 1 mm × 1 mm.

The setup used for cutting and imaging was an Apreo volumescope SEM from FEI (Thermo Fisher Scientific Inc.) with an in-situ microtome. The setup inside the electron microscope is displayed in Figure IV-1. With this setup, the mechanically achievable slice thickness can be down to 25 nm (40 nm guaranteed and used) with a cutting window of 2 mm [106] and a total z-travel range of 1.2 mm [107]. With an optical deconvolution technique (Multi-Energy Deconvolution (MED) mode in high vacuum modus) the microtome and SEM setup used here should be even capable of producing 10 nm slices. As in MED mode the same sample position is imaged with different acceleration voltages [108] and the samples used here were not beam resistant (low-vacuum mode used), the MED mode could not be used within this investigation.

IV.2. Sample Preparation and Imaging

The SBFI technique requires a stiff material to be cleanly cut by the diamond blade and the sample has to be conductive. Both was not the case for the particle-loaded foils (from DB14, II.1.3.3). Thus, the foil samples needed to be altered to render the samples suitable for SBFI.

IV.2.1 Embedding Foils

A well-known sample preparation method from biology is embedding soft specimen into an acrylic resin. This is mostly used for watery biological samples like cells and tissues including multiple dehydration and resin infiltration steps (see *ROMEIS – Mikroskopische Technik, Chapter 7.1.3* [109]). As the foil sample itself was supposed to be dry, direct embedding of the foil into a resin without dehydration was possible. A conductive epoxy resin with colloidal silver particles (EM-Tec AG32, supplier: microtonano.com) was used to avoid charging of the sample surface. To reduce the sample preparation effort, four different stretched foils were embedded simultaneously into one conductive resin block, separated by thin layers of conductive resin (Figure IV-1 B and Figure IV-3). Care was taken to embed the foils parallel to each other and record their direction of stretching.

IV.2.2 Staining

To enhance the contrast of the particles inside the foil, heavy atoms can be used to stain the matrix or the particles themselves. A pre-test with osmium staining, which is used in biological studies for TEM with sliced samples (see *ROMEIS – Mikroskopische Technik, Chapter 7.2.4.1* [109]), was not successful with these bulk samples as the osmium bled out. Hence, no contrast difference between the particles and the matrix could be observed. However, I found an easy solution by using cesium chloride, a nontoxic metal salt, which acted perfectly as contrast enhancer in this given foil system. By simply mixing CsCl (8 wt% CsCl related to the PVA dry mass by adding 1 M $CsCl_{(aq)}$) into the PVA-PS-dispersion prior drying into a foil, an even distribution of the Cs-ions in the bulk was achieved, leading to an even contrast in the SEM images (Figure IV-2). After mixing, the dispersion was viscose (30 wt% PVA in final mixture) and immediately dried, countering a possible flocculation of the particles with the cesium ions (similar to the flocculation with iron ions in Chapter III.1.3). Although, a possible flocculation was not investigated in detail, it could not be seen in the following evaluation, as mostly separated particles were present in the samples.

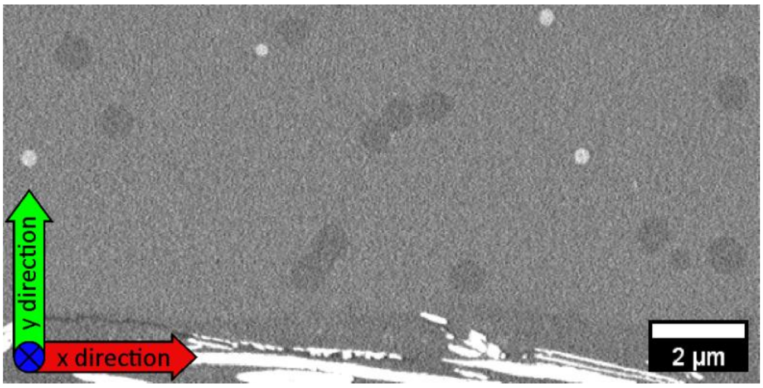


Figure IV-2. SEM image of a foil in low vacuum (xy-plane). The fiducials appeared in white, PVA matrix with CsCl in light grey, and particles of DB14 in dark grey. The silver flakes of the conductive resin are visible as bright white areas on the lower part of the image. For the following images and evaluations in this chapter the x-axis (red) is the blade movement direction, the y-axis (green) is the thickness of the foil and the z-axis (blue) shows the direction of slicing (pointing away from the viewer).

IV.2.3 Fiducials

While the pure imaging of the PS particles inside the PVA matrix was possible with the contrast enhancement, the reconstruction was problematic. An incorporation of fiducial monodisperse particles of silicon dioxide (0.04 wt% SiO_2 related to the PVA dry mass by adding a 5 wt% dispersion of SiO_2 particles; SiO_2-Forschungspartikel, 0.353 µm, micro particles GmbH; SEM evaluated: 344 ± 13 nm, CV 3.8 %) appeared to be beneficial for image alignment and data reconstruction. Without the use of the fiducials, a misalignment of the image stack was often the case; especially when the particles inside the foils were not perpendicularly oriented to the slicing direction (z-direction; results see Chapter IV.7).

IV.2.4 Mounting Sample on Stub

The resin-embedded sample (with multiple foils) was roughly cut by hand with a scalpel or razor blade to a cuboid of approximately 1 mm × 1 mm × 2 mm side length. The cuboid was then glued with two-component epoxy resin (EPO-TEK 375, mixing ratio of part A : B was 9 : 1) onto a special SBFI SEM sample holder (FEI microtome 8 mm SEM stubs with a heightening in the center, supplier: agar scientific) with the small 1 mm × 1 mm sample face upwards (Figure IV-1 B). The resin hardened in the oven at 65 °C over night. The “standing“ cuboid was then cut to a capped pyramid with a surface area of smaller than 1 mm × 1 mm by a Leica EM TRIM2 rotating diamond blade milling system (Figure IV-3). The sides of the pyramid had angles of 30° – 45° (higher degree favorable, but due to mounting on a stub prior cutting not possible). The pyramid was then sputter-coated with 2 nm platinum

(Cressington 208HR) and contacted to the stub by using colloidal silver liquid (TED PELLA Inc.). The surface of the capped pyramid was once more polished with TRIM2 right before the sample was sectioned in the SEM to produce a clean and shiny surface while the sides were still covered by platinum and silver.

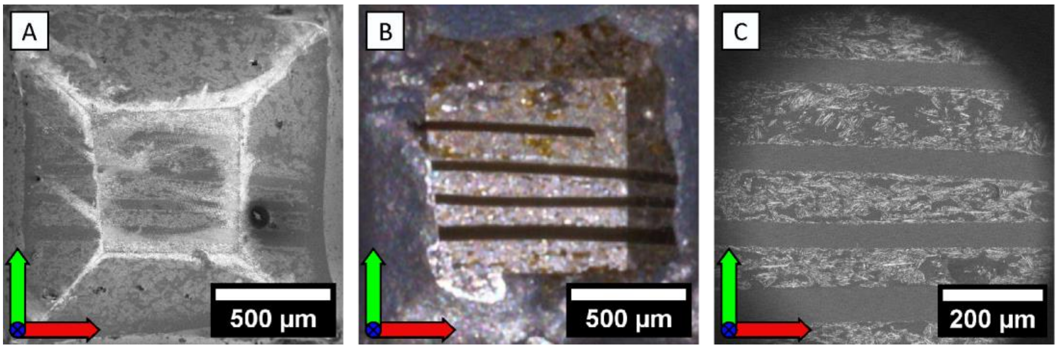


Figure IV-3. A: The sample in SEM under high vacuum condition before silver liquid but with platinum sputtering. Visible charging of the sample surface. B: Camera image of sample before SBFI SEM, after final preparation with silver liquid and polishing. C: SEM overview with low vacuum conditions prior cutting the sample. The low vacuum detector had a maximum circular image area of 600 µm in diameter.

IV.2.5 Installing the Sample in the SEM and Performing SBFI

The following steps were adapted from the process guidelines of the device manufacturer (notes from training by Thermo Fisher Scientific staff). The sample stub was mounted into the SBFI device on the pulley of the SEM (Figure IV-1 A) in a way that the direction of cutting was along the foil. The width of the stretched foil, when looking on the top (see Figure II-5, y-axis), was the x-axis/direction in the reconstruction (Figure IV-3, red axis) and the perpendicular y-direction (green axis) was the thickness of the stretched foil. Then, the diamond knife path had to be adjusted to cut the complete surface area of the capped pyramid. The blade was roughly approached by hand utilizing the reflection of light on the polished sample surface (gap closed fast until light reflection was just a fine line). The thereafter-used automated final approach was stopped after the first sample cuts. A sample polishing of around 30 slices of 100 nm (total 3 µm, cutting velocity 0.8 mm/s, Figure IV-3 B) followed. After cleaning the diamond blade by vacuuming, the debris trap was mounted and the whole device was moved into the vacuum chamber.

The chamber was set to low vacuum (0.3 – 0.4 mbar, with more charge-effects of the sample a lower vacuum was used). The blade was automatically approached to the sample (100 nm slices, 0.8 mm/s) with images taken after every 5th slice. The sufficient approach was checked by mixing SEM signals. Thereby, the second image was subtracted from the first image to identify changes from cut to cut. As soon as the mixed signal showed a difference, five additional cuts were performed to finalize the fine approach.

Then the sample was left in the vacuum for at least 2 – 3 h to equilibrate to the conditions.

After equilibration, the diamond blade had to be re-approached, due to shrinkage of the sample in the vacuum environment. The shrinkage was > 5 µm and dependent on the dryness; e.g. after 6 months drying within a desiccator and orange gel (silica gel with humidity indicator from Merck) the shrinkage was just 2 – 3 µm (Chapter IV.7). The scanning areas (two per foil), the cutting velocity (0.2 mm/s), resolution (10 nm × 10 nm), slice thickness (40 nm) and total slicing depth (up to 500 slices = 20 µm) were defined in the computer program MAPS (Thermo Fisher Scientific version 3.8).

The cutting process was performed automatically by MAPS over 2 days. The data was stored as tiff image files (resolution 4096 pxl × 3536 pxl, 10 nm/pxl) without any automated enhancement as this caused problems in the reconstruction multiple times. The tiff file stack was then uploaded to the software AMIRA EM (Thermo Fisher Scientific, version 2019.3) for 3D reconstruction.

IV.3. General 3D Reconstruction

The 3D reconstruction was done in the software AMIRA EM (version 2019.3) from Thermo Fisher Scientific. AMIRA is a powerful 3D image analyzing tool for high quality 3D data reconstruction that can be used not only on SBFI image stacks but also on multiple 3D datasets. As usual, with power comes also complexity, and thus it took quite a while to find suitable parameters for the displayed reconstruction routine here. The parameters used depended on the samples and on the taken image stacks; thereby especially the charging of regions or scan errors of individual images were an issue.

Within this sub-chapter, the main parts of the reconstruction used are explained to give a guideline to following researchers for similar reconstructions (commands and methods of AMIRA displayed in italic). Further help for reconstructions with AMIRA can be found in the handbook (*User's Guide Amira Software 2019* [110] including step-by-step protocols) and the internal help in the AMIRA software.

My final workflow contained seven steps to a first general reconstruction (this chapter), followed by a necessary distortion correction (see Chapter IV.4).

IV.3.1 Post-Processing on Image Stack

AMIRA deploys an *EM Post Processing* method for batch processing several steps including image stack contrast and brightness enhancement (*Stack Normalization, Gaussian Filter on TIFF Stack* and *Stack CLAHE*) and image alignment (*Stack Alignment* and *Stack Elastic Alignment)*. Thereby, the images are processed slice-by-slice and stored after every step as image files in a separate folder. The *EM Post Processing* method is normally first choice to start with the data reconstruction. However, the parameters to adjust are limited with this batch method. Therefore, just the contrast and brightness adjustments with the *EM Post Processing* steps *Stack Normalization* (parameters: Normalization Mode: Auto, Sampling Step: 2, Quantile min: 9, Quantile Max: 95) and *Gaussian Filter on TIFF Stack* (parameter Standard Deviation: 1.5) were used within the final reconstruction workflow.

IV.3.2 Alignment of the Image Stack

The alignment of the images can be done within the post-processing tool (*EM Post Processing*) or by using the methods *Auto Align Slices* or *Align Slices* (manual hand align) after uploading the images as a stack to AMIRA using the method *Tiff Stack Access*. However, only the methods not included in *EM Post Processing* gave the possibility to use explicitly the fiducials for image alignment by defining the brightness range of the to-be-aligned features.

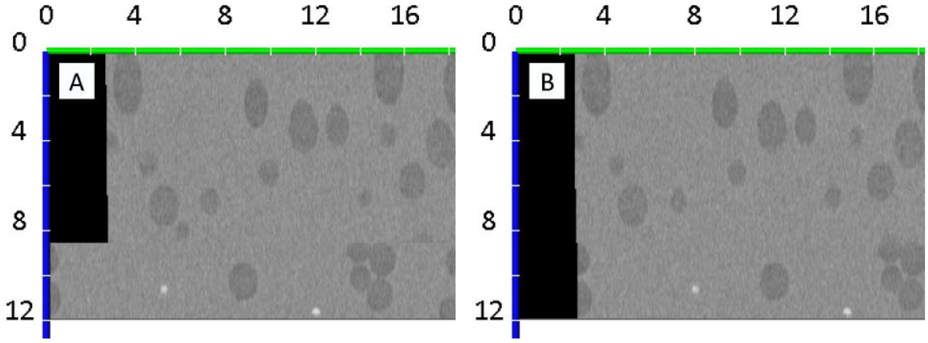


Figure IV-4. Alignment of the slices with A: automatic alignment and B: additionally by hand to correct errors in automatic alignments (like here the shift at around 8.5 µm). The fiducials are white, the PVA matrix is light grey, and the particles (from DB14) are dark grey. Scales in µm. Z-axis (blue) from top down, to show the slicing direction.

In the final workflow, the image stacks were aligned with *Auto Align Slices* (Figure IV-4 A). The brightness range of the fiducials was easily identified by the method *Interactive Thresholding* (not finally applied, just to identify the brightness range via the displayed histogram). However, sometimes, manual hand alignment needed to be done after the auto alignment procedure (Figure IV-4 B) or was performed alone. Hand alignment was performed by overlaying the image of one slice with the next slice to see the alignment by the pixel deviation of the fiducials (Figure IV-5).

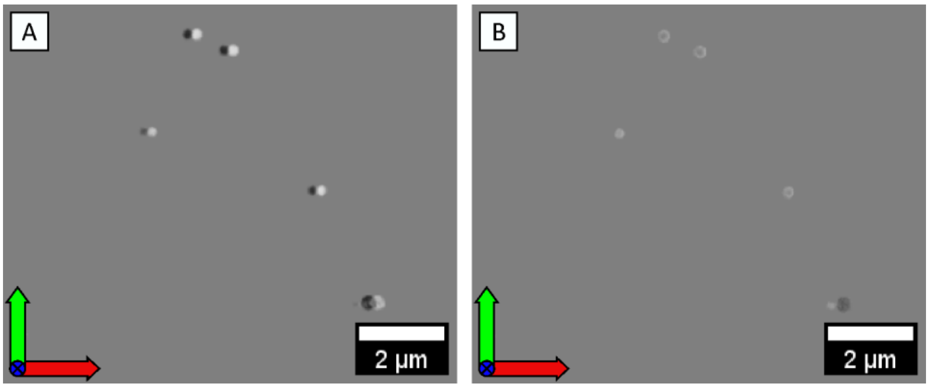


Figure IV-5. Overlaid images during alignment by hand with slice 1 in white, slice 2 in black, overlay in grey. A: wrong alignment, B: after alignment by hand to center of fiducials.

IV.3.3 Noise Removal

The images were taken with a resolution of 10 nm × 10 nm, leading to a lot of *SCHOTTKY* noise. The denoising was performed by applying two filters, one after another (*Median filter* and *Non-Local Means filter*). With denoising, the features blurred which was, however, no problem for the reconstruction (Figure IV-6).

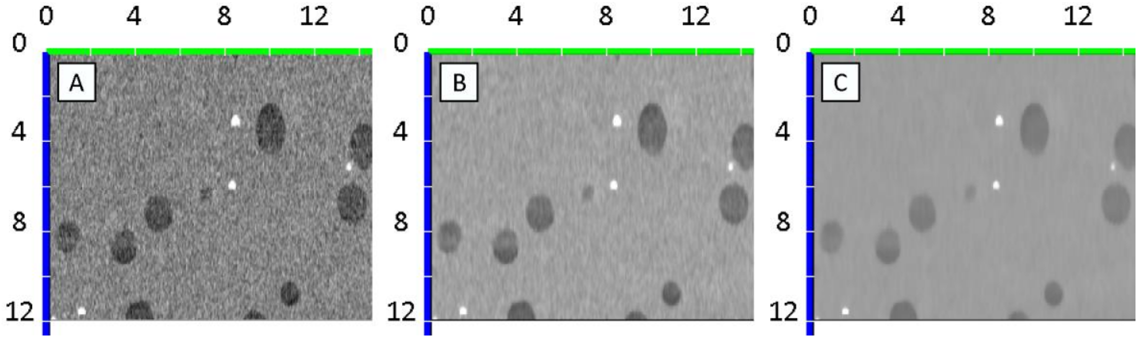


Figure IV-6. Denoising of the image stack. A: before, B: after median filter, C: after non-local means filter. Scales in µm. The SCHOTTKY noise fades from A to C. The images displayed are brightness and contrast enhanced via CorelDraw (version 17.4.0.887). However, if B and C still look the same in the printout, please review the figure in the digital version of the thesis.

IV.3.4 Threshold from the Background

The denoised volume was then thresholded to separate the particle volumes from the background. Thresholding is the most important step of the whole reconstruction as it defines whether a voxel is counted as belonging to a feature or not. Therefore, special care has to be taken at this step. AMIRA offers a variety of different kinds of thresholds. Within the final workflow, I used the method *Interactive Thresholding*, as this method gives access to a brightness histogram where one can choose the different parts of the image due to their brightness. Thus, a separation of the features was possible as the fiducials were white, the background (PVA matrix with CsCl) was light grey and the particles were dark grey.

However, sometimes the particle contrast to the background in some areas of the volume or image stack was so faint, that with the global definition of *Interactive Thresholding*, the particle region could not be separated. The particles exhibited the same grey value as the background in other regions of the sample volume (Figure IV-7).

A method to solve this problem was the use of the *island removal* tool of the *Segmentation* tab menu in order to define sites of small areas, which should blend with the environment (background or particle). However, this adjustment did lead to slightly capped ellipsoids, as the tips of the ellipsoids were detected as islands at the respective slice as well. For individual slices to be lighter than others, the handbook advises to use an additional *Automated Histogram Equalization* (CLAHE) step on the image stack. This was as well tested (parameter contrast limit 2), but discarded, as it often corrupted the segmentation even more.

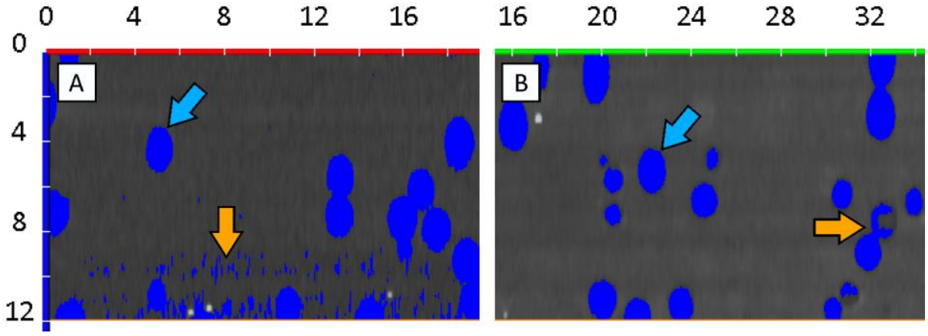


Figure IV-7. Thresholding of one sample by Interactive Thresholding (same sample and threshold in A and B, but other plane directions, A: xz, B: yz). The majority of the particles were cleanly thresholded (e.g. particles at blue arrows). However, in some areas, the global threshold gave problems (orange arrows), as the surrounding got as dark as the particles (small freckles in A) or the particles got as light as the surrounding matrix (incomplete filled particles in B). Scales in µm.

IV.3.5 Separate to Individual Particles

After thresholding, the image type was binary with the thresholded feature as one constant volume (1) inside a background (0). Consequently, the individual objects/particles needed to be separated using *Separate Objects* with *Chamfer – conservative* method as the only mathematical method available for separation with the given version of AMIRA in 3D interpretation mode. Although this was actually the second critical part, sometimes the separation was not done correctly. Some particles/fiducials were separated right through the middle of the volume (e.g. halved ellipsoids), while others were not separated at all (connected particle volumes e.g. dumbbells). Due to the limited adjustment possibilities of the *Chamfer* method, a solution to inaccurate separations could not be found. The standard parameters used were: *Neighborhood*: 26, *Marker Extent*: 4, *Output Type*: split, *Algorithm Mode*: repeatable.

IV.3.6 Analysis of Particle Properties

The separated 3D volumes were then analyzed by the method *Label Analysis*. The tabular data was exported as xml or csv data file. The parameters that should be exported were chosen from a large list of volume properties (important parameters listed in Table IV-1).

*Table IV-1. Important parameters of the final AMIRA workflow exported datasets. * including the start and end position in xyz coordinates + the orientation in Phi and Theta (OrientationPhi and OrientationTheta); Phi is defined as angle of vector to z-axis (0° – 90°) and Theta as angle within xy-plane (-180° – 180°).*

Parameter name	**Definition**
BaryCenterX, BaryCenterY, BaryCenterZ	XYZ coordinates of the center of mass
OrientationPhi, OrientationTheta	Orientation of the largest Eigenvalue
Volume3d	Voxel volume
BoundingBoxDx, BoundingBoxDy, BoundingBoxDz	Dimension of the box surrounding the 3D volume
Length3d *	Length, the maximum of the Feret diameters
Width3d *	Width, the minimum of the Feret diameters
Breadth3d *	Maximum width perpendicular to length
Thickness3d *	Diameter perpendicular to length and breadth
BorderVoxelCount	Number of voxels touching the ROI border

IV.3.7 Filter by Measure

The separated objects contained all different kinds of features. Filtering the features with the method *Filter by Measure Range* and employing *BorderVoxelCount* = 0 left the dataset with features not touching the borders. All particles not fully inside the region of interest (ROI) were discarded. By applying the same method again and filter by *Volume3d*, the respective particles could be separated into single (small volume) or connected/unseparated particles (large volume).

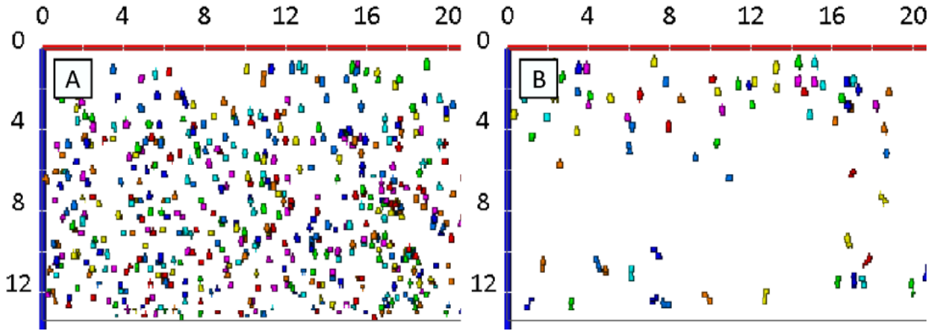


Figure IV-8. Example of separated fiducials after filtering to A: small volumes, B: larger volumes (excluded). Random colorization for better visual separation. Scale in µm.

In particular with the fiducials, there occurred some problems when filtering by a volume range, as distorted fiducials sometimes possessed the same volume as two connected fiducials (Figure IV-8). The solution for further evaluation was to filter the fiducials first to small volumes, to get rid of the major part of connected particles, and then treat the remaining connected particles as outliers by using the median and not the average size values for the correction factor calculations (Chapter IV.4) at the given height.

IV.3.8 Finalization

Apart from the so far described general steps, cropping of the images was sometimes necessary. This was especially important after alignment or when there were strong brightness differences due to charge effects that mostly happened on the start or end of the x-direction (scanning direction). Additionally, the filter of *BorderVoxelCount* = 0 was not working all the time when the image was not cropped, leaving some cut particles.

The so reconstructed images showed aligned ellipsoids.

The reconstruction with AMIRA took 1 – 2 days per dataset. In total, 24 datasets with particles and fiducials were evaluated.

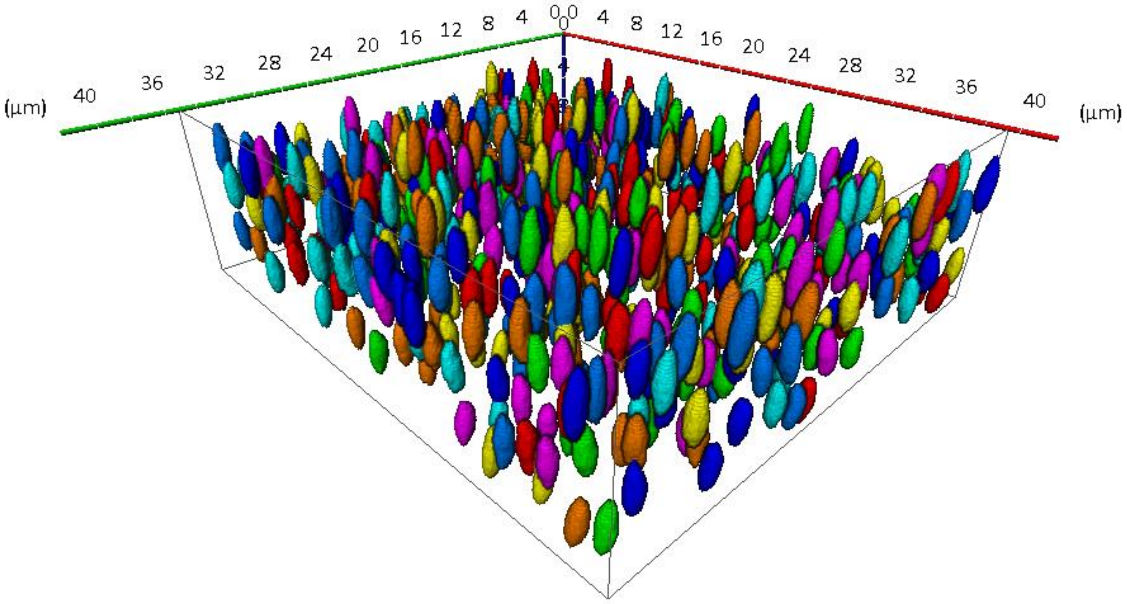


Figure IV-9. 3D reconstruction by AMIRA in perspective view; random coloration to visualize separated particles. Scales in µm.

IV.4. Distortion Correction of Reconstructed Particles

The 3D reconstructed particles displayed differences in length and shape within the reconstructed volume (Figure IV-10 B). These differences could not be found in particle dimensions extracted by 2D SEM image evaluation of recovered particles (compare SEM image and reconstruction of DB14 0 % stretched particles; Figure IV-10 A and B). Thus, the 3D reconstruction did not show the true shape. It was distorted.

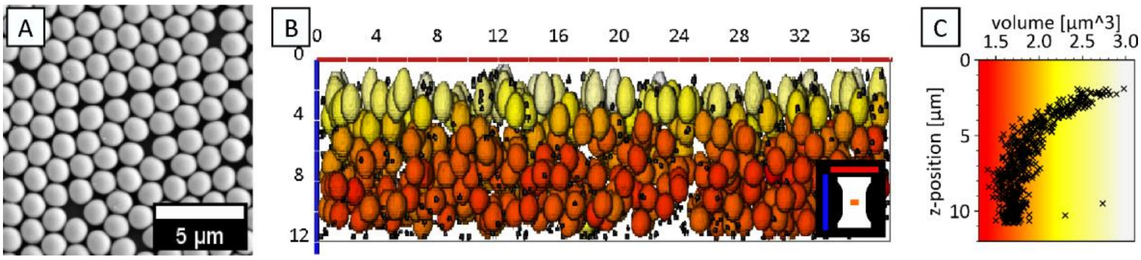


Figure IV-10. Combined figure of A: Electron microscopy image of 0 % stretched particles (spherical, DB14). B: Reconstruction of those particles (x-axis red, z-axis blue; in µm): Inset of the position of the imaged area on the stretched foil. C: Volume of the reconstructed particles dependent on their z-position (color range of 1.5 µm³; Colors at specific values: 1.4 µm³ red, 2.15 µm³ yellow, 2.9 µm³ < white/grey; z-position is the center of mass of the 3D volumes).

The distortion displayed an elongation of the reconstructed particles in slicing direction (z-direction). The initially set constant slice thickness of 40 nm (defined by the cutting program) was not reached and a lower actual slice thickness was gained. Additionally, the distortion was decreasing with increasing slicing depth. The reason for this behavior was found in the ongoing adaption of the foil to the vacuum conditions inside the microscopy chamber. This enclosed a shrinking of the foil in stretching direction (z-direction, slicing direction). Although, shrinking could also appear in x- or y-direction, it was not noticeable in all the datasets.

The particles reconstructed from the first slices (Figure IV-10 B, starting at top, z-height 0) differ drastically in volume from the particles reconstructed from slices later in the series (Volume dependent coloration by *Colorize by Measure*; large volume yellow to low volume red). Therefore, I implemented a z-dependent distortion correction for the reconstruction.

The individual correction factors were calculated, depending on the slice position, with the help of the fiducial particles (previously used for the slice alignment). The fiducials are supposed to be spheres (344 ± 13 nm CV 3.8 %) as the maximum AR by 2D SEM image evaluation of the fiducials was 1.06 from > 400 evaluated particles. The idea was that the fiducials would remain spherical during the stretching process, as SiO_2 particles are not easily deformed. When correcting the distortion of the fiducials back to a sphere, the resulting calculated correction factors could as well be used for the particles, as their reconstruction was from the same image stack.

IV.4.1 Python Script Calculated Correction

For convenience, the z-position-dependent correction factors were calculated using a Python script (code see Appendix VIII.6.2).

Within the python script, the data tables of the fiducials that were exported from the AMIRA software were loaded to a *Pandas* data frame to access the respective columns. The dataset was cleaned from empty rows, which were the result of filtering the dataset (see Chapter IV.3.7).

Next, the correction factor was calculated by dividing the z-dimension of the 3D volume representing the fiducial (z-dimension of the bounding box) by the respective diameter of the fiducial (minimum of x- or y-dimension of the bounding box). An explanation to the values used can be found below in Chapters IV.4.1.1 and IV.4.1.2.

After calculating the correction factor for each fiducial, the values were binned in z-position. To perform the binning, a bin size for the z-axis was defined (*zBinSize*). Then the *zBin* values of the fiducials were calculated using a floor division of z-coordinates of the fiducial center (*BaryCenterZ*) and *zBinSize*. The resulting zBin value defined to which bin of the z-axis the particle was count to. The definition of the bin size can be found in Chapter IV.4.1.3.

The values for every bin were calculated as median to take care of outliers in the dataset. The number of fiducials in the bins was usually more than ten and just a few times lower at the start or end of the dataset.

The correction factor for every bin was transferred to a corrected z-position of the images in the file stack. Bins without correction factor, especially bins at the border where no fiducials were evaluated, were set to the nearest correction factor in the list. The corrected z-positions of the images were then exported to the .info-file of the image file stack.

By loading the image files with this new-z-height-containing .info-file in AMIRA and analyzing the volumes again (*Label Analysis*), the distortion correction of the reconstruction was finished.

IV.4.1.1 Definition of Z-Dimension used

The z-bounding-box-values (*BoundingBoxDz*) were used for the z-dimension of the correction factor. However, the bounding box (Figure IV-11 purple box) depended on the particle location during slicing and was generally smaller than the true particle dimension (maximal two times the slice thickness; here 80 nm). Another possible dimension, the length vector of the fiducial (*Length3d*), was tested but discarded, as it was not oriented directly in z-direction (Figure IV-11 yellow arrows).

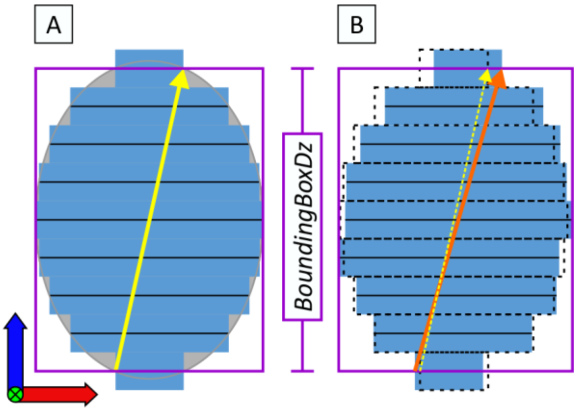


Figure IV-11. Sketches of a sliced fiducial in xz-plane. A: The spherical distorted fiducial (grey) was sliced (black lines representing the physical slices) and the volume was then reconstructed in 3D images (blue area). The bounding box (purple) and Length3D vector (yellow) are shown. B: Misalignment of slices led to a bounding box that is a little bit larger than in A (but same BoundingBoxDz value). The length vector is as well larger with slightly different orientation (angle to z-axis, compare yellow and orange arrows).

IV.4.1.2 Definition of Diameter used

The diameter used was defined as the smallest dimension of the fiducial in x- or y-direction (bounding box values *BoundingBoxDx* and *BoundingBoxDy*). The smallest dimension was chosen to counterbalance possible misalignment of the fiducials in x- or y-direction (Figure IV-11 B). However, this diameter definition will be incorrect when the misalignment is simultaneous in x- and y-direction. The mean fiducial diameter of the SEM images was explicitly not used as the fiducials in foil were reconstructed smaller (Figure IV-12 blue dashed line). An overall median value of the smallest dimension in x or y was also not used as in some slices of the image stack, the fiducials were imaged less brightly, leading to a smaller diameter and z-dimension. Surprisingly, those smaller diameters and z-dimensions were seen in different foils of the same slicing job at the same height (Figure IV-12, at z-position 8.5 µm; red frame). However, the resulting correction factor (using both “smaller” dimensions) was again similar to the others.

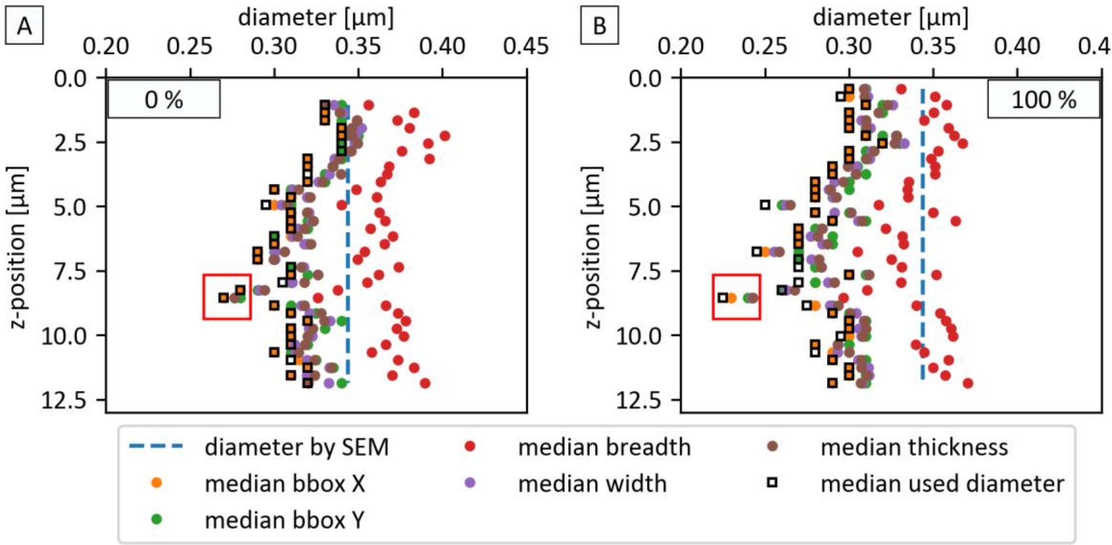


Figure IV-12. Comparison of different possible diameters for the correction factor. A: 0 % stretched sample; B: 100 % stretched sample from same slicing job but different foils. Red frames show smaller diameter on same z-position. Diameter of fiducials similarly distributed for both foils but shifted to lower values for B, 100 % stretched sample. Bin size 0.3 µm.

Other definitions of the diameter (*Width3d, Breadth3d,* and *Thickness3d*) were tested but discarded, as they are dependent on the Feret diameters of the particle. The minimum of the Feret diameters is *Width3d*, while the maximum is *Length3d*. *Width3d* and *Legth3d* are independent of each other and normally not oriented directly in x-, y- or z-direction (compare Figure IV-11 orientation of length).

Breadth3d is the “largest distance between two parallel lines touching the object […] lying in a plane orthogonal to [*Length3d*]” (cited from the internal AMIRA program help library).

Thickness3d is the distance orthogonal to *Length* and *Breadth3d*. Therefore, *Breadth3d* and *Thickness3d* are dependent on the orientation of *Length3d*. However, as the orientation of *Length3d* can differ very easily in near spherical volumes when slight misalignment occurs (Figure IV-11), the parameters were not used to define the diameter.

Note here that the most accurate diameter would be the x- or y-dimension of the fiducial at the image slice where the respective fiducial shows its maximum area. Unfortunately, this value could not be extracted from the dataset.

IV.4.1.3 Definition of Bin Size

The z-dimension of the sample is divided into bins. The bin size (*zBinSize*) used was 0.3 µm, which was a bit smaller than the smallest diameter of the fiducials found in the SEM image evaluation (0.31 µm). By choosing a bin height of 0.3 µm, the only possible position for the center of the fiducials was in an x-y-"plane" (monolayer), and not on top of each other.

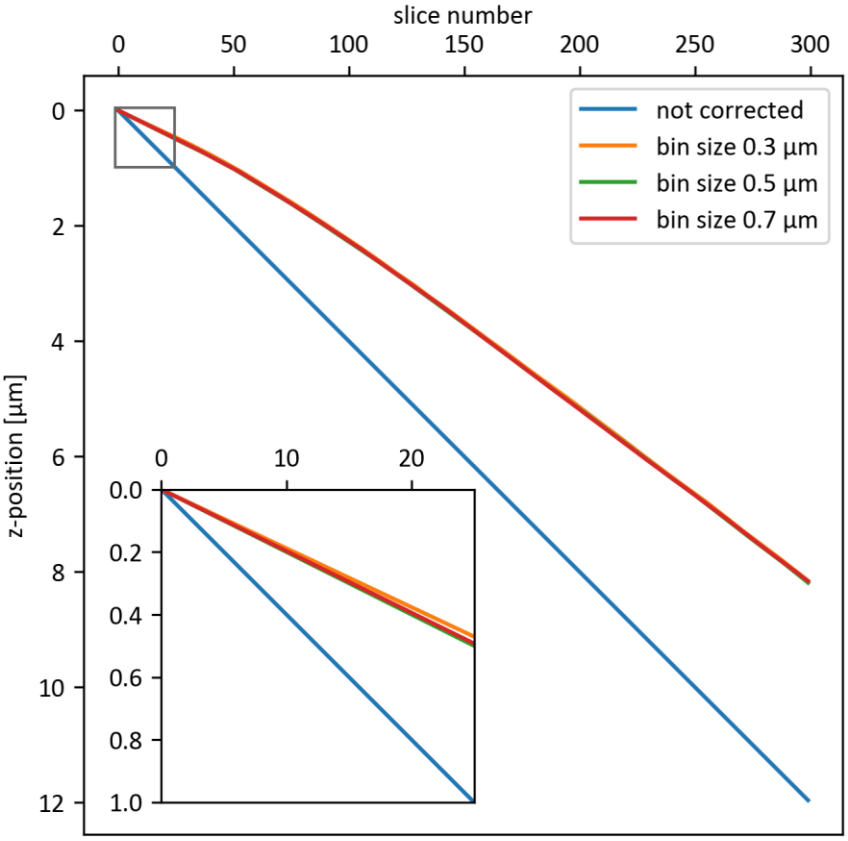


Figure IV-13. Plot of the slice number and z-position from 0 % stretched particle sample with different bin sizes. Only in the zoom-in, a deviation is visible which had no effect in the end. Within all datasets, the mean reduction of the z-position of the last image was about 74 ± 4 %. The correction of the shown dataset (compression of 68 %) was one of the highest to show the differences in bin sizes. The beginning showed a large correction/compression, which was later a more or less stable correction of the z-positions.

However, tests with bin sizes of 0.5 µm, around 1.4 times the diameter of the fiducials, left the possibility that two separated fiducials above each other were count to the same bin (max 2 fiducial planes). A bin size of 0.7 µm, double of the diameter, would give the possibility of maximal 3 fiducial planes. Either way, those larger bin sizes tested showed no significant differences in the corrected slice positions nor in the reconstructed images and particle values. A small deviation of the final image on the z-positions with different bin sizes was visible only in the beginning of the dataset where the distortion was large (Figure IV-13).

IV.5. Results of Fiducials

The correction of the reconstruction gave access to the volume properties of particles and fiducials. The corrected fiducials were compared to the reconstructed fiducials to validate the correction. It was expected to get a more narrow size distribution after correction.

A comparison problem was that the fiducial parameters *Length3d, Width3d, Breadth3d,* and *Thickness3d* displayed values dependent on the Feret diameters that were not fixed in orientation. The orientation of the fiducial particles (defined by "the eigenvector of the largest eigenvalue" [111]) was changing drastically in its angle to the z-axis (parameter *OrientationPhi,* Figure IV-14 A). Originally, the orientation was nearly vertical (< 10°) while after the correction the orientation was mostly horizontal (> 80°). This shift of the orientation was due to an underestimation of the z-bounding box of the fiducial as explained later. Without the underestimation, a more homogenous histogram over all angles (0° – 90°) can be expected.

With the shift of the orientation, completely new directions for the Feret diameter-dependent values were present. A comparison of these values was, therefore, not reasonable. However, parameters independent from the Feret diameters, like *Volume3d* or the bounding box values, could be compared (Figure IV-14 B and Figure IV-15).

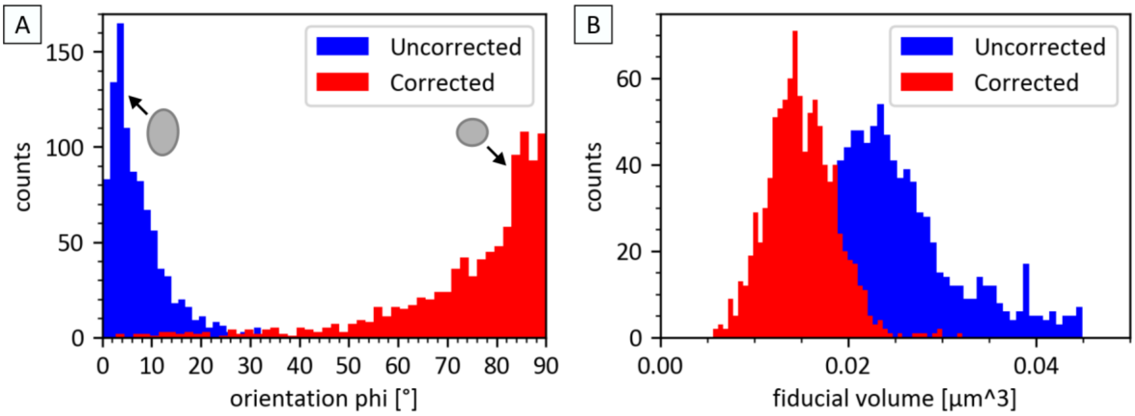


Figure IV-14. Histograms of A: orientations with insets of approximated optical representation (ellipses), and B: volumes of the fiducials with uncorrected data (blue) and corrected data (red). Values from 0 % sample.

As expected for the correction, the fiducial volume distribution narrowed and shifted to smaller values. The corrected values of the x- and y-bounding box should be and were almost the same with the uncorrected values (no altered directions), but differed slightly due to the grid change in interpolation (Figure IV-15 A and B). As expected, the spread of the uncorrected bounding box in z-direction decreased and the distribution shifted to smaller values after the correction.

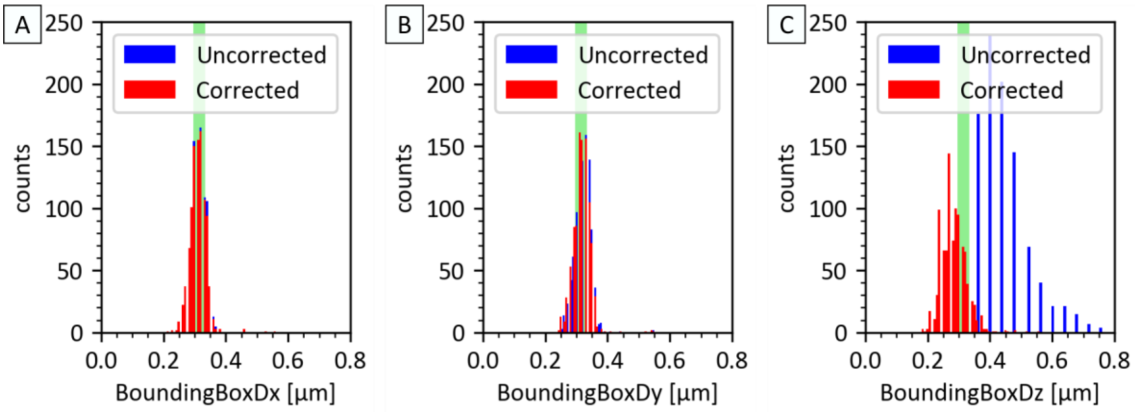


Figure IV-15. Histograms of bounding box values. A: BoundingBoxDx in x-direction, B: BoundingBoxDy in y-direction, and C: BoundingBoxDz in z-direction. The quantiles of BoundingBoxDx were equal to BoundingBoxDy with 0.3 (25 %) and 0.33 µm (75 %). The underlying green box is the region spanned by these quantiles. The maxima of BoundingBoxDz deviated from this region. Values from 0 % (spherical) sample.

However, by evaluating the corrected fiducials one can see that the new *zBoundingBox* values were not as large as the x- or y-bounding box values (Figure IV-15, green area) and smaller than the computed values using directly the correction factor on the fiducials at given z-positions (Figure IV-16).

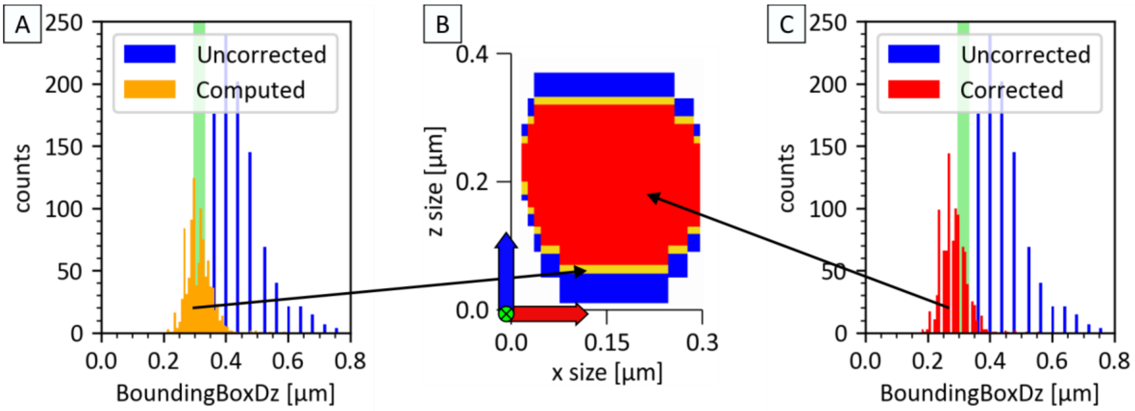


Figure IV-16. Visualization of interpolation. A: Computer calculated values during preparation of .info file. B: Slice through a fiducial during correction using Ortho Slice visualization in AMIRA (xz-plane). The uncorrected is blue, computed yellow, and corrected red. C: Values from label analysis (reproduction from Figure IV-15 C). The size of computer calculated bounding box z-value is more close to the x- or y-bounding box values.

This difference arose from the evaluation of the re-loaded image files with new z-positions. The re-loaded image stack had uneven z-coordinates. To evaluate the volume, AMIRA needs uniform z-coordinates. Therefore, the volume was interpolated to uniform z-positions using the method *Arithmetic* (*Result Type* = regular, size was set to the image stack values of x- and y-dimension; *z* = (image stack z-dimension (nm) / 10 + 1), *result channel* = value (label), *expression* = A). The chosen z-value defined the new voxel to a cuboidal shape of 10 nm × 10 nm × 10 nm.

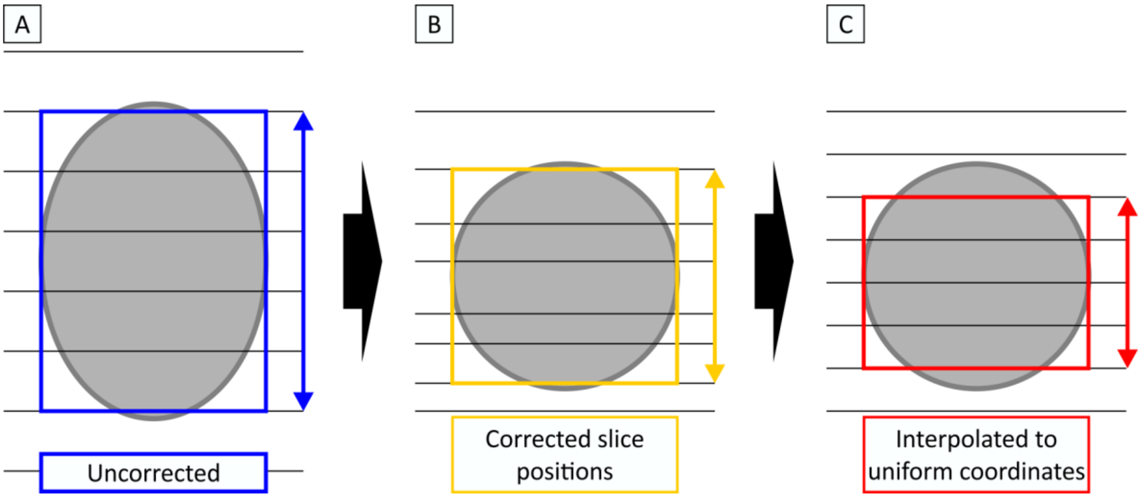

A
B
C
Uncorrected
Corrected slice positions
Interpolated to uniform coordinates

Figure IV-17. Sketch for the explanation of the smaller z-bounding box values of correction due to interpolation. The distorted fiducial (A) was corrected back to a sphere (B). The sphere had uneven slice distances. C: By interpolation to a constant grid, the bounding box size in z-direction reduced drastically.

Due to the interpolation, some parts of the particles were cut off. The amount of the reduction was hereby depending on the individual particle location respective to the new grid (Figure IV-17). A general value for the reduction could not be defined. A comment on the new grid size in z-direction and the influence on the z-values of the bounding box can be found in Chapter IV.5.1.

The difference of the calculated to corrected dataset was unavoidable, as the interpolation step of the uneven coordinates was necessary for *LabelAnalysis*.

IV.5.1 Definition of the New Grid in Z-Direction

The z-size reduction of the fiducials should be maximum one slice height of the new grid. Consequently, the z-size reduction by the *Arithmetic* method should decrease when using a smaller z-size of the new grid for interpolation. Unfortunately, this was not the case as the reduction stayed the same (around 30 nm less, see Ortho Slice visualization of fiducial, Figure IV-16 B), regardless of interpolation to 5 nm, 10 nm or 40 nm grid (or in between, when using the number of slices as a parameter). On top, the interpolation mistake was not just taking place when uneven coordinates were present. Deloading and loading the images again with an uncorrected "*.info*-file" with even coordinates did also change the z-data to smaller values when the interpolation was set to a smaller grid e.g. to 10 nm. Just the step of interpolating the uncorrected .info-file to the same even coordinates (40 nm) led to no mistake as there was no interpolation conducted within the program.

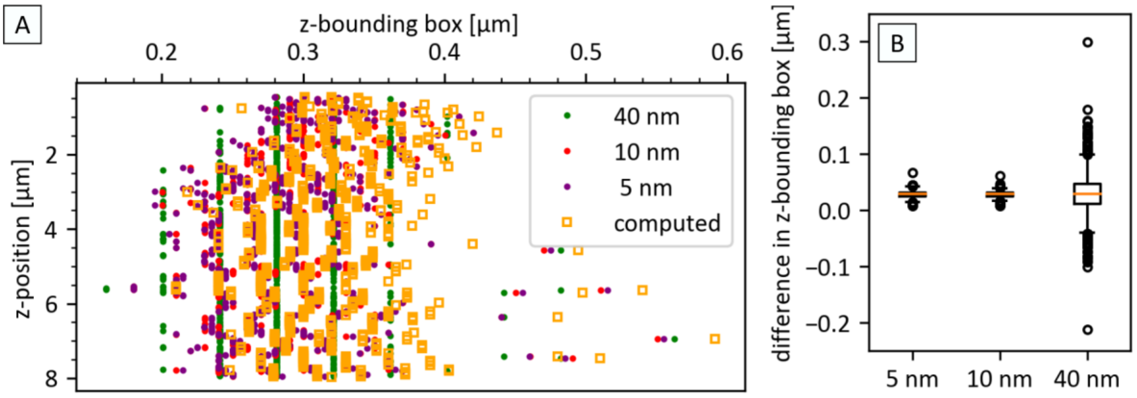


Figure IV-18. Z-values of the bounding boxes for different interpolation. A: Individual values. The computed values are shown by using the z-position values from 10 nm interpolation and were always larger than the interpolated ones. B: Boxplot of the difference of the z-bounding box values (AMIRA values subtracted from the computed values) for different interpolation grids; the difference did not change for 5 or 10 nm grids. For 40 nm grids (larger than new slice positions) the differences were larger. Values from 0 % stretched sample.

Finally, for the interpolation a value of 10 nm was chosen, as the difference to the computed values was quite constant (around 30 nm less, Figure IV-18 B) and the voxel became a cube shape of 10 nm × 10 nm × 10 nm. The interpolation with 5 nm gave small differences as well but the resulting dataset was too large for further computation (> 20 GB). The interpolation to 40 nm (larger than the new slice positions) showed a very broad distribution for the difference in bounding box z (even the z-box enlarged sometimes).

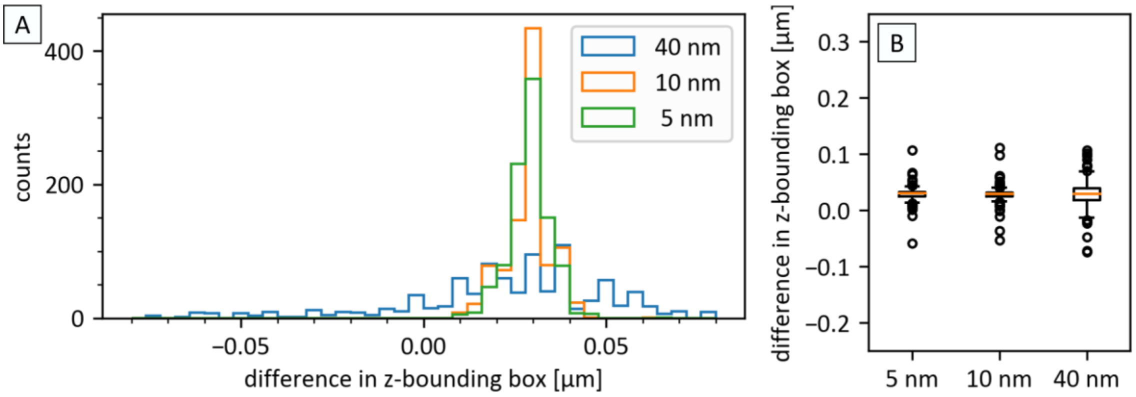


Figure IV-19. Position corrected differences of z-bounding box values. A: Histogram. Mean is about 30 nm. B: Position corrected boxplot of the difference. Values from 0 % sample.

This broadening was mainly due to different indexes of the interpolated particles in comparison to the computed ones. Sorting the particles with their x- and y-position and then plotting them again resulted in less but still broad distributions for 40 nm interpolation (Figure IV-19). Surprisingly, the distributions for 5 and 10 nm interpolation widens when doing the sorting before difference calculation. That means, the one-by-one comparison of the fiducials for different interpolation was not reliable to the number, as the index of the fiducial might change during interpolation (e.g. shift of the center to higher z-positions).

IV.5.2 Checking the Correction of the Fiducials

With the correction implemented in the previous chapter, the fiducials bounding box AR (*zCorrector*) should be corrected to 1 to represent a spherical particle. As seen in Figure IV-20, the computed values were much closer to 1 when compared to the uncorrected values. As expected, the median values of the computed bounding boxes were 1 (Figure IV-20 mean value for computed, dark yellow diamonds). The correction was, therefore, sufficient when looking solely on the aspect values of the bounding box.

The bounding box AR for the corrected particles (AMIRA values) was lower than the computed values as previously discussed. The difference in distortion depended on the z-position was corrected to a constant average value (Figure IV-20 mean value for corrected, dark red diamonds: 0.93 ± 0.11).

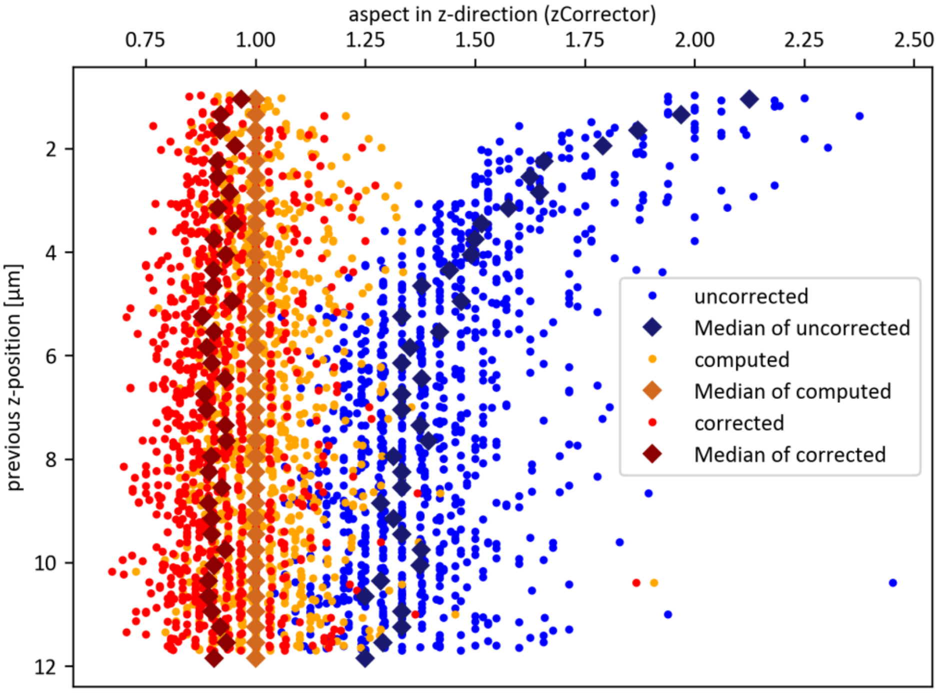


Figure IV-20. Aspect of fiducials in z-direction. Uncorrected data blue, computed orange/brown, and corrected red. Mean value for computed: 1 ± 0 and for corrected: 0.93 ± 0.11. Values from 0 % sample.

The correction with the fiducials should, therefore, be sufficient and due to the median of the computed values of 1, no further correction with the help of the fiducials could be made. Note that using the corrected values of AMIRA for further data enhancement resulted in a stretching of the particles again (correction factor < 1).

IV.6. Particle Results

IV.6.1 Optical Impression

The applied correction narrowed the AR and the volume distribution of the particles. However, especially in the beginning of the datasets, where the major distortion was found, the correction seemed not sufficient. The spherical particles (SEM image evaluation AR 1.1) are still elongated (Figure IV-21 C). This means the correction factor at those positions was too low. However, this is in contrast to the previous finding for the corrected fiducials, where the z-correction was too high (see Chapter IV.5.2).

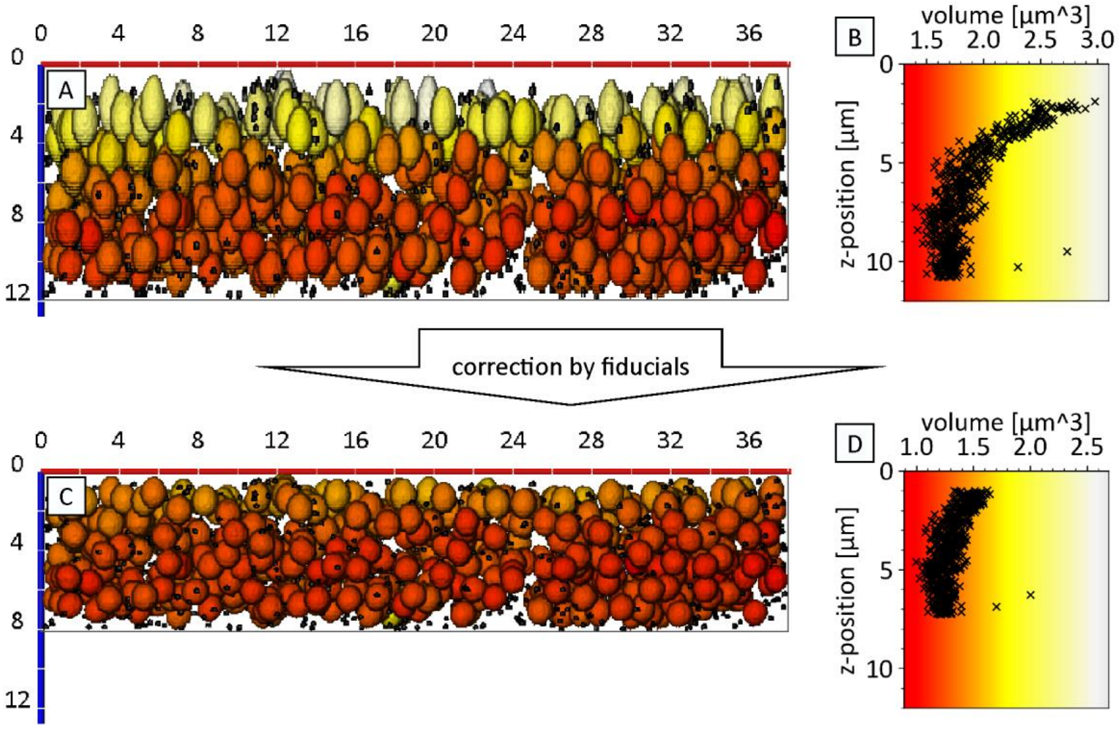


Figure IV-21. 3D visualization of unstretched particles (DB14 0 %, spherical) A: before and C: after correction. Scales in µm. Fiducials shown as black dots. Color range is equal in both images (range of 1.5 µm³), but shifted for the corrected particles to lower values. B, D: Volumes of particles respective to their particle center position. The volume distribution narrowed but the corrected volume still had a shift in the z-position of the particle centers (crosses).

The spherical particles were randomly distributed and a distortion, apart from the previously mentioned, was not visible.

For particles that spanned over more z-bins, like at an AR of 4 (DB14 100 % stretched), the distortion previously mentioned was not so drastic but as well noticeable by its volume distribution (Figure IV-22). The distortion correction at stretched particles narrowed as well as their volume distribution.

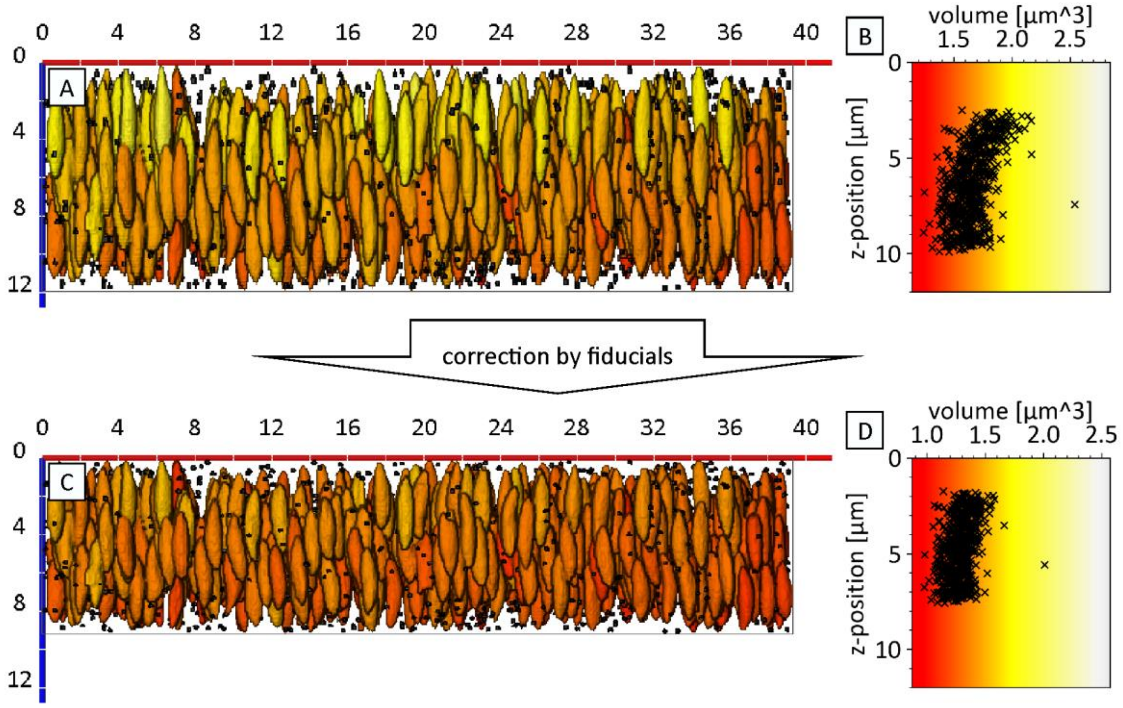
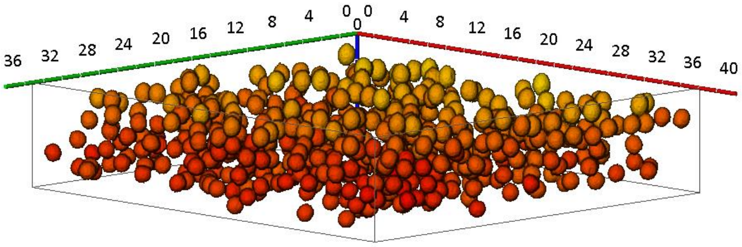
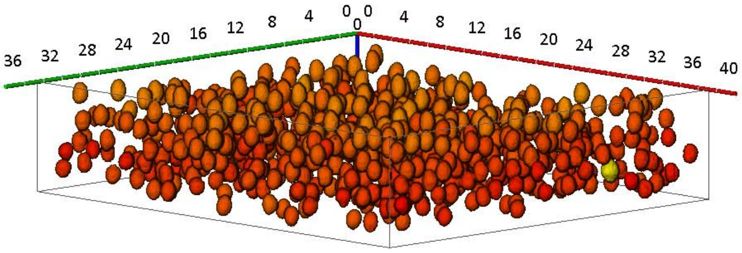
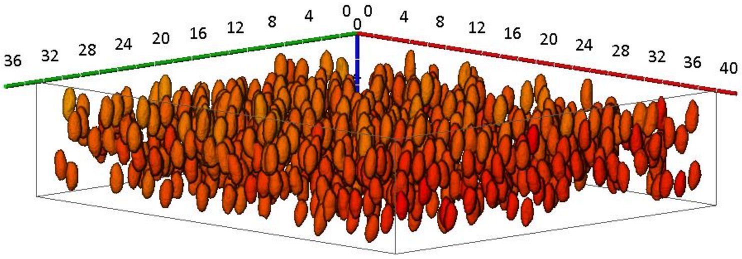
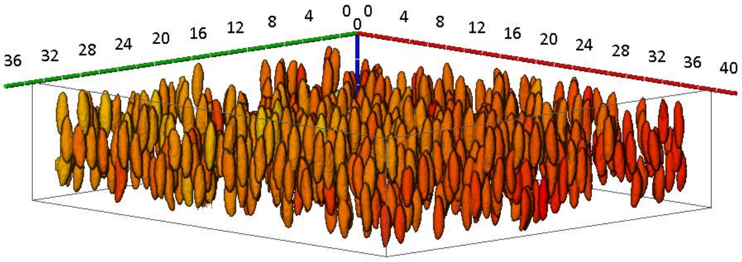


Figure IV-22. Reconstruction of 100 % stretched particles (from DB14) A: before and C: after correction. Scales in µm. On the right side were particles with a lower volume, thus they are colorized more reddish. Color range is equal in both images (range of 1.5 µm³), but shifted for the corrected particles to lower values. B, D: volume of particles respective to their particle center position. The distribution was narrower than for the 0 % stretched particles.

A lot of the 100 % stretched particles showed different kinds of shapes like bent, half domes, capped, or pointy peaks, or they had "wavy" outlines. As the wavy outline shape could not be found in the SEM images of the recovered particles, it was concluded to be a result of the reconstruction. The other different shapes resulted from particles hindering each other during stretching. This aspect is further discussed in Chapter IV.8.

Samples of 0 %, 15 %, 50 % and 100 % stretching (all from DB14 particles) led to different ARs (Table IV-2). Their corrected reconstruction images did sometimes also show deformed particles. In general, the particles appeared monodisperse and oriented in the same direction. A more detailed analysis of the orientation will be given in Chapters IV.6.3 and IV.6.4.

*Table IV-2. Overview of different stretched samples (origin from PS-PSS particles DB14, II.1.3.3.2); corrected 3D images in orthographic view, coloration volume dependent, scales in µm. Color range is equal in all images (range of 1.5 µm³), but shifted. For better visualization, the fiducials are not shown. *AR calculated by Length3d/Breadth3d. Datasets all from Batch 2 (see next chapter).*

	3D reconstruction corrected by z-box of fiducials (and x- or y-box as target diameter)	**AR***	**AR by SEM**
0 %		1.1	1.1
15 %		1.4	1.4
50 %		2.2	2.4
100 %		3.7	4.0

IV.6.2 Total Values Comparison

In this subchapter, the reliability of the reconstruction is addressed. Therefore, the values of the 3D reconstruction on the two positions per foil were discussed and compared to SEM image evaluation values of recovered particles from the respective foils (Table IV-3). Additionally, two foil batches with different content of particles inside were reconstructed to evaluate concentration differences. Note that the interpolation by *Arithmetic* function was used for the reconstruction of the particles as well. A reduction in z-height of around 30 nm had to be taken into account. In general, this error could add to the general reduction error by slicing of the particles in 40 nm steps, which could be 2 times the slice height (approx. 80 nm). The worst case would show a reduction of an individual particle of up to 110 nm in z-direction. Therefore, the values might be discussed for the whole bulk but not for individual particles.

The foils of batch 1 had 1 g particles (PS-PSS particle DB14, II.1.3.3.2) dispersed in 25 g PVA (3.85 wt% particle content of dry foil), while the foils of batch 2 had 1 g particles (DB14) dispersed in 9 g PVA (10 wt% particle content of dry foil). Thus, batch 2 contained more particles per volume (some 3D images of batch 2 can be seen in Chapter IV.6.1).

Table IV-3. Particle values from SEM image evaluation. The concentration of the particles inside the dry foil: Batch 1 3.85 wt% and batch 2 10 wt%. Values rounded to 10 nm because of voxel size.

sample	SEM length [µm]	SEM width [µm]	SEM AR
Batch 1 0 %	1.46 ± 0.05	1.37 ± 0.03	1.1 ± 0.0
Batch 1 15 %	1.76 ± 0.10	1.23 ± 0.04	1.4 ± 0.1
Batch 1 50 %	2.66 ± 0.13	1.02 ± 0.05	2.6 ± 0.2
Batch 1 100 %	3.72 ± 0.22	0.90 ± 0.05	4.2 ± 0.4
Batch 2 0 %	1.46 ± 0.03	1.34 ± 0.03	1.1 ± 0.0
Batch 2 15 %	1.71 ± 0.06	1.22 ± 0.05	1.4 ± 0.1
Batch 2 50 %	2.51 ± 0.17	1.06 ± 0.07	2.4 ± 0.3
Batch 2 100 %	3.74 ± 0.23	0.93 ± 0.07	4.0 ± 0.5

The two foil batches differ only slightly in their AR. Nevertheless, the low particle concentration resulted in a slightly higher AR for higher stretching ratios. As the foils were stretched separately, this difference could also result from a slightly different stretching of the foils. Either way, the values were within the error.

The mean values of the reconstructed corrected particles were calculated using two individual 3D reconstructions of different positions in the same sample foil. The individual results can be found in the supporting information IV.10.2. Here, the combined results

(Table IV-4) will be discussed as the difference of the 3D reconstructions were within the error range of each other and the combination was similar to the evaluation of different SEM images to generate the SEM values.

Table IV-4. Reconstructed and corrected mean values by the combination of two datasets of the same sample; combined datasets of batch 2 contained more than 1100 particles per stretching sample; due to lower slicing depth, higher distortion, and lower particle content batch 1 contained just 200 – 500 particles (lowest at 100 %).

sample	SBFI length [µm]	SBFI breadth [µm]	SBFI thickness [µm]	SBFI AR 1	SBFI AR 2
Batch 1 0 %	1.53 ± 0.11	1.34 ± 0.03	1.29 ± 0.03	1.1 ± 0.1	1.2 ± 0.1
Batch 1 15 %	1.94 ± 0.14	1.25 ± 0.04	1.17 ± 0.04	1.6 ± 0.1	1.7 ± 0.1
Batch 1 50 %	2.70 ± 0.17	1.06 ± 0.04	0.98 ± 0.04	2.6 ± 0.2	2.8 ± 0.2
Batch 1 100 %	3.75 ± 0.30	0.91 ± 0.07	0.81 ± 0.05	4.1 ± 0.4	4.7 ± 0.5
Batch 2 0 %	1.47 ± 0.13	1.34 ± 0.04	1.29 ± 0.04	1.1 ± 0.1	1.1 ± 0.1
Batch 2 15 %	1.72 ± 0.13	1.24 ± 0.04	1.19 ± 0.04	1.4 ± 0.1	1.5 ± 0.1
Batch 2 50 %	2.31 ± 0.16	1.07 ± 0.04	0.96 ± 0.05	2.2 ± 0.2	2.4 ± 0.2
Batch 2 100 %	3.42 ± 0.22	0.93 ± 0.05	0.82 ± 0.05	3.7 ± 0.3	4.2 ± 0.4

IV.6.2.1 Values of Length

As expected, the length was increasing with the stretching of the foil (see Chapter II.3.1). By comparing the length of the SEM and the reconstruction, it is noted that the 3D lengths were slightly larger at batch 1 0 % and 15 %, and smaller at batch 2 100 %. However, the reconstructed values (dataset 3D in Figure IV-23) were mostly in the range of the error of the SEM values.

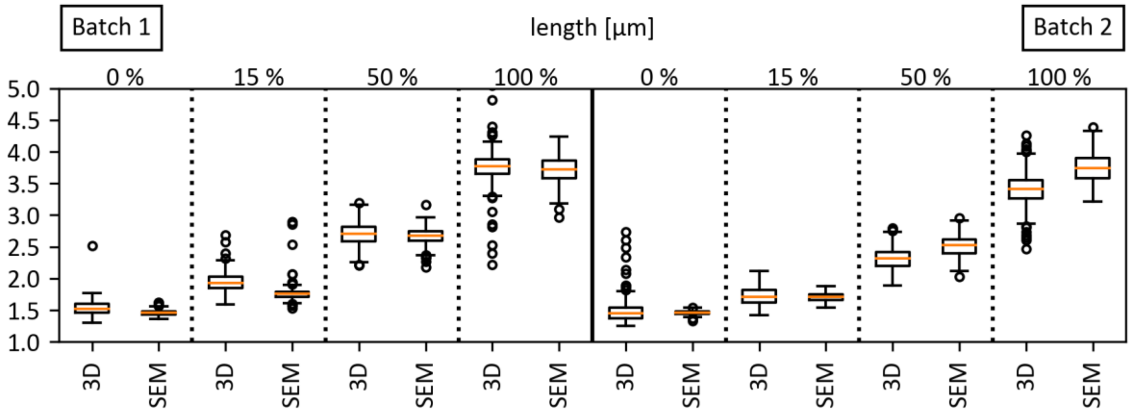


Figure IV-23. Boxplot of the length of corrected particles. Combined datasets of SBFI and SEM values. Individual datasets are displayed in supporting information IV.10.3.

IV.6.2.2 Values of Breadth and Thickness

It is known that the volume of one particle can only increase when it fuses with a second particle. However, the data of these occasions was avoided in the reconstruction as large particles were filtered. Therefore, the volume should be conserved and the other dimensions of the particle have to shrink with the stretching of the foil.

As breadth is defined as the largest diameter perpendicular to the length (see Table IV-1), the breadth values were larger than the thickness values (perpendicular to length and breadth). In comparison to the SEM values, the breadth values did fit to the SEM values better than the thickness values (Figure IV-24). The interpretation of the difference in breadth and thickness is discussed in Chapter IV.6.4.

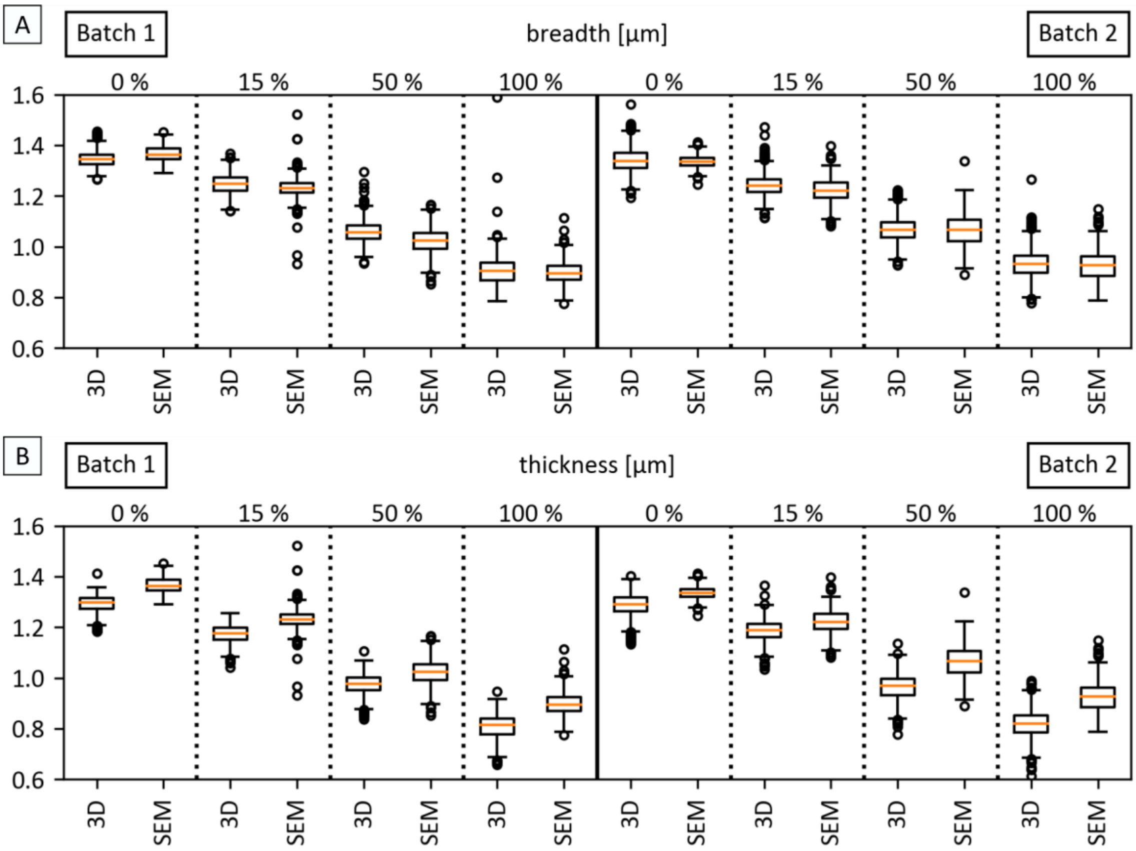


Figure IV-24. Boxplots of combined dataset values of A: breadth and B: thickness with SEM width values for comparison. The breadth values were closer to the SEM values. Individual datasets are displayed in supporting information IV.10.3.

IV.6.2.3 Values of ARs

The AR is calculated by defining a length and a diameter of the volume (similar to the definition of the correction factor in Chapter IV.4.1). Three different kinds of ARs could be calculated and compared to one another. For AR_1 and AR_2, the length is defined as the value of *Length3d* (representing direction a in the ellipsoid mathematical Equation IV.6-1) while the diameter can be either *Breadth3d* or *Thickness3d* (representing b and c direction).

$$V = \frac{4}{3}\pi \cdot a \cdot b \cdot c \qquad \text{(Equation IV.6-1)}$$

To distinguish the different ARs, AR_1 was defined as the aspect of length/breadth, while AR_2 was length/thickness. The third AR, aspect ratio of particle waist (AR_{waist}), calculated by breadth/thickness, will be discussed in Chapter IV.6.4.

In SEM images, one cannot say if breadth, thickness, or a value in-between, is visible as the diameter of the particle (see Chapter IV.6.4.1). Therefore, to compare 3D values with SEM values, a mean diameter (d_{mean}) of the respective ellipsoid was calculated and used in the calculation of $AR_{intermediate}$ (Equation IV.6-2, Figure IV-25).

$$AR_{intermediate} = \frac{length}{d_{mean}} = \frac{length}{\frac{breadth+thickness}{2}} \qquad \text{(Equation IV.6-2)}$$

The calculated AR shows that for batch 1, the $AR_{intermediate}$ were slightly larger than the SEM values while the batch 2 values were very similar to SEM. The SEM ARs were, however, always within the error range of the $AR_{intermediate}$.

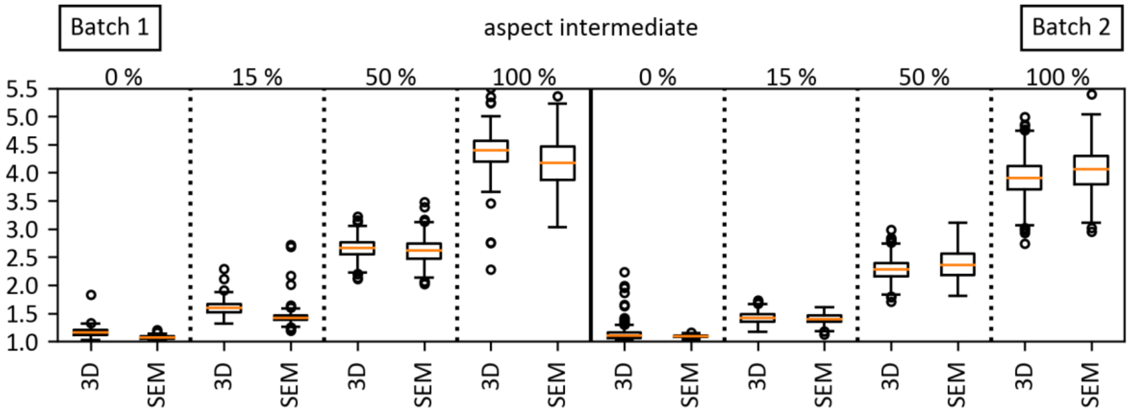


Figure IV-25. Boxplots of combined SBFI values of intermediate AR and SEM mean AR. The boxplots of the individual datasets of AR_1, AR_2 and $AR_{intermediate}$ can be found in the supporting information IV.10.4.

IV.6.3 Orientation Distribution of the Particles

The particles seemed to be oriented in one direction, respective to the stretching direction. With the vertical embedding of the foils, the stretching direction was more or less represented by the z-direction.

IV.6.3.1 The Problem with AMIRA's Value *LengthOrientPhi*

The orientation distribution in z-direction is represented by the length orientation angle phi (φ, range 0° – 90°). As already mentioned, the orientation of length is dependent on the voxel dimension and, therefore, quantized. A value of 0° (directly in z-direction) was not possible for the largest Feret dimension (compare to sketch of sliced volume, Figure IV-11 A).

With the given value *LengthOrientPhi*, calculated by AMIRA, it was found that 5.72958° was the lowest value for phi in all samples. A 5° angular difference might count for 130 nm or 13-pixel-deviation in a 1.5 µm particle. This was surprisingly high. Additionally, there were no angles to be found between 6° and 20°. It seemed that this was not just from quantizing but also from internal value calculation of the AMIRA program. As AMIRA did calculate the length vector orientation to the z-axis wrongly or at least binned the values in too large bins, *LengthOrientPhi* could not be taken to evaluate the particle orientation.

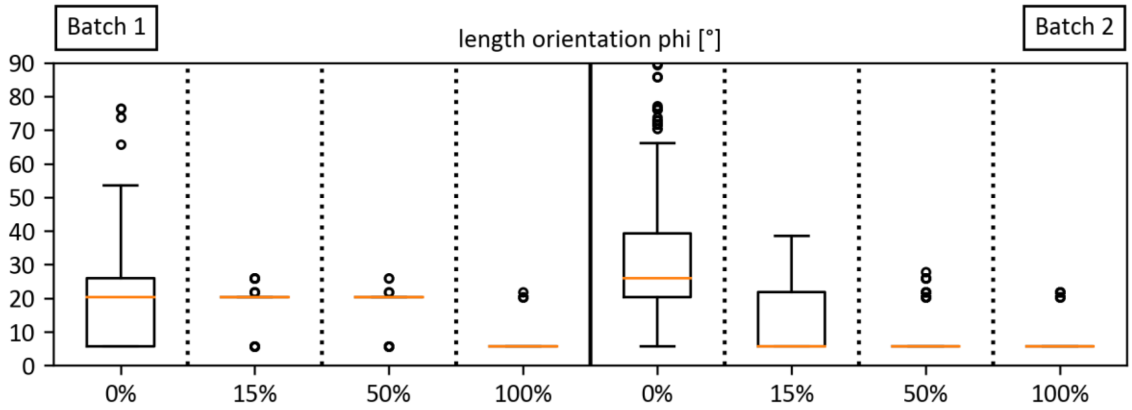


Figure IV-26. Boxplots of the AMIRA value LengthOrientPhi. The orientation respective to z-axis was quantized and did not differ much in the individual datasets. The combination had a very low spread. Individual datasets of the different samples can be found in SI IV.10.5.

Note: The orientation values of *BreadthOrientPhi* and *ThicknessOrientPhi* did not show such binning within the AMIRA dataset, while *WidthOrientPhi* did show it as well.

IV.6.3.2 Self-Calculated Orientation Phi and AMIRA's *OrientationPhi*

As *LengthOrientPhi* was not useable, a more trusted orientation was calculated by using the given start and end coordinates of the length parameter (from AMIRA dataset) and basic math equation (Equation IV.6-3) for the angle φ between vectors (u, v: 3D vectors, here u for z-axis (0, 0, 1), and v for the length vector):

$$\varphi = \cos^{-1}\left(\frac{u \cdot v}{|u| \cdot |v|}\right) \tag{Equation IV.6-3}$$

The calculated phi orientations showed values lower than 5° and between 6° – 20°. The orientation was narrow for the stretched particles 15 – 100 % while, as expected, it was wide for unstretched, near-spherical particles (0 %, no fixed direction of length inside sphere). The orientation got closer to z-direction 0° for higher stretched particles, which was expected, as the particle size grew and the length orientation got more defined (Figure IV-27 A).

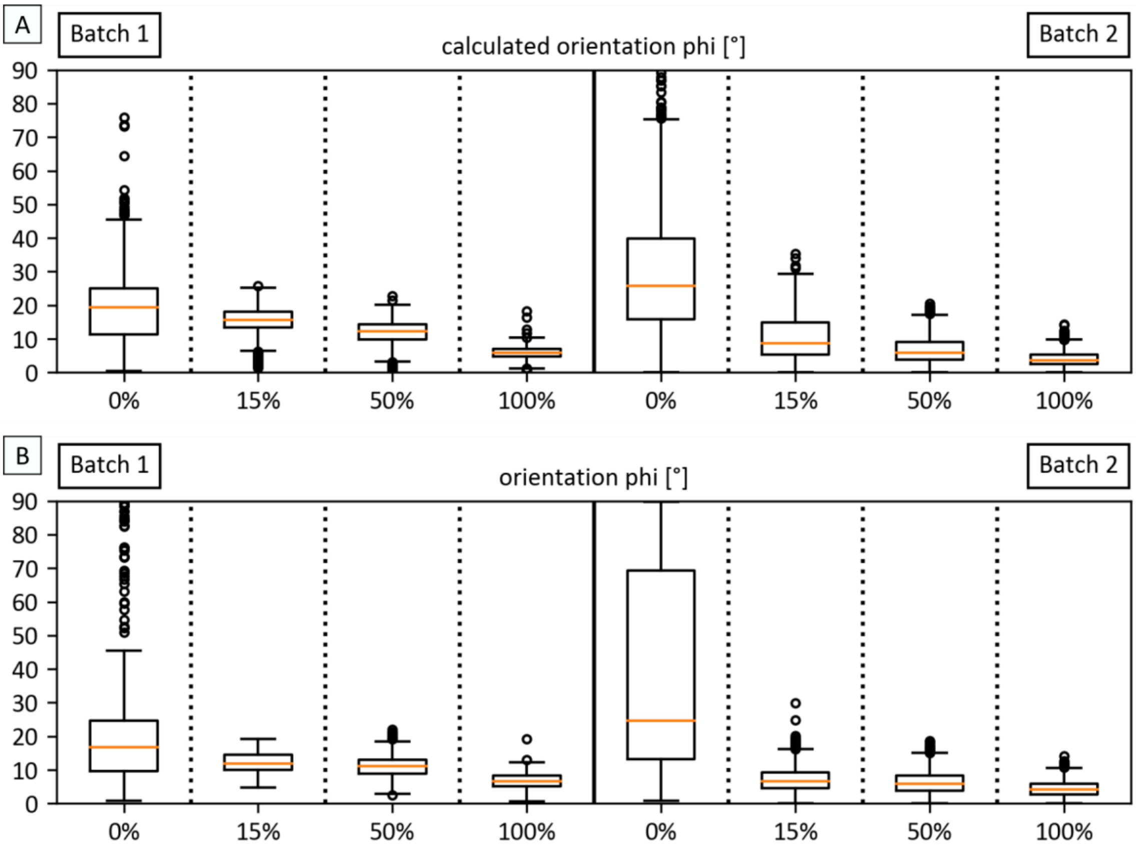


Figure IV-27. A: Boxplots of calculated orientation of length vector; data has distinct values due to voxel size (see Figure IV-29 polar plot) but does show values of more homogenous spread compared to parameter LengthOrientPhi. B: Boxplots of orientation calculated by AMIRA using the Eigenvalues of the 3D volume.

An additional value representing the orientation of the 3D object (particle volume) was calculated by AMIRA. This parameter, *OrientationPhi* (Figure IV-27 B), depicting the largest

Eigenvalues of the volume and has the benefit, that it is not quantized. Therefore, if one does not need the true orientation of the length vector but wants the orientation of the 3D object, a suitable substitute is the *OrientationPhi* value from the AMIRA dataset.

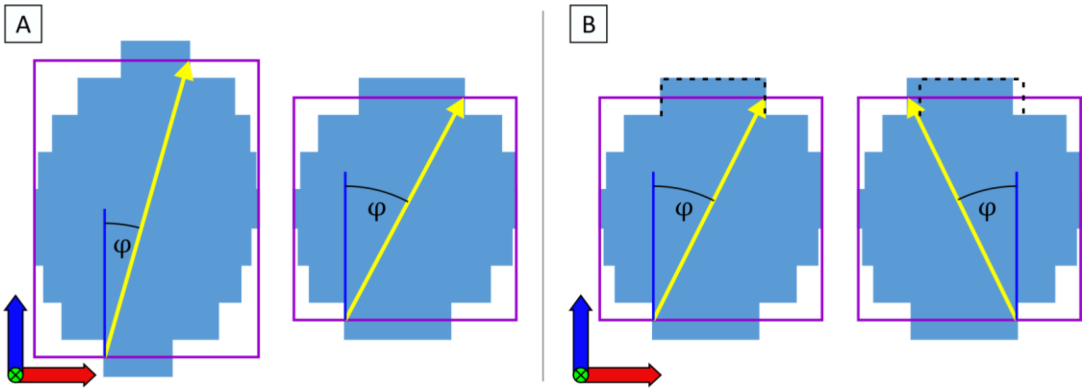


Figure IV-28. Sketches of xz-planes of particles oriented in z-direction. The bounding boxes (purple) and the length vectors (yellow arrows) are visible. A: Particles have different ARs. The orientation angle phi (φ) is larger with lower AR. B: Same particles but upper slice misaligned on the right one (black dots). Orientation changed not much in value of phi (φ), but a lot in theta (θ, angle in xy-plane, here the x-axis contribution changed to negative values).

To fully describe the orientation of a particle in order to see if the particles of one foil were pointing in the same direction, the second orientation angle theta (θ), being the angle in the xy-plane, needed to be evaluated as well. The distribution of theta can differ drastically with the amount of stretching (different ARs), capped particles or misalignment (Figure IV-28).

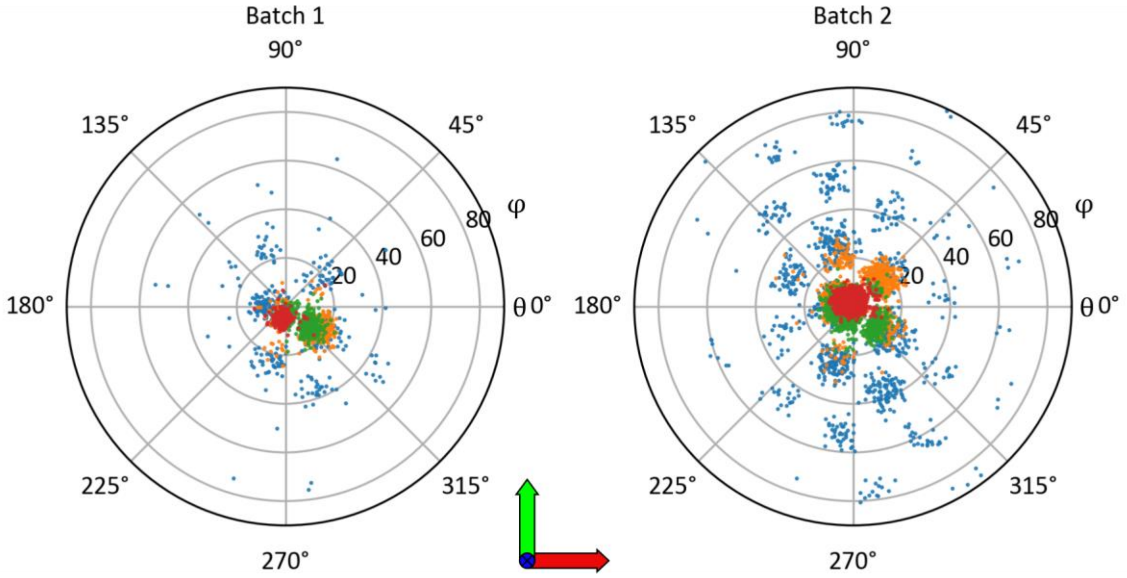


Figure IV-29. Polar plots of the calculated angle theta (θ) with angle phi (φ) as radial values (0° – 90°) of combined datasets. The particles precess around a value for the stretched particles of batch 1. The quantization because of the voxel size is visible. Samples: 0 % stretching (blue), 15 % (orange), 50 % (green), and 100 % (red). The plots of the individual datasets can be found in the supporting information IV.10.5.

The orientation in z-direction narrowed naturally with higher AR. The orientation precessed around a value pair of angles. The combined values for two individual reconstructions still showed a preferred orientation. For calculated values, the quantization was visible (Figure IV-29). For batch 2, the spread of the angles in xy-plane was larger. In general, the more the particles became spheres, the higher was the spread in theta, as the orientation of the length vector changed easily (compare 100 % to 15 % and 0 %). The orientation of length for near spherical particles was more random. A more defined picture was derived from the orientation values of the Eigenvalue (*OrientationTheta* and *OrientationPhi*, Figure IV-30). Here, the quantization was not visible and thus, the orientation was narrower.

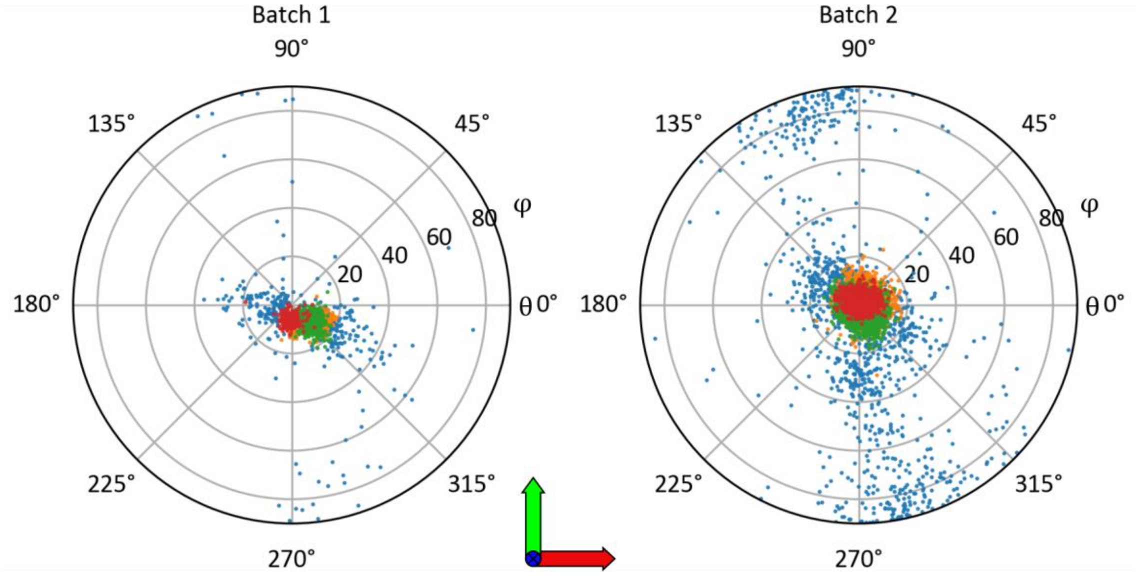


Figure IV-30. Polar plots of AMIRA orientation values OrientationTheta and OrientationPhi. Dataset contains the combined values of two individual datasets. Samples: 0 % stretching (blue), 15 % (orange), 50 % (green), and 100 % (red). The plots of the individual datasets can be found in the supporting information IV.10.5.

Regardless of self-calculated or by use of AMIRA's *OrientationPhi/OrientationTheta* values, the angles for stretched particles 15 – 100 % were all oriented in one direction. The orientation was clearer for stretching to 100 % (higher AR), as the length axis was more defined. With 0 % stretching, the orientation varied over all angles because the particles were similar to a sphere and had no defined long axis available (perfectly represented at batch 2.1, see supporting information IV.10.5).

IV.6.4 The Third AR

This subchapter investigates if the particle can be described as a flat or as a full rugby ball. Depending on the two values *Breadth3d* (b) and *Thickness3d* (c) a third AR (AR_{waist}) was defined. In a pure mathematical one-dimensional deformation of a sphere, the two parameter used in AR_{waist} would be the same (b = c) resulting in an $AR_{waist} = 1$. A flattening of one or the other side would, however, result in an $AR_{waist} > 1$. The AR_{waist} cannot become smaller than 1 as *Breadth3d* is defined as the larger dimension of the waist.

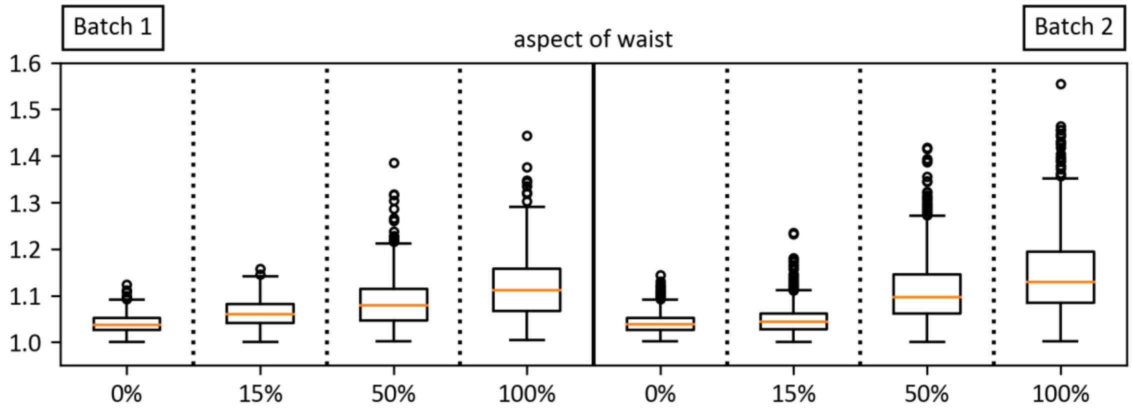


Figure IV-31. Boxplot of aspect ratio of particle waist (AR_{waist}) for the different samples. Combined datasets; individual datasets see supporting information IV.10.6.

The mean AR_{waist} was not 1, but maximum at around 1.1 for all samples (Figure IV-31). A factor of 1.1 translates thereby to a difference of the two axes in the order of 50 – 100 nm (5 – 10 pxl). While for low strain particles the mean AR_{waist} was close to 1.05 (2 – 5 pxl), the distribution did increase with foil strain. Therefore, I conclude that the reconstructed particles, at high strain, could not be described as regular ellipsoids with only two different axes anymore. The shape of high strain particles was more like a flat rugby ball with three different axes, or at least had one flattened short side. This finding for the shape was very important for and visible in the monolayer assembly in Chapter V.

IV.6.4.1 Comparison to SEM Data

The investigation of the width distribution gained from SEM images of recovered particles in combination with the average minor diameters (*Breadth3d* and *Thickness3d*) of the respective SBFI evaluation showed that the major peak of the histograms aligns with the average breadth value of the particles (Figure IV-32). The average thickness from the SBFI evaluation can be found next to the major peaks as a small peak (Figure IV-32 B) or as a shoulder of the main peak as well (Figure IV-32 A). As the major peaks can be related to the breadth values that are also the largest minor diameters, the particles can be count as lying flat on the substrates.

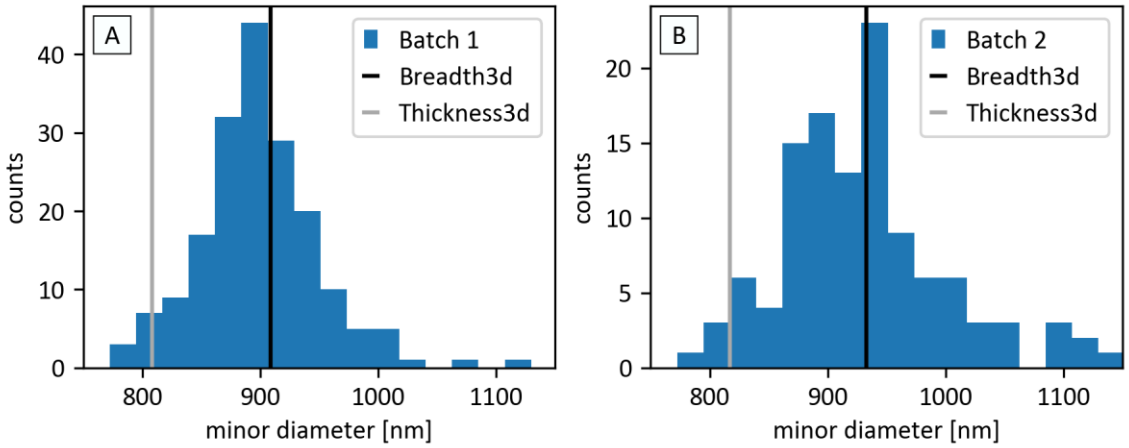


Figure IV-32. Minor diameter histograms of SEM evaluation of the big SBFI particles (DB14) from 100 % foil strain. The histograms include the average values of Breadth3d and Thickness3d from the SBFI evaluation of the different batches. A: batch 1, B: batch 2. A, B: Bin size (= pixel size of evaluated images): 22.33 nm.

With the result that indeed the thickness was visible in the SEM dataset an investigation on the data of the small 108 % stretched particles of Chapter II.3.1.1 (inter-sample reproducibility investigation of FN269) was done. Unfortunately, for the minor dimensions they showed only broad distributions. I related this on the pixel resolution of the evaluated images (22.33 nm/pxl at the big DB14 particles of this chapter and 7.443 nm/pxl at the small FN269 particles). Therefore, a detailed investigation of the minor diameter distribution at small particles and a statement on $AR_{waist} > 1$ was impossible (see supporting information IV.10.7).

Note that the SEM comparison and search for different minor diameters relied on the particles being randomly oriented with their minor axes either parallel or orthogonal to the respective sample surface. However, intermediate orientations (partially on the side-lying, tilted particles) give intermediate minor diameter values. Thus a broad minor axis distribution, as visible in the histograms, was expected.

IV.6.4.2 Deformation of Particles in Different Regions

The particles might experience different forces in different areas of the foil. With the investigation of the AR_{waist} on the xy-plane, a probable deformation of the particles at different regions could be identified.

As the following evaluation was dependent on the position within the foil, selected individual datasets will be discussed. Figures for all datasets can be found in the supporting information IV.10.8.

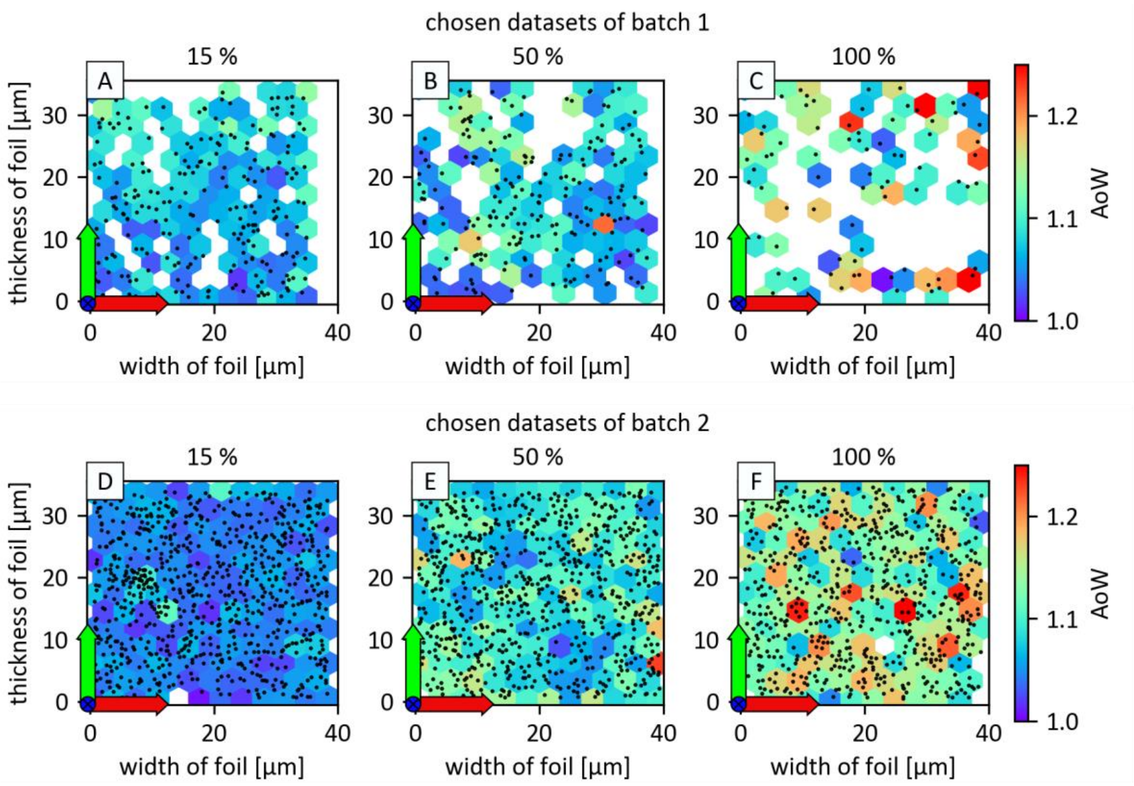


Figure IV-33. XY plots of AR_{waist} from different chosen samples: Batch 1: 15 % (Batch 1.2), 50 % (Batch 1.2), 100 % (Batch 1.2); Batch 2: 15 % (Batch 2.1), 50 % (Batch 2.2), 100 % (Batch 2.2); not shown: 0 % as AR_{waist} distribution was very homogenous. Note that the thickness of the foil is the y-axis while the width is the x-axis. The foil/particle stretching direction is towards the reader. Figures for all datasets can be found in the supporting information IV.10.8.

The visualization shows that the size of the AR_{waist} parameter was not dependent on the xy-location (Figure IV-33). A mostly even distribution was found for nearly all datasets and especially in the much more particle-containing datasets of batch 2. However, this was not the case at batch 1 15 %. Only there, the AR_{waist} raised to higher values along the y-axis (constant color shift along the thickness of the foil (y-axis)). That means the particles experienced a higher deformation on one side of the field of view. As this behavior was just present in the batch 1 15 % sample, it could not be further investigated.

Apart from the size of AR_{waist}, it was necessary to investigate its orientation in xy-plane (orientation of *Breadth3d* in xy-direction, parameter *BreadthOrientTheta*). A preferred orientation of the AR_{waist} would be a strong hint of uneven deformation (compression or stretching) in another direction, e.g. along the thickness or width of the foil, despite the major stretching direction of the foil.

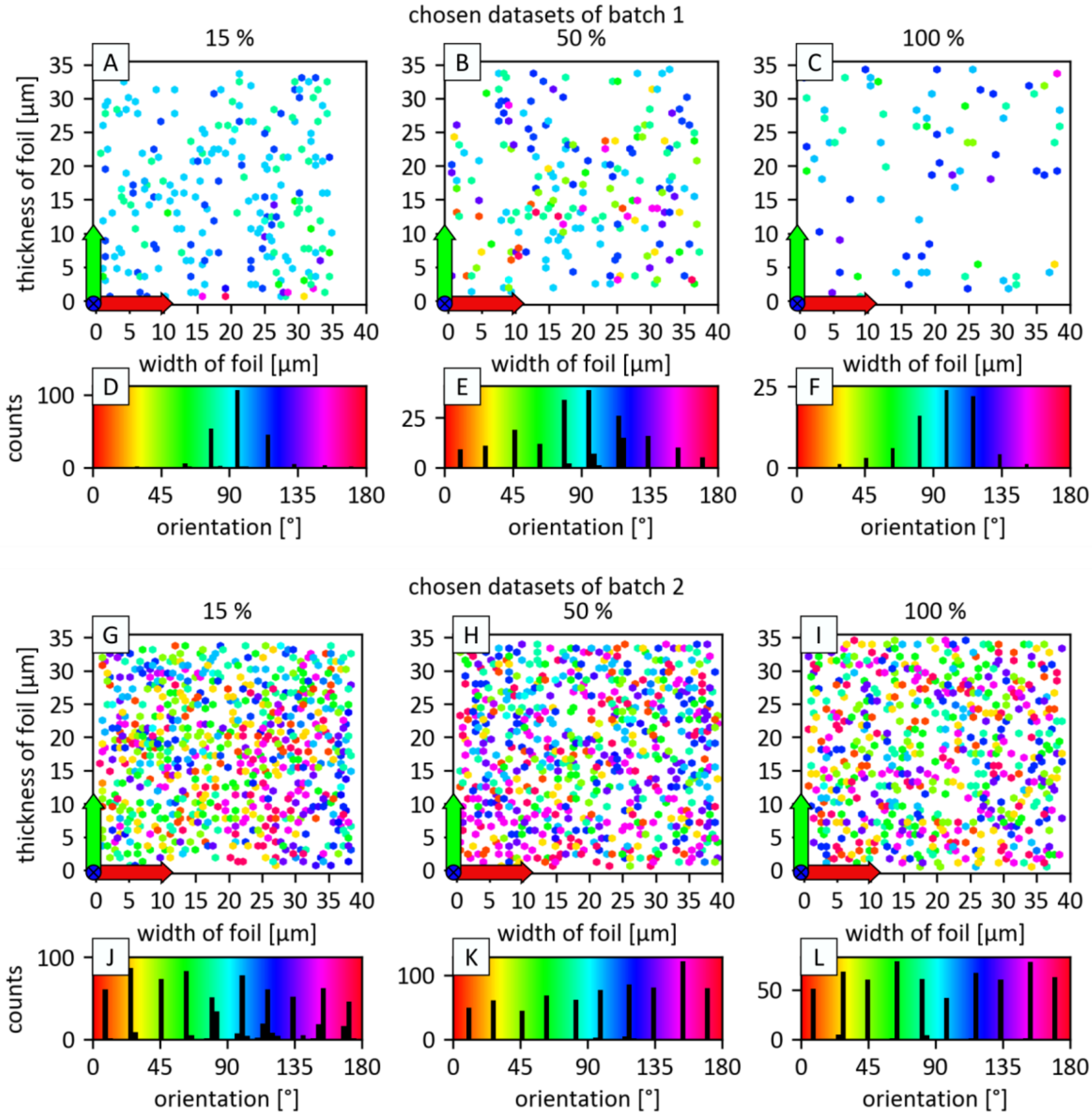


Figure IV-34. Orientation of Breadth3d on xy-plane. Batch 1 (A-F) did show a preferred orientation for breadth to more or less 90°. Batch 2 (G-L) had no orientation preference. For the plots, the values of BreadthOrientTheta were changed from range -135° – 45° to the range of 0° – 180° by adding 180° to all values smaller 0°. Below the xy-plots (A – C, G – I), the respective color bar encloses the histogram of angles (D – F, J – L). Plots of individual datasets can be found in the supporting information IV.10.8.

The visualization of the AR_{waist} orientations at Batch 1 and batch 2 did show different behaviors within their orientation distributions. Batch 1 displayed a preferred orientation in y-direction (around 90° to x-axis) over the whole sample area. This hints on an additional compression of the particles of batch 1 along the width of the foil.

Note that the batch 1 15 % sample where a shift in the size of the AR_{waist} along y-axis was present did not have different orientations of the AR_{waist} with the shifting. Thus, the compression was just a bit stronger along the y-axis of this sample.

In contrast to the samples of batch 1, batch 2 showed a more random distribution with no preferred orientation (Figure IV-34 G – L) which, therefore, could not be referred to an additional compression in thickness or width of the foil.

IV.7. Reconstruction of Differently Oriented Foils

In Chapter IV.2.1, I introduced the samples with the particles embedded in such a way that they were cut perpendicular to their long axis. In other words, the particles stood upwards. To evaluate the process of reconstruction further, pieces of the exact same foil (100 % strain, foil of batch 1, 3.85 wt% particle content) were embedded not just vertically (Figure IV-35 A) but also horizontally (B) and in a diagonal manner (C).

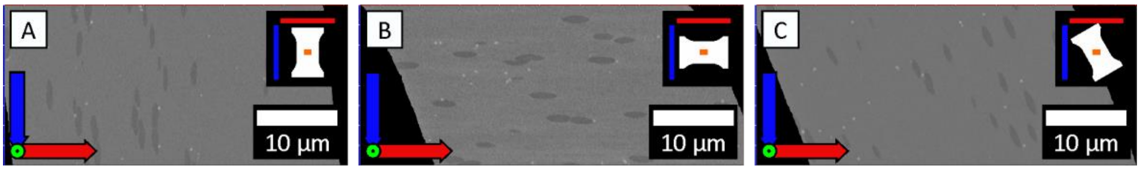


Figure IV-35. Ortho slices in xz-direction of A: vertical, B: horizontal, and C: diagonal embedded foil (x (red) = knife direction, z (blue) = image stack) with particles in the direction of the embedded foil (stretching direction). The here shown images of the stack shift after alignment via fiducials (black triangle areas) show the importance of alignment. Insets show the position of the sample within the stretched foil (orange).

The foil pieces were from an area right next to each other in the center of the stretched foil and embedded together to form a sample. The sample was prepared 6 months prior cutting and stored in a desiccation box with orange gel (silica gel with humidity indicator from Merck). Additionally, 3 h of equilibration time in the vacuum chamber (shrinking of just 2 – 3 µm, normally > 5 µm) helped to minimize the distortion by shrinking during cutting. Nevertheless, a distortion was still present. The cutting, imaging, and reconstruction were performed in the exact same way as before. For the differently oriented foils, alignment by the fiducials was very important as the unprocessed image stacks showed misalignment. Especially with the diagonal foil, the alignment without using the fiducials did result in vertical standing particles and completely distorted fiducials (see high-resolution image in the supporting information IV.10.9). The applied alignment shift of the slices was much larger with the horizontally or diagonally than with the vertically embedded foil (Figure IV-35 black area).

IV.7.1 Reconstruction before Correction

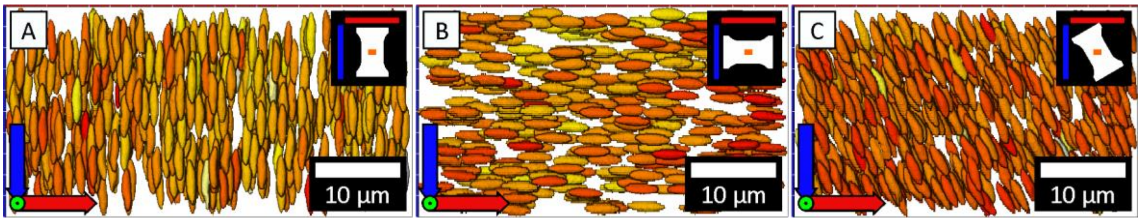


Figure IV-36. Reconstructed particles before correction (xz-view). No distortion with z-position was visible as volumes were regularly distributed. Volume coloration delta of just 1 µm³ (compare to Figure IV-22 with 1.5 µm³ range), same volume coloration for datasets (1.2 µm³ – 2.2 µm³). Insets show foil orientation.

In the 3D reconstruction (Figure IV-36), the vertical foil showed elongated slim particles, while the horizontal foil displayed thickened particles, both because of the z-distortion. For the diagonal embedded foil, the z-distortion was not clearly visible. The volume of the particles did not change drastically from the first to the later slices, as the distortion was more constant due to the long equilibration and curing times prior cutting.

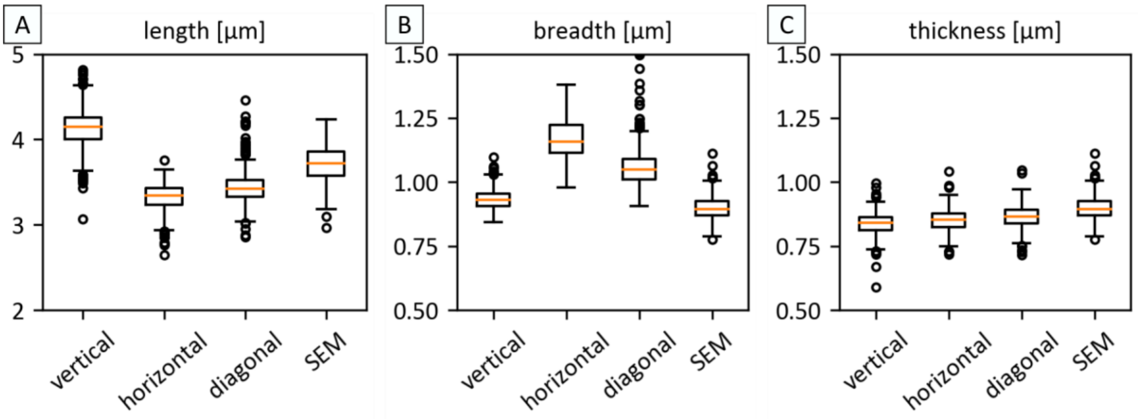


Figure IV-37. Boxplots of uncorrected values of A: length, B: breadth and C: thickness for vertical, horizontal and diagonal particles.

As expected, the length value of the vertical uncorrected particles was larger than in SEM (Figure IV-37). For horizontal particles, the breadth value was much larger than the SEM width. Both were similar in thickness to the SEM width as the distortion was just in one direction (z-direction). For the diagonal foil, the uncorrected length was lower than in the SEM but breadth was larger and thickness was similar. This was surprising, as the distortion in z should increase the length. An increased breadth could be explained by the z-direction being a part of the breadth of the diagonal particles. Unsuspected was as well, the lower length of the horizontal particles compared to the SEM, as the particles should be imaged in same dimensions. One reason could be that there was a length reduction because of the slicing in 40 nm steps, capping the tips and reducing the length.

IV.7.2 Corrected Reconstruction

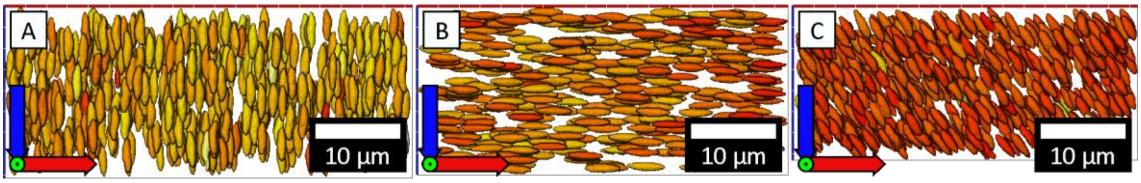


Figure IV-38. Particles after correction of the reconstruction (xz-view). Delta of color 1 µm³, but shifted to 0.84 µm³ – 1.84 µm³. The dataset of the diagonal particles (C) is volume wise much lower than the vertical (A) or horizontal (B) datasets. The particles are all oriented in one direction (polar plots in the supporting information IV.10.9).

After the distortion correction (Figure IV-38), the values of the horizontal and vertical datasets were equal to each other (Figure IV-39). The samples came close to the expected dimensions, but length and thickness were lower than the SEM values, while breadth was similar to the SEM width. Consequently, the AR_2, calculated from length and thickness, was close to the AR of the SEM, while the AR_1 (length/breadth) was lower.

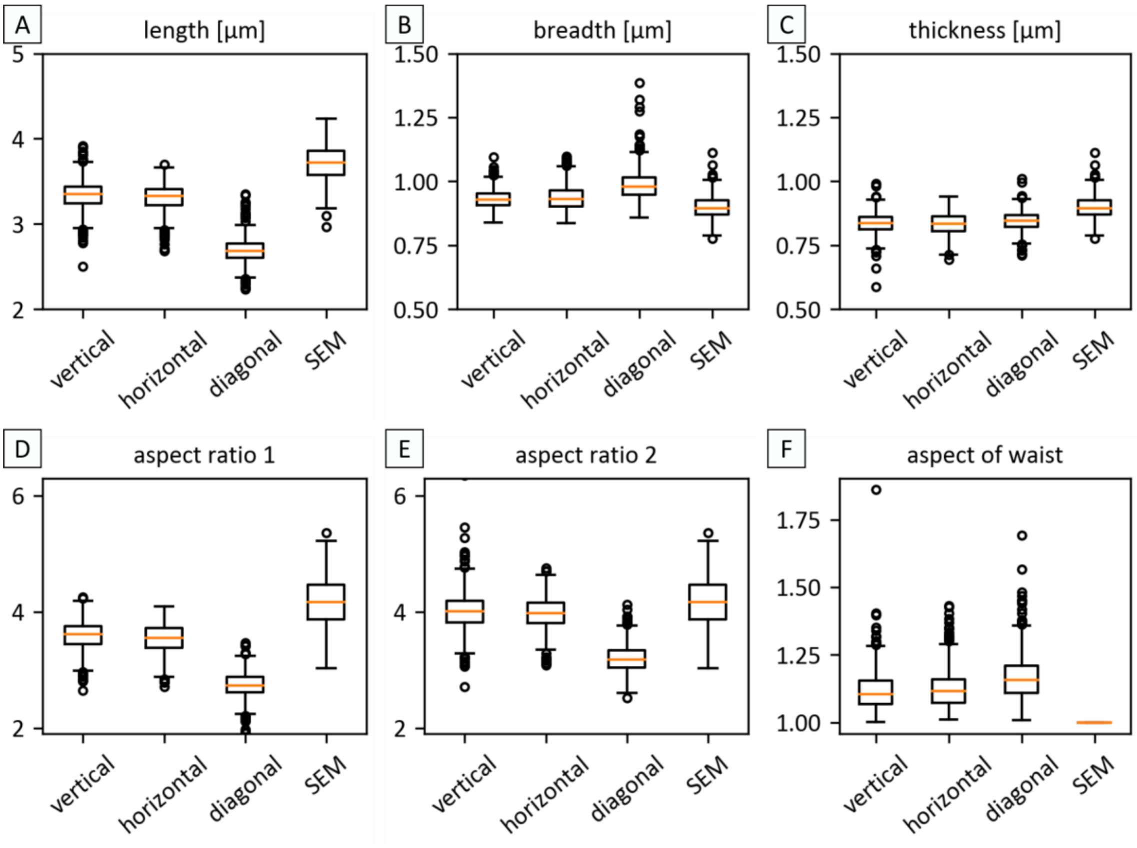


Figure IV-39. Boxplots of parameters from the different oriented foils after correction. F: The AR_{waist} of SEM values was 1, as for SEM values breadth = thickness.

IV.7.3 Diagonal Embedded Foil

While the vertical and horizontal data corrections lead to similar values, the diagonal particle reconstruction displayed the particle length dimension being much lower (Figure IV-39). Thus, the depending ARs were also lower. However, the AR_{waist} at diagonally embedded particles was much larger than the AR_{waist} of horizontal or vertical particles. This gave a hint that there was still a distortion present within this diagonal reconstruction.

By looking at the new slice positions (Figure IV-40), the corrected ones of the horizontal and vertical datasets were close to each other, while the new slice positions of the diagonal foil were much lower, resulting in a more squeezed/corrected diagonal dataset.

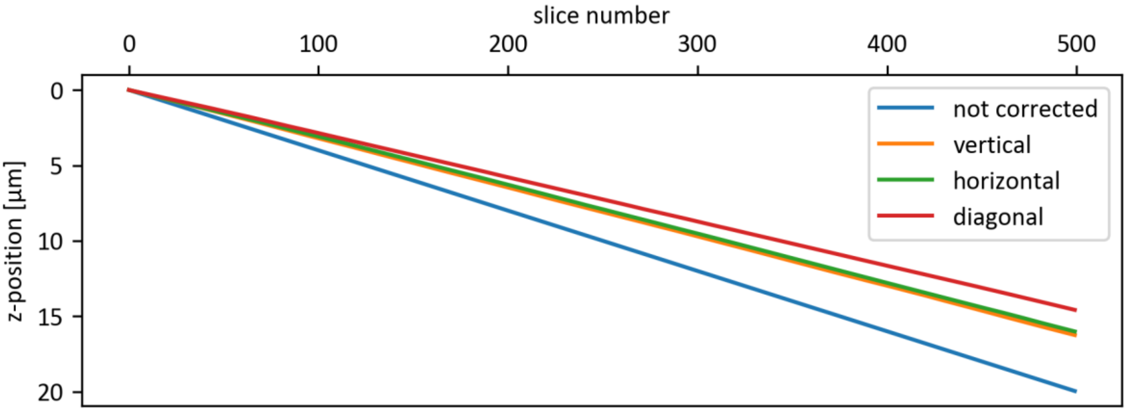


Figure IV-40. Slice positions before and after correction by fiducials. The correction of the diagonal (red) dataset was higher than the horizontal (green) or vertical (orange) datasets, resulting in lower z-positions.

A reason for the lower length and higher correction must lay within the diagonality and the cutting direction. A dependency on the shrinking was excluded as then the shrinking of vertical and horizontal data would be different too, and the slice height would differ much.

A test of using the same distortion correction from the vertical on the diagonal dataset did still not result in the same length values for the reconstructed particles. Length3d was still much lower as for horizontal or vertical datasets. Breadth showed a z-dependency, as it got larger. The thickness did not change (Figure IV-41).

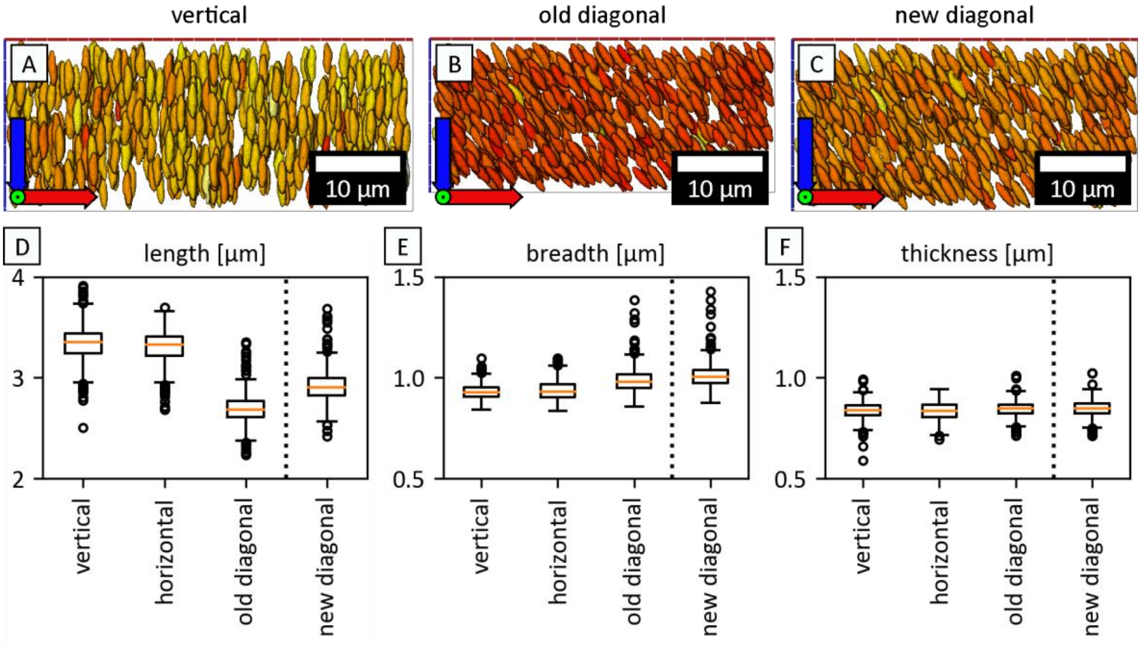


Figure IV-41. Test of diagonal reconstruction with image stack position values of vertical correction. A – C: 3D reconstructions in xz-view with the same volume coloration. D – F: Boxplots of the different samples including the new diagonal values.

The diagonal cut foil was pointing towards the blade during cutting (knife path along x-axis). When now the fiducials were dislocated in the cutting direction from slice to slice, the particle would be aligned to a much lower tilting as they actually have. This would explain the lower as-expected length of the particle prior to correction. To clarify this, further research needs to be done where e.g. the tilting could be in the direction away from the knife path. If there the particles would have a much longer as-expected length, it would be a strong hint that the fiducials were dislocated during cutting.

IV.8. Deformation by Nearby Particles

When particles are next to each other in the foil, they distort themselves during the stretching process, resulting in uneven-shaped particles. Thus, the shape of a particle is dependent on the position of other particles. Distortions did also arouse from fiducial particles (see SEM image of recovered particles with imprints of fiducials, supporting information IV.10.10) or could be from uneven strain of the surrounding matrix. As nearby particles are unavoidable and their amount increases with higher particle content, the deformation in shape by nearby particles needs to be taken into account in the particle stretching process.

The different kinds of deformation of the particles could be monitored with the horizontal-embedded foil of Chapter IV.7 and, of course, in recovered particles.

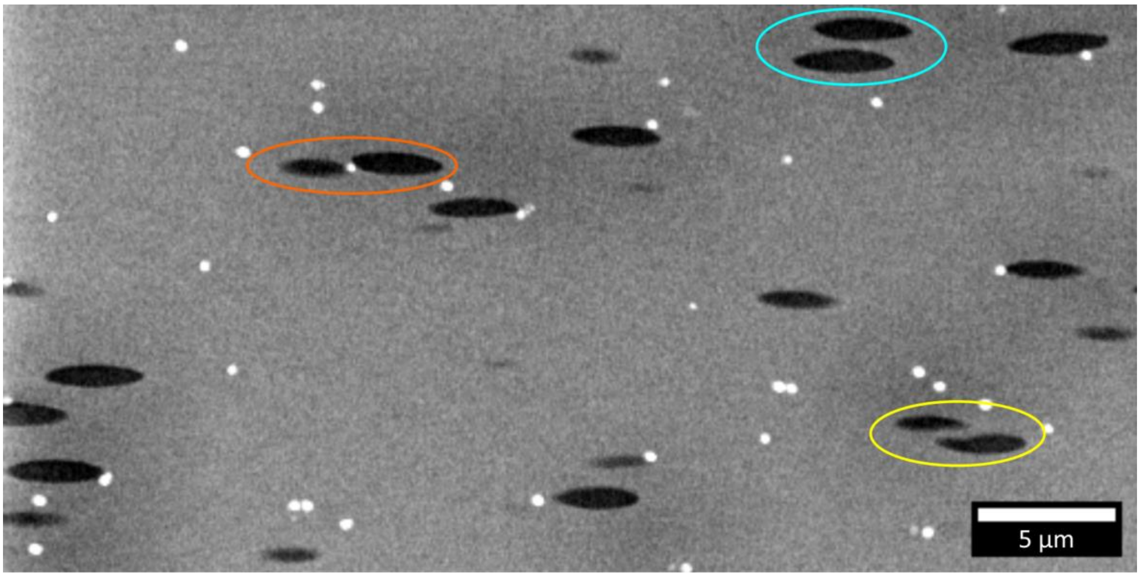
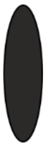
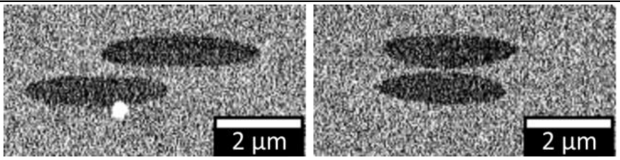
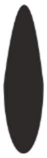
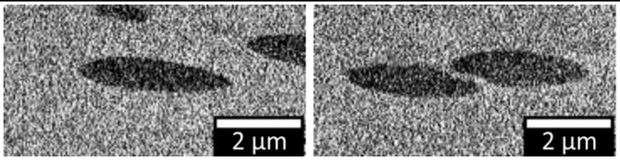
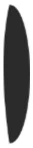
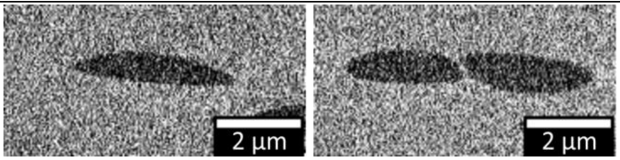
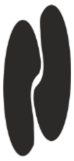
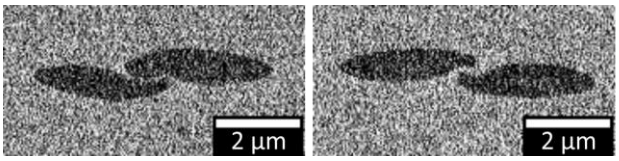
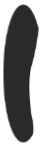
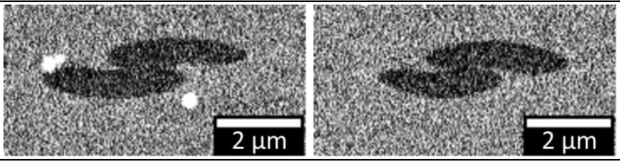
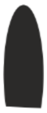
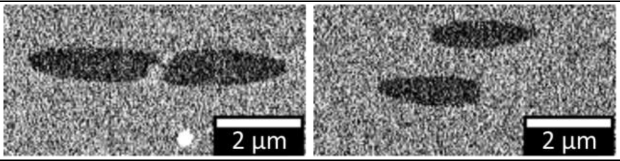
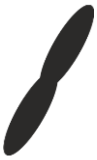
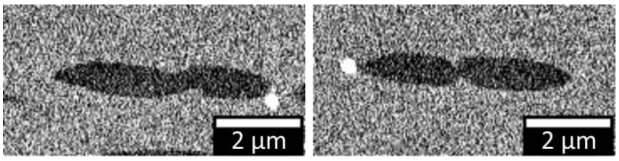


Figure IV-42. SEM image of deformed particles (100 % stretching). The particles were cut in the direction of stretching (sample of horizontal embedding, see Chapter IV.7). Even-shaped particles (cyan), capped particles (orange) and neck building particles (yellow) were all visible on the same slice.

Mostly, particles next to each other and located perpendicularly to the stretching direction did not show an uneven shape (Figure IV-42 cyan ellipse), but sometimes had a pointier tip on one side. Most of the uneven-shaped ellipsoids were capped particles, which developed when the nearby particle was in the direction of stretching (Figure IV-42 orange ellipse) and the two particles did not fuse during heating. However, in between those next-particle-location-directions, shapes developed with a pointier tip on one side, which deformed further to interlocking yin-yang-like shapes with a large body and a slim neck (Figure IV-42 yellow ellipse). This neck building and interlocking of nearby particles left one side of the ellipsoid with a smaller tip. In some occasions, this neck building resulted in bent particles, as the particle volume was deflected partially by the other particle volume.

The main shapes discovered for the stretched particles were depicted in Table IV-5.

Table IV-5. Experienced deformation in shape for single particles by matrix or one other particle. The SEM images of particles inside the foil were normalized and the contrast enhanced via Fiji/ImageJ (version 1.53c). The frequency was estimated by investigating 763 shapes on 20 slices of the vertical embedded sample.

	Shape	**In foil**	**rel. frequency**
ellipsoidal			85 %
- pointy tips			
- dome, **one flat side**			
interlocking **yin-yang**			6 %
- bent			
bullet			7 %
fused, **dumbbell**			2 %

When more than two particles were next to each other, several other possible deformations arose (Figure IV-43). However, it was not certain that deformation always occurred (see supporting information IV.10.10 for close together but evenly deformed particles).

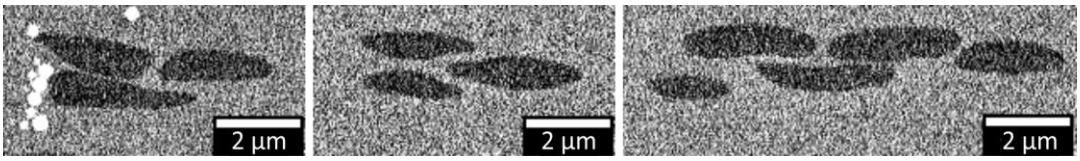


Figure IV-43. SEM images of particles in foil where multiple particles were next to each other resulting in deformations.

The uneven deformation was not just found within the samples for SBFI investigations (DB14). Other examples were the particle clusters of Chapter II.3.2.2, which formed when using a (too) high weight percentage of small particles inside the foil (around 23 wt% particle in dry foil). The particles inside those clusters were elongated and deformed to something like bricks or parallelepipeds (Figure IV-44).

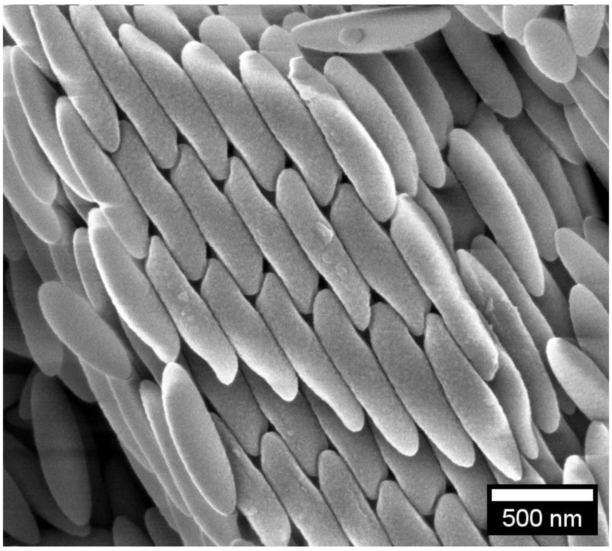


Figure IV-44. SEM image of clustered particles of Chapter II.3.2.2; Foils were stretched to 108 % and contained 6 wt% particle mass in wet 20 wt% PVA dispersion (10 : 3 PVA : PS wt ratio, 23.1 wt% particle in dry foil).

There might be a way to exploit such a deformation to gain particle sticks or bricks with flattened surfaces. It would also be interesting to force this behavior and make a composite material from such particles, as it reminds of fracture resistant composites with brick-and-mortar structure. However, experiments in this direction were not performed within this thesis and could be a subject for future projects.

IV.8.1 Quantification of the Particle Deformation

As seen in the 3D reconstruction and in Table IV-5, many particles altered each other. The proposal to gain monodisperse particles, especially with larger stretching ratios and larger wt% of particles, was certainly not true for the exact shape.

Therefore, it would be beneficial to quantify the difference of the shapes in contrast to a regular mathematical ellipsoid. A 3D quantification method might be to calculate the mathematic volume for a regular ellipsoid and compare it with the volume of the 3D reconstruction (voxel bound). Unfortunately, all particles were around 10 % less in volume by voxel compared to their calculated mathematical expression, because of missing voxels of the 3D corrected particles due to interpolation. For the mathematical expression, the values of *Length3d*, *Breadth3d* and *Thickness3d* were used as a, b, c in the ellipsoid equation (see Chapter IV.6.2.3, Equation IV.6-1). A quantification of how much the particle shape differs from a perfect mathematically described ellipsoid (more volume = flattened tips/capped particle or less volume = pointy tips) was, therefore, not possible.

Another interesting parameter for a sample would be to quantify the number fractions of uneven-shaped particles. As the ellipsoids can deform in all directions, values gained from individual slices of the datasets might not be sufficient. However, to show the importance of deformation of shape some images were evaluated by hand. The major groups of ellipsoid (including pointy and dome), interlocking (including bent) and bullet shape were evaluated for 20 images (2 positions with 10 images all 50 slices; total 763 shapes). An ellipsoidal shape was found with 85 % of the visible particle shapes. Thus, 15 % of the visible shapes were classified as not being ellipsoidal (interlocking 6 %, bullet 7 %, fused 2 %).

The amount of uneven shapes is of course underestimated, as the particles can be altered in 3D. Unfortunately, this cannot be investigated by looking on xy-planes. A possible approach to perform a quantification of the uneven shape number fractions with the complete 3D datasets might be to define an ellipsoidal shaped hull around the center of a particle with the doubled diameters of the expected particle. If there would be another particle center within this hull, the particles would be too close together to form an even shaped ellipsoid and exhibit, therefore, a deformation. However, this would not help to distinguish different shapes. Problems with this approach are of course the definition of the expected mathematical ellipsoid (which sizes for long and short axes?) and the center positions of the deformed particles (would not be the same for an even shaped ellipsoid). These theoretical considerations could not be further investigated within this thesis, but should be interesting for future work.

IV.9. Summary and Future Perspectives of Chapter IV

Within the chapter, the steps of creating a 3D reconstruction of polymeric particles inside a polymer matrix are shown. Therefore, the SBFI SEM imaging method was performed on PS particles, which were embedded in a PVA matrix (for the stretching process). The resulting image stacks were processed with AMIRA, with additional help of a Python script.

A huge help for performing the 3D reconstruction was to incorporate cesium chloride and fiducials of silicon dioxide into the matrix. The heavy metal ions of cesium helped to get a contrast of matrix polymer and polymer particles, while the unstretchable SiO_2-particles were beneficial for the alignment of the image stacks and further distortion correction.

The embedded foil underwent an in-situ shrinking because of ongoing adaption of the water sensitive PVA matrix to the low-vacuum conditions, even after an equilibration step of several hours prior to cutting.

A step-by-step workflow in AMIRA depicts the approach from the image stack to a first reconstruction. This reconstruction needed to be corrected due to the in-situ shrinking. By calculating and using a z-position dependent correction factor, the PS particles could then be reconstructed with limited distortion.

The different parameters extracted from AMIRA were evaluated and discussed. An unavoidable reduction in length during the correction of the particle reconstruction was noticed and could be explained by interpolation of uneven coordinates to even coordinates.

The properties of the particles were then compared to the values of 2D SEM image evaluation of recovered particles of the same sample foils. As the reconstructed dataset gave access to the third dimension, the true 3D shape with all three axes of the ellipsoidal particles could be discussed. While the values for length and width were lower than expected, the AR was in the range of the SEM values.

The reconstruction was done for foils with two different concentrations of particles (3.85 wt% and 10 wt% of particles in dry foil), and four different foil strains (0 %, 15 %, 50 %, 100 %), leading to different particle ARs. All the datasets showed a narrow orientation distribution in the strain direction for the major particles' axes.

Especially the highly stretched samples (100 %) showed a flattening of one minor axis and should, therefore, not be described as regular ellipsoids with two different axes. The shape of high strain particles was more like a flat rugby ball with three different axes. The lower concentrated particle foil showed a preferred orientation for this flattening perpendicular to the blade path, leading to the assumption that the foil compressed more in width than in thickness (shear thinning). This was not the case for the higher concentrated samples (sample-to-sample-variation). Therefore, the flattening of one side should be investigated further and might even be investigated using the minor diameters of recovered stretched particles.

The effect on the reconstructed size of differently oriented particle foils revealed that further research needs to be done with diagonally embedded particle-loaded foils, as much lower length values were reconstructed than for horizontally or vertically embedded particles (possible due to fiducial relocation during cutting).

Finally, a short glance was given to the shape distortion resulting from particles hindering each other during stretching. Although, considerations for a quantification of the deformations were made, the 3D datasets were not suited for such quantifications. However, the main shape distortions could be shown qualitatively inside the foil by individual 2D SEM images of a horizontally embedded particle sample image stack.

IV.10. Supporting Information to Chapter IV

IV.10.1 SI for Chapter IV Introduction – TEM Tomography

The 3D reconstruction from individual particles (Figure SI IV-1) are made from data gained from TEM tomography measurements by Dr. Markus Drechsler, Bayerisches Polymerinstitut, KeyLab Electron and Optical Microscopy (BPI), University Bayreuth. Reconstruction by group member Dr. Qimeng Song, chair of Physical Chemistry I, University Bayreuth.

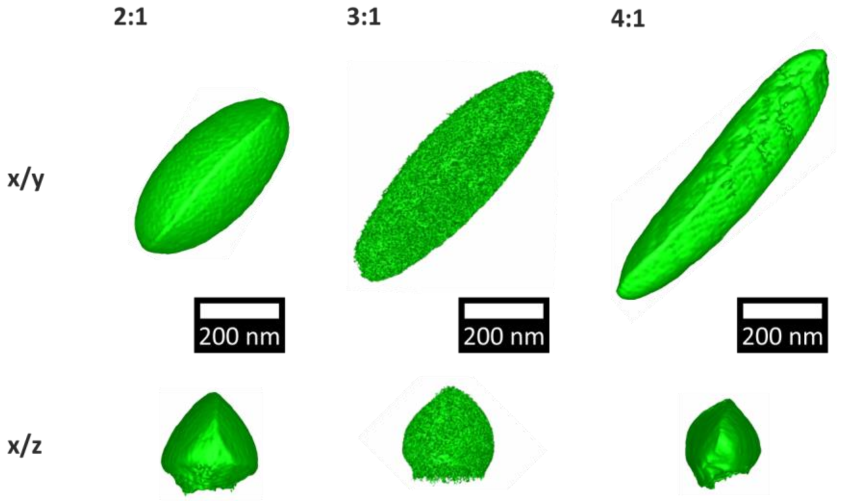


Figure SI IV-1. Reconstructed 3D images of TEM Tomography measurements on FN269 particles of Chapter II.3.1. Due to tilting restrictions and shadowing, the upper and lower part of the particles were interpolated, leading to artefacts. Viewing planes are depicted on the left.

IV.10.2 SI for Chapter IV.6.2 – Individual Data of the Batches

Each foil was measured at two positions. The individual batches (Table SI IV-1), therefore, have the numbers 1.1, 1.2, 2.1, and 2.2. Note that the batches 1.1 and 1.2 had around 3.85 wt% particle content and batches 2.1 and 2.2 10 wt% particle content in dry foil.

Table SI IV-1. Mean values for all batches from 3D evaluation by AMIRA. Values rounded to 10 nm (voxel size).

sample	**Length [µm]**	**Breadth [µm]**	**Thickness [µm]**	**Mean AR$_1$**	**Mean AR$_2$**
Batch 1.1 0 %	1.52 ± 0.10	1.35 ± 0.03	1.29 ± 0.03	1.1 ± 0.1	1.2 ± 0.1
Batch 1.1 15 %	1.90 ± 0.13	1.26 ± 0.04	1.19 ± 0.03	1.5 ± 0.1	1.6 ± 0.1
Batch 1.1 50 %	2.65 ± 0.17	1.06 ± 0.04	0.98 ± 0.04	2.5 ± 0.2	2.7 ± 0.2
Batch 1.1 100 %	3.71 ± 0.33	0.90 ± 0.06	0.80 ± 0.05	4.2 ± 0.5	4.7 ± 0.5
Batch 1.2 0 %	1.54 ± 0.12	1.34 ± 0.03	1.29 ± 0.03	1.1 ± 0.1	1.2 ± 0.1
Batch 1.2 15 %	1.97 ± 0.14	1.23 ± 0.04	1.15 ± 0.03	1.6 ± 0.1	1.7 ± 0.1
Batch 1.2 50 %	2.77 ± 0.15	1.06 ± 0.05	0.98 ± 0.04	2.6 ± 0.1	2.8 ± 0.2
Batch 1.2 100 %	3.81 ± 0.23	0.93 ± 0.09	0.82 ± 0.04	4.1 ± 0.3	4.6 ± 0.4
Batch 2.1 0 %	1.39 ± 0.09	1.31 ± 0.03	1.27 ± 0.04	1.1 ± 0.1	1.1 ± 0.1
Batch 2.1 15 %	1.63 ± 0.08	1.22 ± 0.03	1.17 ± 0.03	1.3 ± 0.1	1.4 ± 0.1
Batch 2.1 50 %	2.22 ± 0.13	1.05 ± 0.04	0.95 ± 0.04	2.1 ± 0.2	2.4 ± 0.2
Batch 2.1 100 %	3.36 ± 0.20	0.91 ± 0.05	0.80 ± 0.04	3.7 ± 0.3	4.2 ± 0.4
Batch 2.2 0 %	1.56 ± 0.12	1.37 ± 0.04	1.31 ± 0.03	1.1 ± 0.1	1.2 ± 0.1
Batch 2.2 15 %	1.83 ± 0.09	1.26 ± 0.03	1.21 ± 0.03	1.6 ± 0.1	1.5 ± 0.1
Batch 2.2 50 %	2.41 ± 0.12	1.08 ± 0.04	0.98 ± 0.04	2.2 ± 0.2	2.5 ± 0.2
Batch 2.2 100 %	3.48 ± 0.22	0.95 ± 0.05	0.84 ± 0.05	3.7 ± 0.3	4.2 ± 0.4

IV.10.3 SI for Chapter IV.6.2.1 and IV.6.2.2 – Boxplots of Data

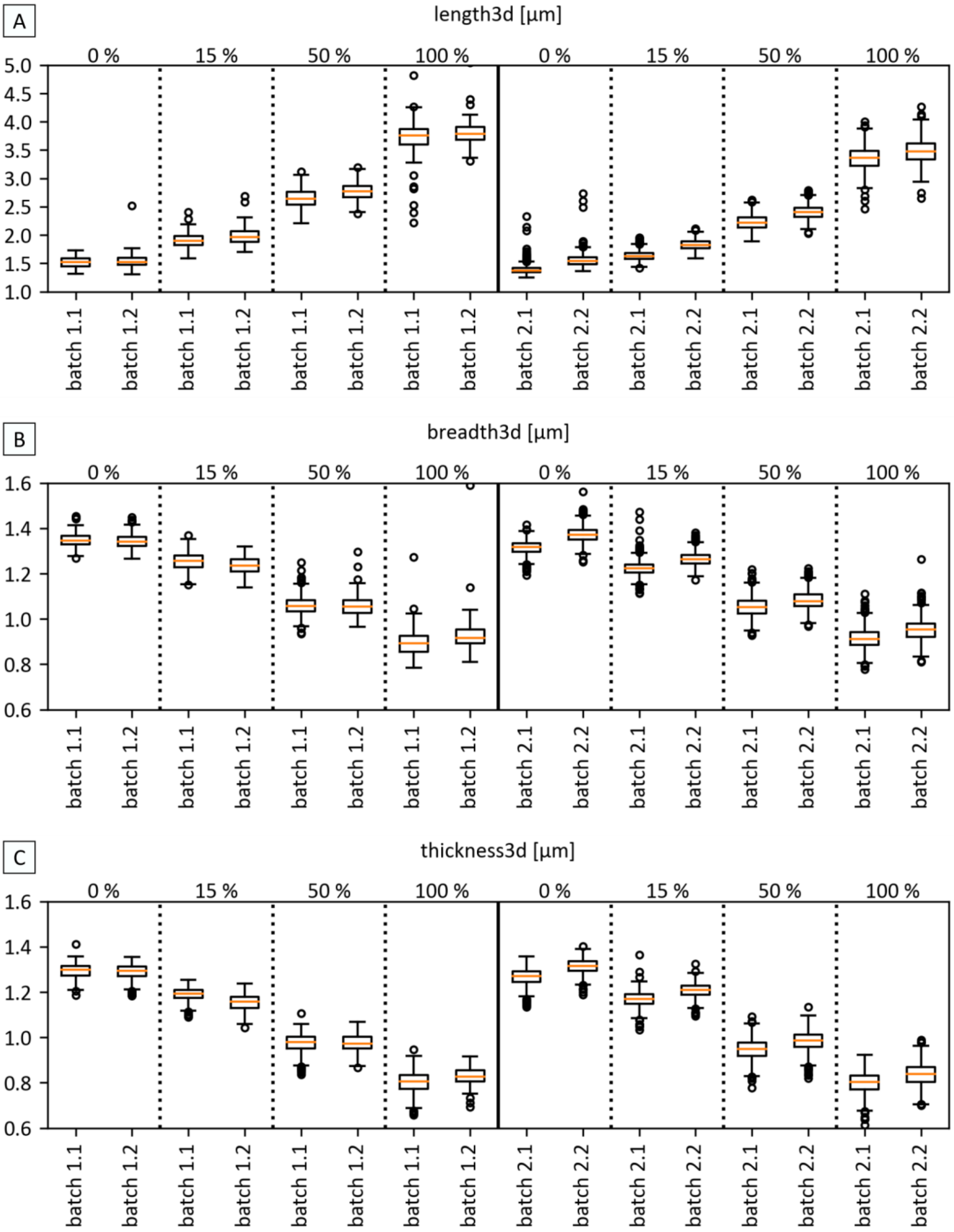


Figure SI IV-2. Boxplots of the different batches from AMIRA evaluations displaying parameters A: length3d, B: breadth3d, and C: thickness3d.

IV.10.4 SI for Chapter IV.6.2.3 – Individual Boxplots of ARs

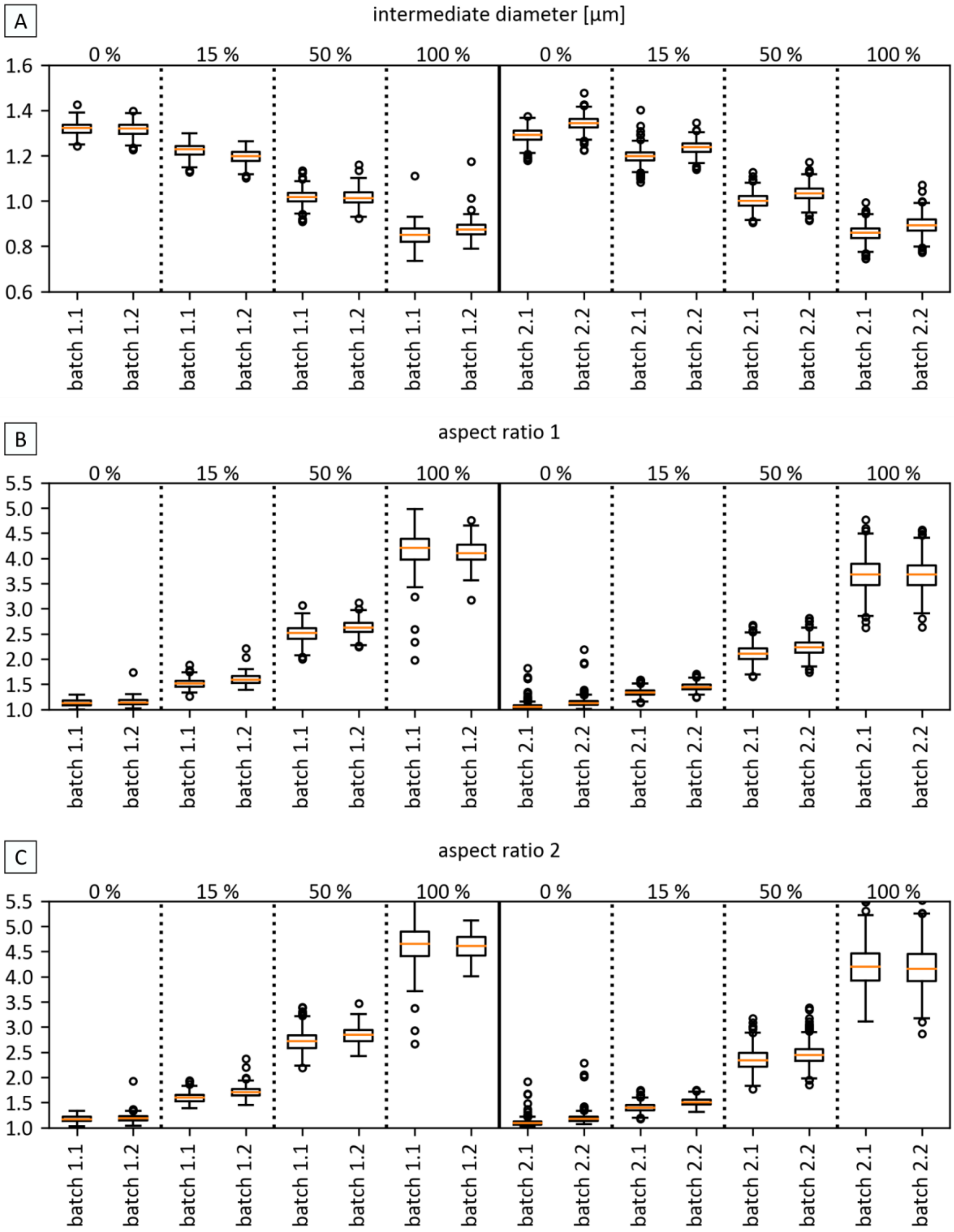


Figure SI IV-3. Boxplots of the different batches from AMIRA-evaluations displaying parameters A: intermediate diameter, B: AR_1, and C: AR_2.

IV.10.5 SI for Chapter IV.6.3 – Orientation Distribution

The boxplots for z-angle distribution:

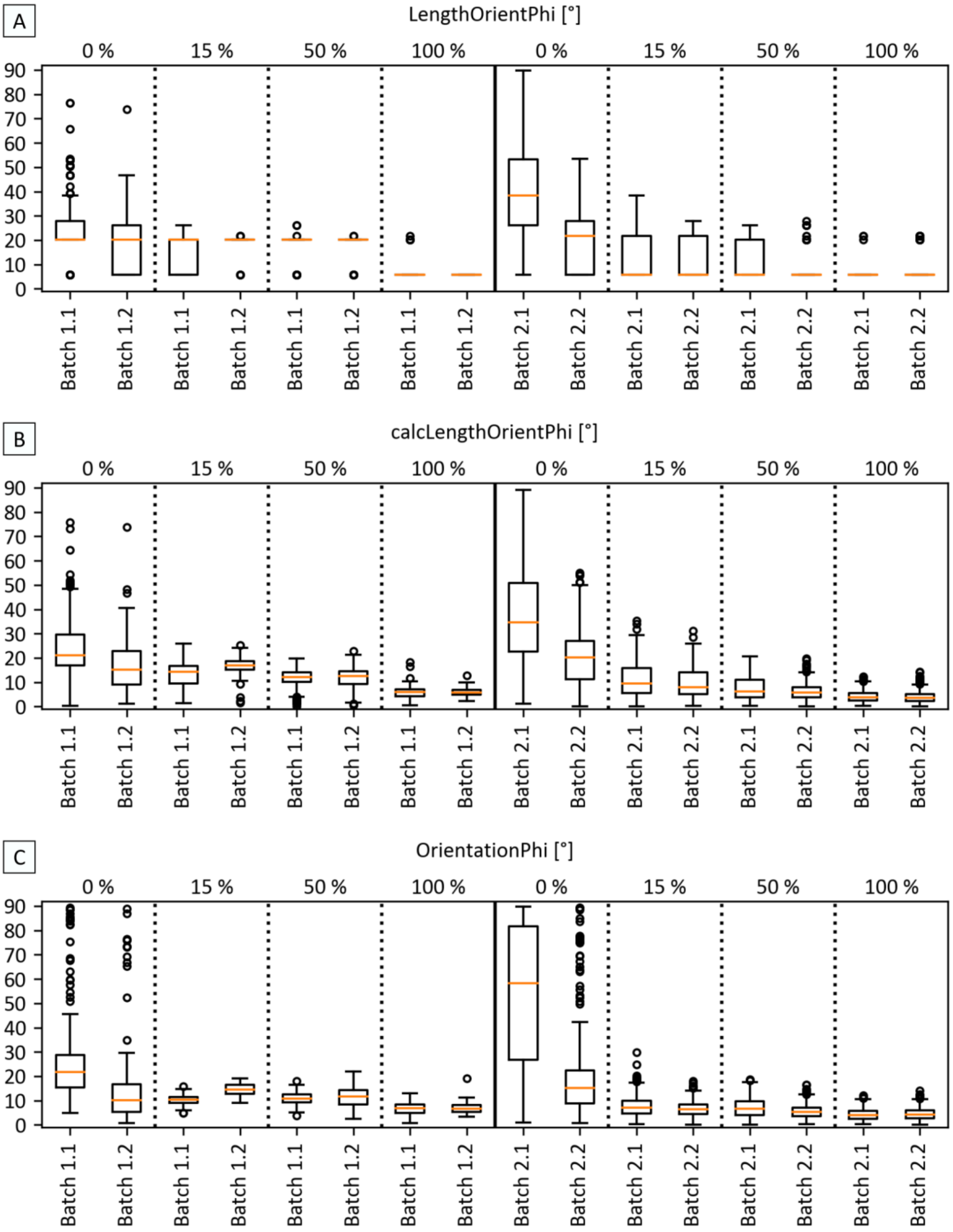


Figure SI IV-4. Boxplots of the different batches from AMIRA-evaluations displaying parameters A: LengthOrientPhi, B: self-calculated LengthOrientPhi, and C: OrientationPhi.

Batch 1 polar plots:

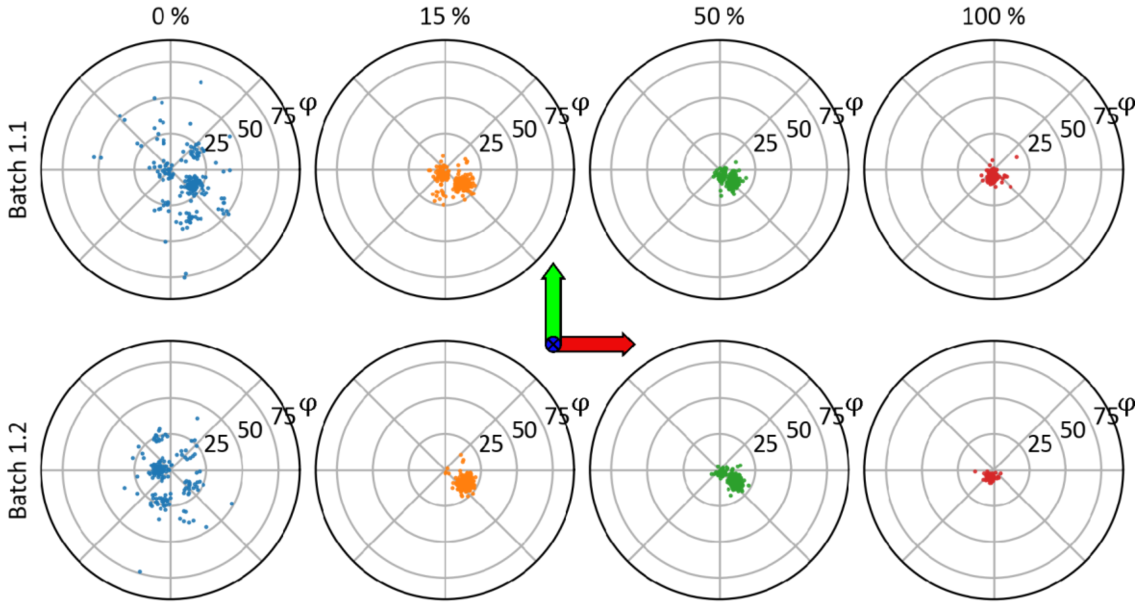


Figure SI IV-5. Batch 1. Polar plots of orientation of the particles from self-calculated angles (discrete values because of voxel), with theta as polar axis and phi as radial axis. Colors: blue: 0 %, orange: 15 %, green: 50 %, red: 100 %.

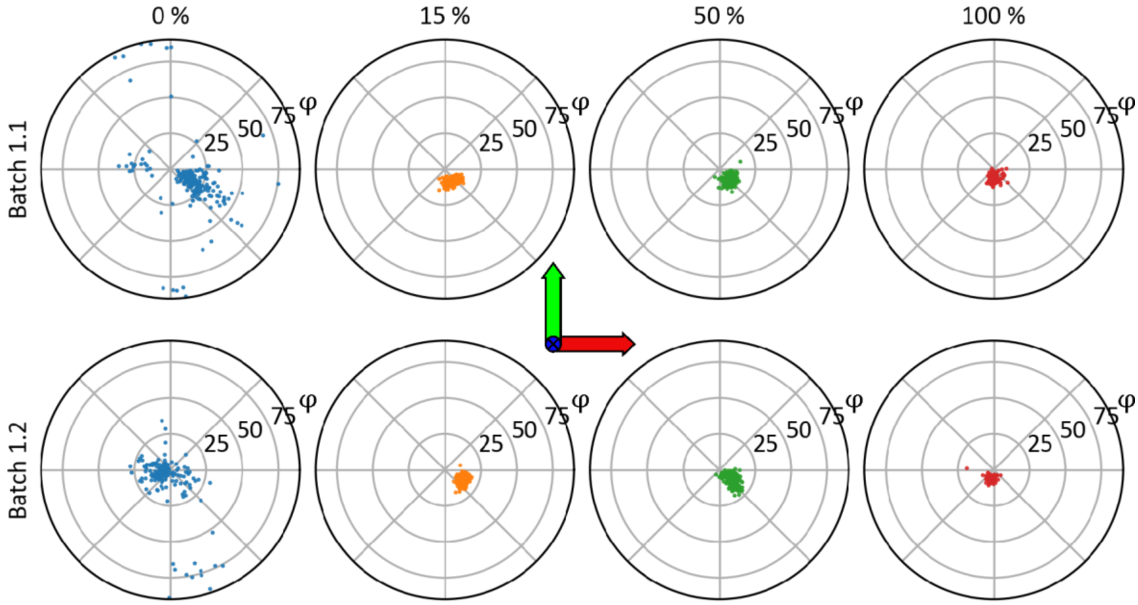


Figure SI IV-6. Batch 1. Polar plots of orientation of the particles calculated by AMIRA, with theta as polar and phi as radial axis. Colors: blue: 0 %, orange: 15 %, green: 50 %, red: 100 %.

Batch 2 polar plots:

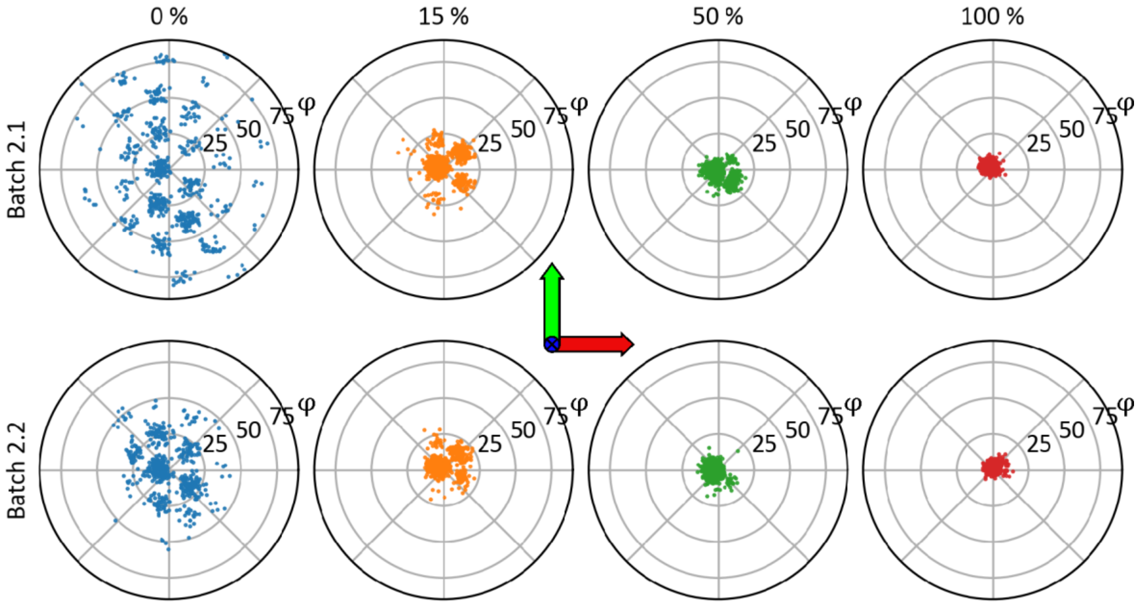


Figure SI IV-7. Batch 2. Polar plots of orientation of the particles from self-calculated angles (discrete values because of voxel), with theta as polar and phi as radial axis. Colors: blue: 0 %, orange: 15 %, green: 50 %, red: 100 %.

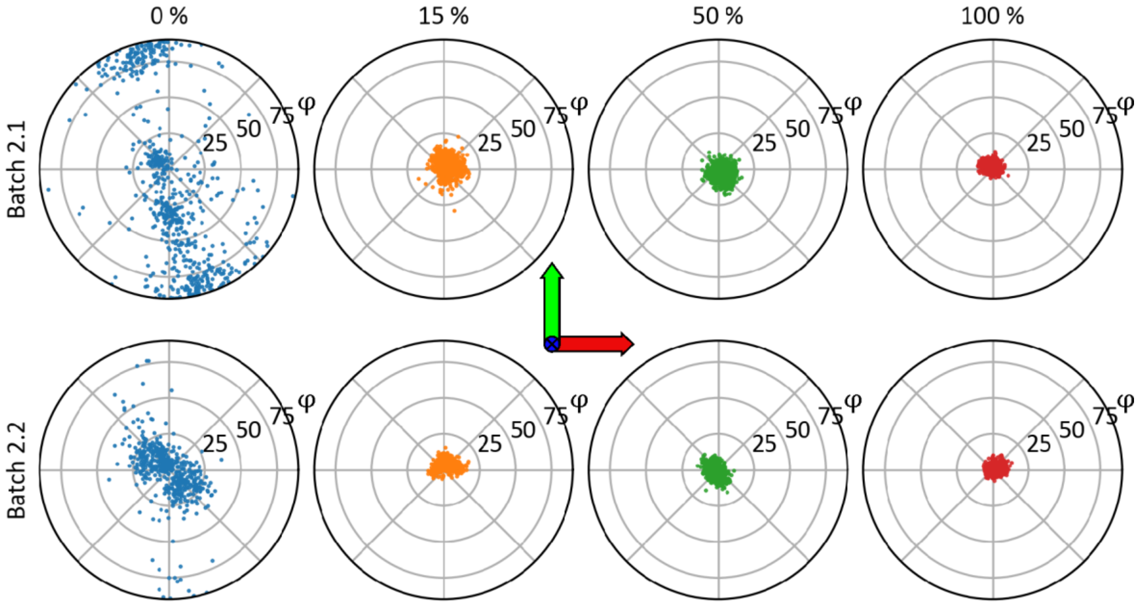


Figure SI IV-8. Batch 2. Polar plots of orientation of the particles calculated by AMIRA, with theta as polar and phi as radial axis. Colors: blue: 0 %, orange: 15 %, green: 50 %, red: 100 %.

IV.10.6 SI for Chapter IV.6.4 – Individual Boxplots of the AR_{waist}

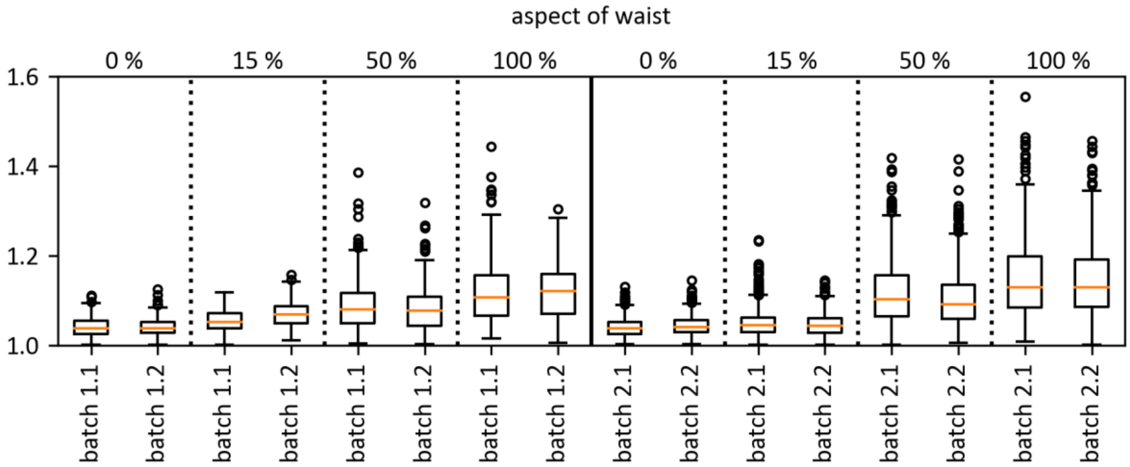


Figure SI IV-9. Boxplots of aspect ratio of particle waist (AR_{waist}) for individual samples.

IV.10.7 SI for Chapter IV.6.4.1 – Width of Small Particles

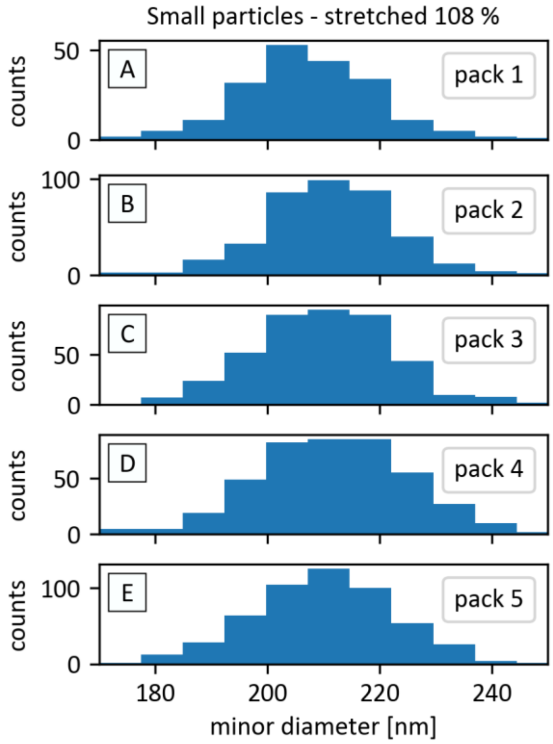


Figure SI IV-10. Histograms of the minor diameter of SEM image evaluation. Bin size (= pixel size): 7.443 nm

IV.10.8 SI for Chapter IV.6.4.2 – AR_{waist} in Different Regions

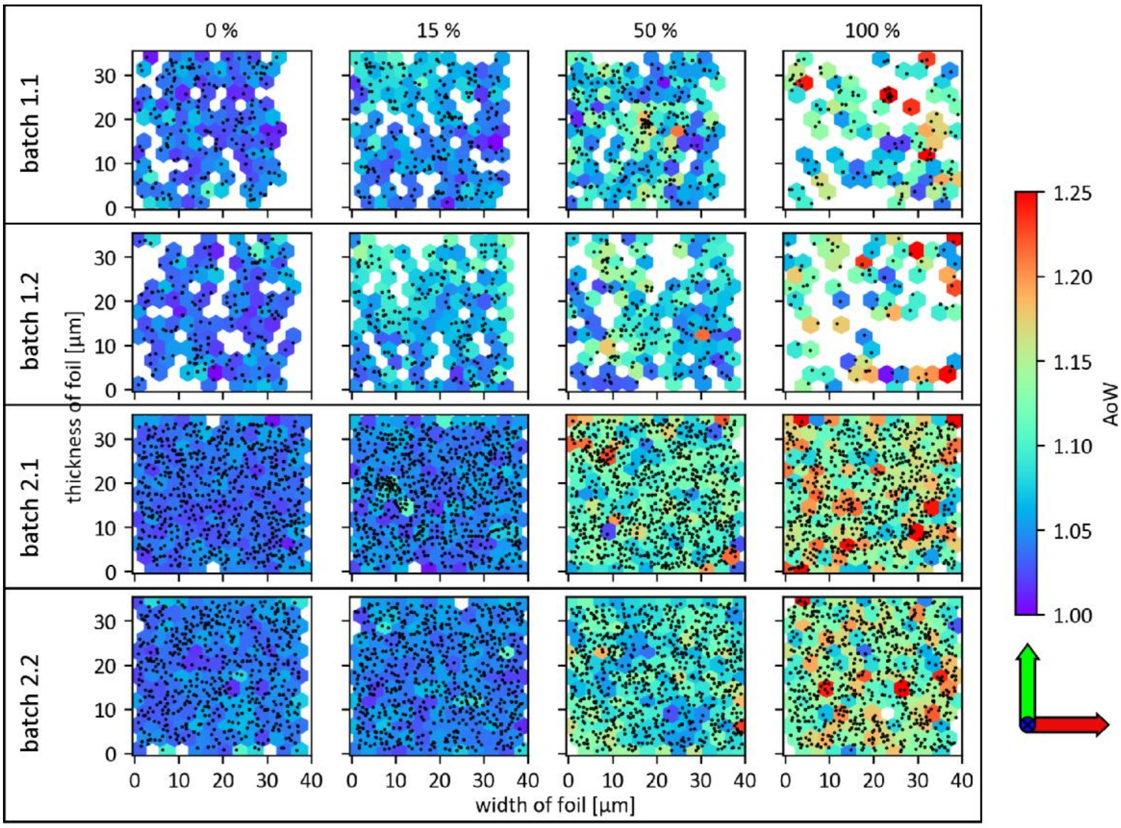


Figure SI IV-11. Distribution of AR_{waist} on xy-plane for the individual samples.

Plots of orientation of particles:

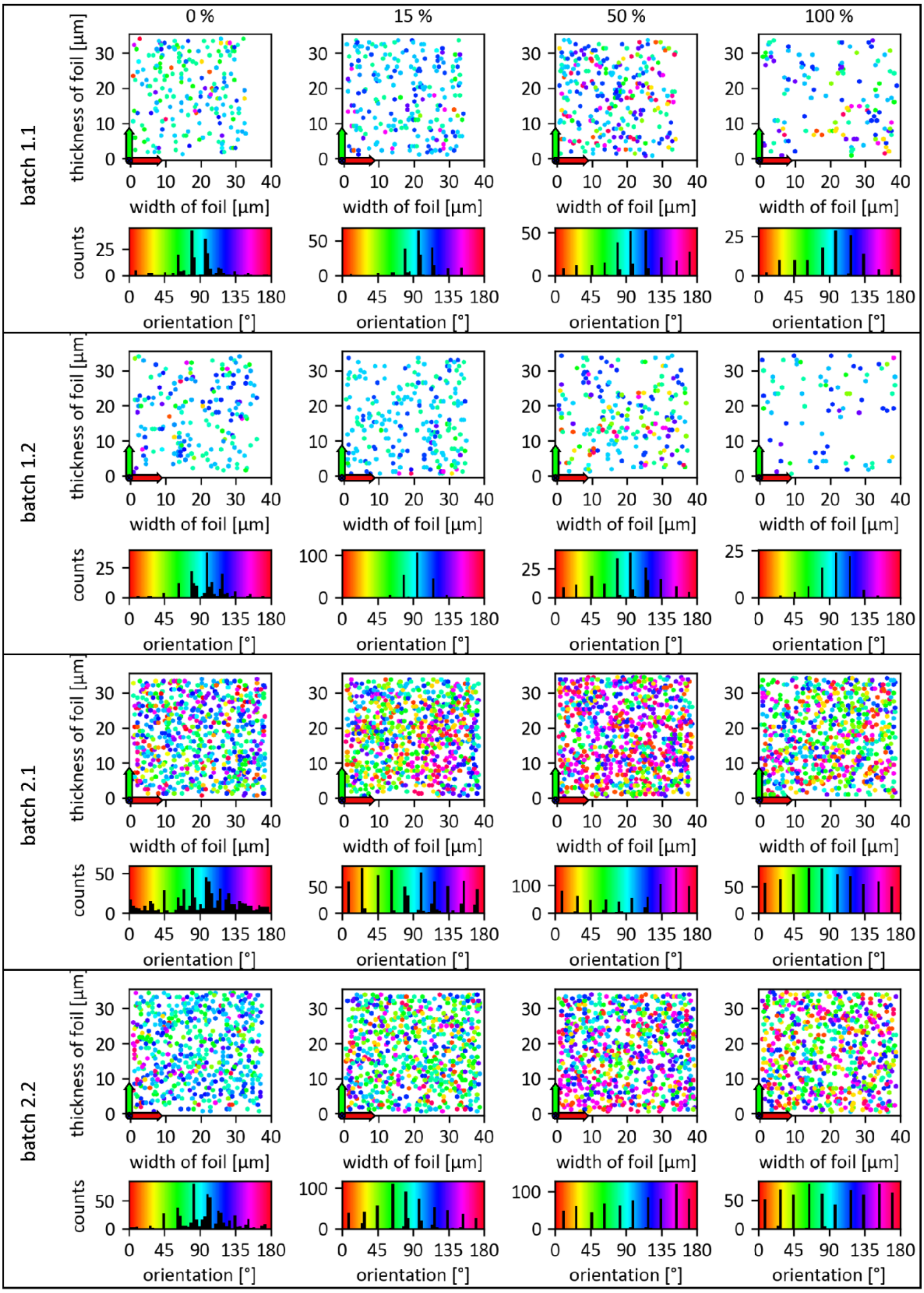


Figure SI IV-12. Orientation theta of breadth on xy-plane and histograms of angles with coloration.

IV.10.9 SI for Chapter IV.7 – Reconstruction by Orientation

The high-resolution “ortho slice“ of diagonal embedded foils does show vertical particles with highly distorted fiducials. As the fiducial cannot deform, this “alignment“ is wrong.

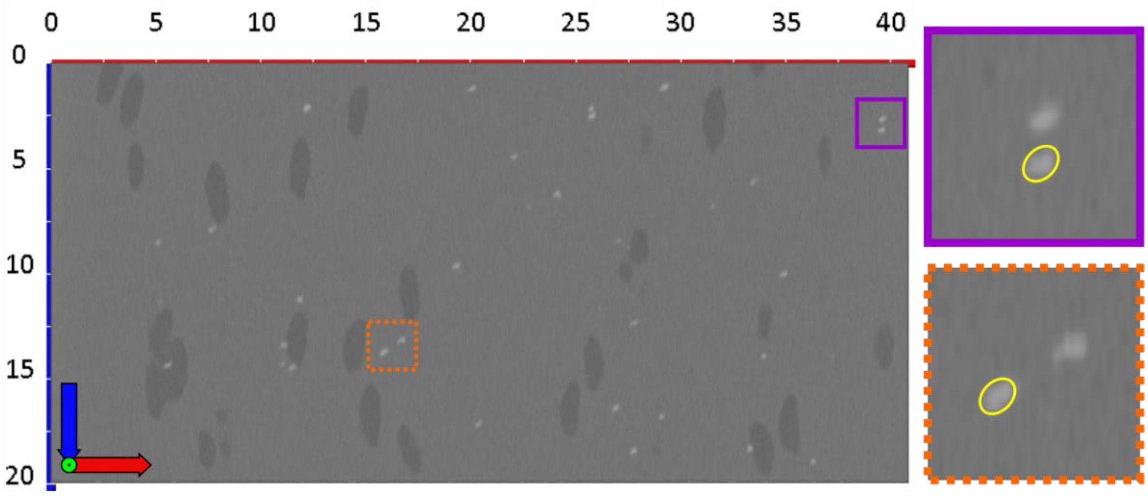


Figure SI IV-13. Ortho slice of the diagonal embedded foil prior alignment by fiducials. Scale in µm, zoom-in frames (purple and orange) 2.5 µm × 2.5 µm. Zoom-ins show the distortion of the fiducials (yellow ellipses).

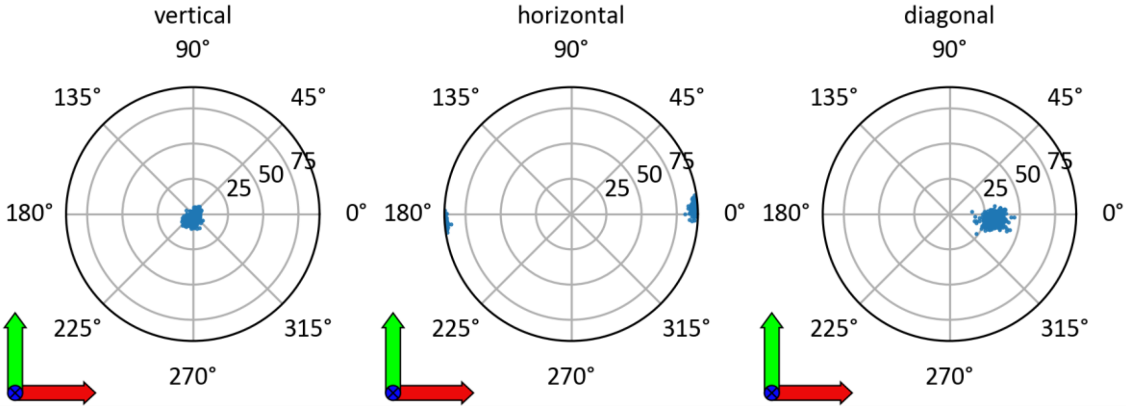


Figure SI IV-14. Polar plots of the corrected datasets with theta as polar axis and phi as radial axis.

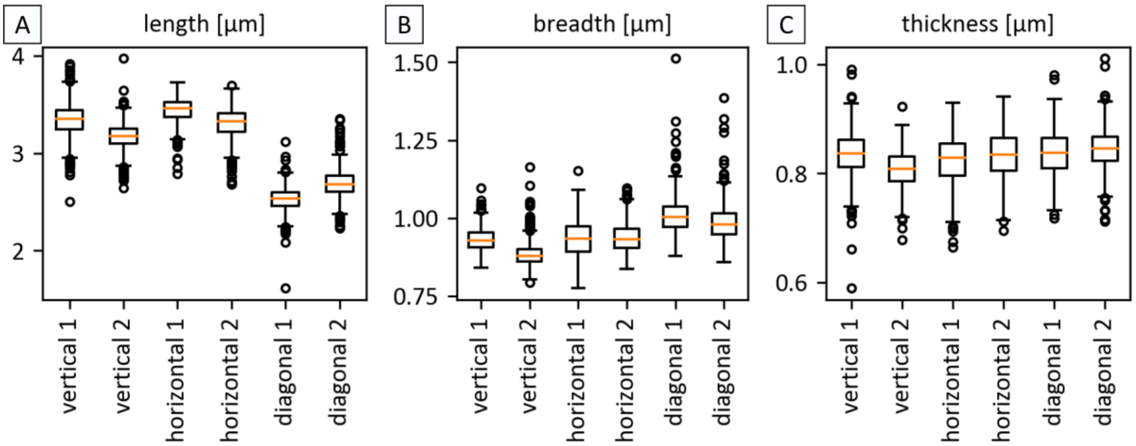


Figure SI IV-15. Individually corrected datasets of vertical, horizontal and diagonal foils.

IV.10.10 SI for Chapter IV.8 – Shape Deformation of Particles

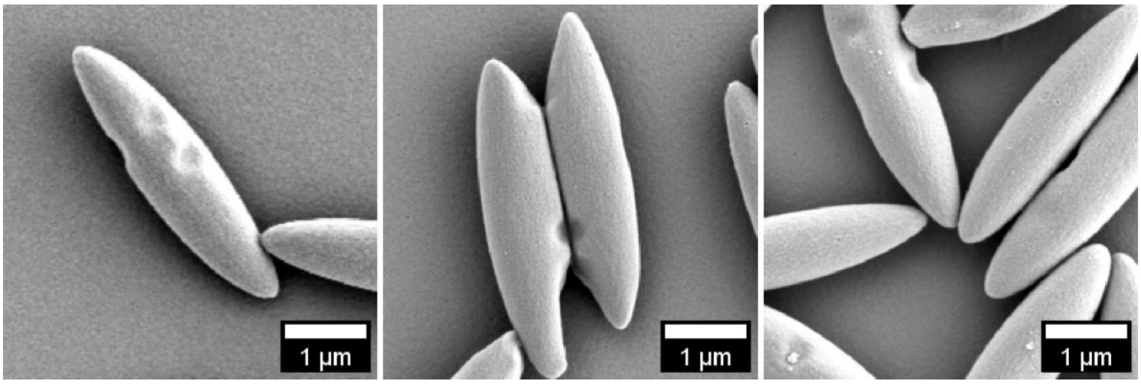


Figure SI IV-16. SEM images of recovered particles. The fiducial shape was imprinted into the particle shape during heating and stretching.

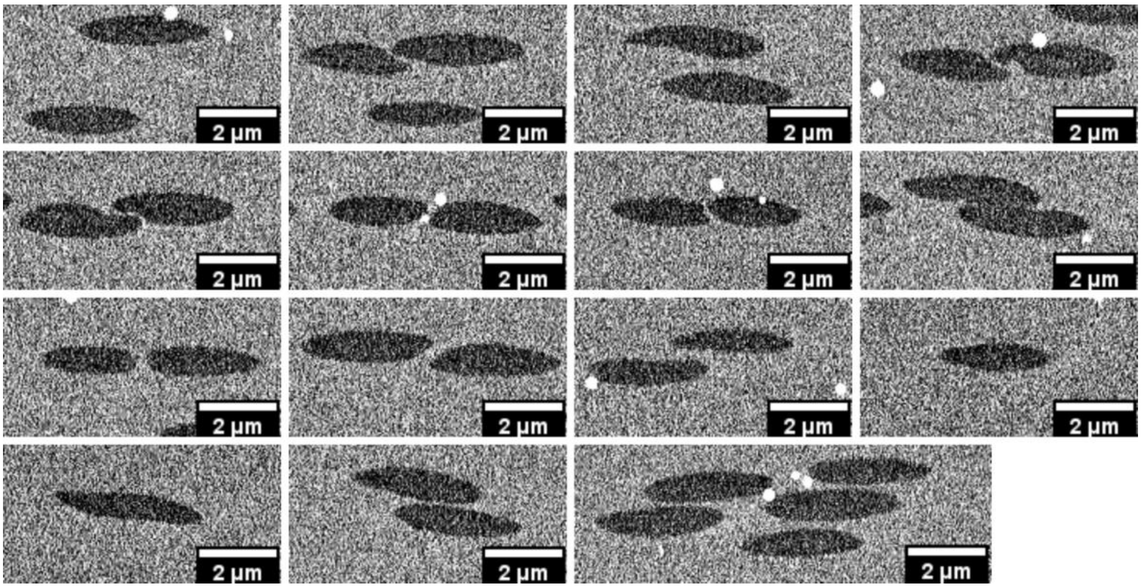


Figure SI IV-17. Further SEM images of particles in foil.

Chapter V. Particles in Monolayer

Micro- or nanometer-sized particles (colloids) can be assembled into monolayers using a variety of different methods (see next page). Monolayers with distinct repeating particle positions are crystalline, show *BRAGG* diffraction peaks (see *Introduction to solid state physics*, *Chapter 2* [112]) and can display color respective to the particle size and the viewing angle. Thus, a particle monolayer can be used as an optical component in applications like optical fibers [113] or wearable color-changing devices by bending [114].

Apart from using the particle monolayers as optical components, particle monolayers are often used in the so-called colloidal lithography. At colloidal lithography the particles of the monolayer are used as a mask either for etching a substrate [115] or for depositioning additional material in the holes in between the particles [10, 116-118]. Physical effects like plasmon-resonances were often studied on the resulting structures [119, 120]. In addition to close-packed (cp) particle monolayers, open-packed particle monolayers are interesting as well for colloidal lithography, e.g. for (transferable) hole-arrays [121, 122] and for anti-reflex coatings e.g. mimicking moth eyes [8].

Spherical, monodisperse, isotropic particles can easily be assembled in 2D hexagonal close-packed layers (hcp) displaying crystalline long-range orders. The most often used methods for hexagonal particle assemblies are directly from particle dispersions, with or without a spreading agent, by using the self-assembly mechanism which arises from the interplay of attractive capillary forces and repulsive electrostatic or steric forces (theory of particle assembly, see Chapter V.2.1). One of the main categories for exploiting 2D self-assembly is convective assemblies methods, resulting from drying dispersions, like e.g. simple drying

(2D coffee rings [123]), tilted-substrate-drying [124], spin-coating [125], drag-coating [14], dip-coating [126], confined convective assembly [127, 128]. Another main category is the self-assembly method on liquid surfaces like water [115, 129-132] or water–oil interfaces [133]. Large areas of hcp particle monolayers in quantities which are interesting for industrial uses can be produced [131, 132]. An overview of different methods can be gained in the paper of van Dommelen et al. [134]. The self-assembly methods on water surfaces of Retsch et al. [129] and Vogel et al. [130] will be subjects in the Chapter V.2. Apart from self-assembly, forced assembly, like rubbing particles on an elastomer substrate (subject of Chapter V.4), can also lead to large area hcp particle monolayers.

Particle monolayers with long-range orders can as well be non-close-packed or open-packed particle monolayers. One process of making such open-packed particle layers is by exploiting and controlling the repulsive forces (electrostatic or steric) between particles. Methods to use are e.g. self-assembly on fluid surfaces [135, 136], spin-coating of a high-viscose particle-in-monomer dispersion [137], or co-precipitation of the monomer TEOS and particles by dip-coating or convective self-assembly [138]. However, these assemblies led just to hexagonal open-packed particle crystal lattices. Another method to gain different open-packed particle lattices was investigated by Hummel et al. [139] exploiting particle-substrate interactions. They developed a method to transfer a close-packed monolayer into an open-packed monolayer by applying an one-dimensional lattice stretching vector during the pick up of the monolayer. Depending on the lattice orientation of the close-packed layer, the resulting open-packed particle layer shows different controllable lattice types. The method of Hummel et al. will be investigated in the Chapters V.1.2, V.3 and V.6.

Apart from controlling the particle-particle or particle-substrate interaction, a mechanically controlled method to gain open-packed monolayers was reported by Yan et al. in 2005 (partial subject of Chapter V.5, see also Koh et al. [140]). They used a process of picking up a silica hcp particle layer on an elastomer like PDMS and stretching or swelling the PDMS to get new particle positions with a followed transfer to a PVA layered substrate [141]. Further investigations of Li et al. cycled the process using calcination for setting the particles free from PVA and redo the process of pick up and transfer with PDMS [142]. With this method, they were able to control the lattice types and particle spacing in a very precise way.

While isotropic monodisperse spherical particles can order very easily, ellipsoidal particles are more difficult to arrange due to their anisotropic shape. Additionally, ellipsoidal particles show an anisotropic wetting behavior compared to spherical particles [70, 143]. Therefore, several researchers investigated the assembly behavior of ellipsoidal-like particles e.g. on interfaces [144, 145] or in 2D dispersions [105, 146].

Nevertheless, with the difficulties that come with the anisotropic nature, ellipsoidal particles were successfully assembled by using e.g. convective vertical assembly [147], *LANGMUIR-BLODGETT* troughs [148], and directed dewetting [90].

When assembling the particles in dense-packed monolayers, the anisotropic nature of the particles leads to anisotropic properties of the monolayer. Angular diffraction patterns show a difference due to anisotropic lattice spacings, which might be exploitable for optical components. Especially when the anisotropic particles can be controlled in their orientations, switchable optical devices are accessible [149].

Monolayers of ellipsoids, used as masks, could give access to hole-arrays with ellipsoidal pores. This might be beneficial for filter applications (see Chapter VI).

Structure of the chapter

The first sub-chapter (Chapter V.1) gives an overview of the used methods to describe monolayers. Therefore, it introduces to the theory of *BRAVAIS* and the access to different possible 2D lattices by a theoretic one-dimensional transfer of the hcp lattice in a defined direction. The motif of the transferred lattice is then substituted to ellipsoids. The main lattice configurations for ellipsoidal crystalline lattices are shown. The transfer to open-packed lattices is calculated. The ellipsoids, however, will still build a *BRAVAIS* lattice with their center of mass. Further, 2D test data of hcp particles are stretched one-dimensionally, leading to close-packed ellipsoid layer images. The particle correlation function ($g(r)$) and the angular correlation function ($\mathrm{g}_2^{\theta}(\mathrm{r})$) are introduced and the differences of the respective lattices are discussed. Additionally, the *FOURIER* transformation of crystalline particle layer images is explained.

The next sub-chapter (Chapter V.2) starts with the theory of floating particle monolayer assemblies. Two floating particle self-assembly methods are described. The methods are deployed on different stretched ellipsoidal particles (different size ranges and ARs). The experimental data reveal the different lattices found on dense packed ellipsoidal particle monolayers.

After the assembly into floating monolayers, particles are mostly transferred to dry substrates giving additional possibilities to alter the respective lattices. Thus, the earlier mentioned method of Hummel et al., to transfer an hcp monolayer by a 1D transfer vector to different *BRAVAIS* lattice types [139], is implied on spherical and ellipsoidal monolayers in the next sub-chapter (Chapter V.3). The particle positions are discussed in means of their $g(r)$ and $\mathrm{g}_2^{\theta}(\mathrm{r})$.

The hcp lattices can also be used to make anisotropic particle monolayers as Lele et al. describes [150]. They used the previous discussed particle stretching method (Chapter II.2.1.2) directly onto 2D hcp layers after annealing those 2D layers. Due to the annealing before the stretching, the particles show a deformed shape and melt into a plate. Therefore, they cannot be regarded as individual particles anymore. Another possibility to come to highly-ordered ellipsoidal 2D monolayers of individual particles can be applying the rubbing method of Park et al. [151] onto dried ellipsoidal particles. The potential of this approach is shown in the following sub-chapter (Chapter V.4).

Additionally, a proof-of-concept study of combining the rubbing method to form closely packed particle layers and their transfer to a water surface, can be found in Chapter V.4.

In Chapter V.5, the rubbed cp monolayers of spheres and ellipsoids are opened up by stretching the elastomeric substrate. It is shown that the necking behavior of the elastomeric substrate needs to be taken into account for the resulting lattice. Square and hexagonal lattices can be achieved with different origin lattice orientations.

A short study for creating open-packed highly-ordered ellipsoidal-like particle monolayers is depicted in Chapter V.6. The study combines the methods of rubbing, lattice transfer, and stretching. Some interesting-looking crystalline lattices with retained particle positions are shown and their evolution is explained.

Although the Chapters V.2 – V.5 have own sub-chapter summaries due to their size, the particle monolayer chapter closes with a comprehensive summary (Chapter V.7).

V.1. Particle Monolayer Description

Monolayers of particles can be described by the particle position and, if the particles are anisotropic e.g. ellipsoidal, by their orientation within the ensemble. Additionally, the size of domains of different orientations might be interesting.

For the position description, the pair correlation function can be calculated using the center of the respective particle on the microscopy image (center positions by python program of Chapter III.3). Additionally, the *FOURIER* transformation of an image (FFT by Fiji/ImageJ) delivers distinct peaks in the case of ordered layers. Such peaks can be observed as well by using laser diffraction. However, laser diffraction is averaging over a large area of the sample due to the laser spot size, including multiple particle monolayer domains.

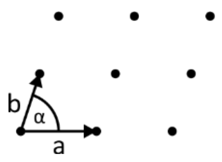
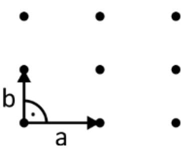
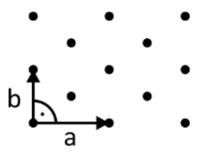
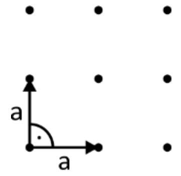
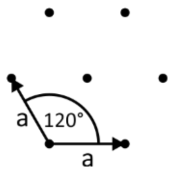
The particle orientation within the monolayer was evaluated using a python program. Histograms showed the main orientations on the microscopy image. The evaluation of one zoomed-out image with > 300 particles should already give suitable statistics. However, as care was taken to take the images of one sample from the same direction, an overall orientation of the particles in the monolayers was accessible by using multiple images on different regions of the sample. Additionally, an angular dependent distribution function gave the possibility to discuss short- and long-range orientations within the monolayers.

Apart from the description of individual lattice types, one has to take into account that there are different domains of position (lattice) or orientation in the same sample. The identification of domain size was not always possible as most ellipsoidal monolayers exhibited smooth domain transitions. The best way to identify different domains was to use the orientation colorization of the python program.

V.1.1 Theoretical Description of Lattice Positions

All possible regular lattices can be described by *BRAVAIS*. The theory is basic and found in solid state text books (see *Introduction to solid state physics*, *Chapter 1* [112]). A 2D periodic lattice is spanned by 2 translation vectors and the angle between them. In 2D, there are five possible *BRAVAIS* lattices (Table V-1). The five lattice types are grouped in four crystal families with different symmetries (point groups, noted by *SCHOENFLIES* notation).

Table V-1. Table of the five different BRAVAIS lattices with their crystal family and point group. The letters and lower numbers give the amount of C: proper rotation axes (cyclic) and D: additional twofold axes (dihedral).

	oblique	**rectangular**	**centered-rectangular**	**square**	**hexagonal**
lattice					
vectors	$a \neq b$	$a \neq b$		$a = b$	$a = b$
angle	$\alpha \neq 90°$	$\alpha = 90°$		$\alpha = 90°$	$\alpha = 120°$
crystal family	monoclinic	orthorhombic		tetragonal	hexagonal
point group	C_2	D_2		D_4	D_6

The base or motif of a lattice can be of different shape. Therefore, it does not matter if the motif is a spherical particle or an ellipsoid. *BRAVAIS* lattices can be transposed by 1D transfer of the lattice vectors under a certain angle. Square and hexagonal lattices are as well centered-rectangular lattices, but with a higher degree of symmetry.

V.1.2 Transferring an HCP Lattice – Phase Diagram

Spherical particles exhibit a hexagonal lattice when close-packed and are often noted as hcp (hexagonal close-packed). The point group is D_6, with a 6-fold symmetry. The other *BRAVAIS* lattices are accessible by transfer with vector $\vec{T}$ under a defined transfer angle. Hummel et al. [139] investigated this phenomenon using an hcp starting lattice where the particle contact points were parallel to y-axis and looked at the angle β_{Hummel} enclosed by the transfer/stretching vector $\vec{S}$ and the x-axis (Figure V-1, $\beta_{Hummel} = 90° \ - \alpha_{in\ Figure}$). For their purpose, they implemented an additional possible turning angle δ, which would give the possibility to leave the orientation of the starting hcp lattice unknown, as δ turns the lattice to the position of the particle contact towards y-axis. The lattice vectors $\vec{a}$ and $\vec{b}$ result in vectors $\vec{a}'$ and $\vec{b}'$ after stretching.

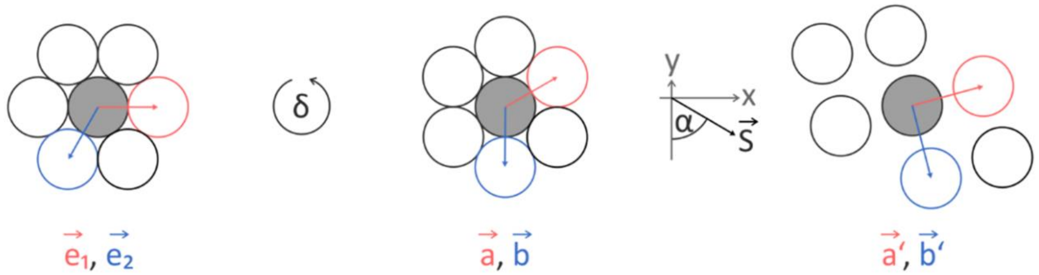


Figure V-1. Hexagonal close-packed (hcp) particles transferred to open-packed particles by lattice transfer. A randomly oriented hcp layer is first rotated with δ to the starting lattice with particle contact in y-direction. Then the lattice is stretched by $\vec{S}$. The angle $\beta_{Hummel} = 90° - \alpha_{in\ Figure}$. Adapted with permission from Hummel et al. [139]. Copyright © 2019 American Chemical Society.

By plotting the angle β_{Hummel} against the transfer/stretching factor S, they got a phase diagram of the different resulting *BRAVAIS* lattices (Figure V-2).

They marked different possible lattices within their diagram. Unfortunately, they forgot to mention that at 0° the obtained crystal lattice will always be centered rectangular (D_2) with vector $\vec{b}'$ resembling the distance of direct contacting particles along the y-axis ($\vec{b}' = \vec{b}$). From 30° to 60°, the phase diagram repeats itself in a mirrored way, and the whole diagram repeats in a 60° fashion.

Hummel et al. [139] published the following equation (Equation V.1-1) to calculate the transfer factor S origin from the hcp lattice of spheres when $\vec{a}' \perp \vec{b}'$ (plot in Figure V-2).

$$S = \frac{1}{\sqrt{\frac{\sin\beta_{Hummel}\cdot(\sqrt{3}-\tan\beta_{Hummel})}{\sqrt{3}\cdot\sin\beta_{Hummel}+\cos\beta_{Hummel}}}} = \sqrt{\frac{\sqrt{3}\cdot\sin\beta_{Hummel}+\cos\beta_{Hummel}}{\sin\beta_{Hummel}\cdot(\sqrt{3}-\tan\beta_{Hummel})}} \qquad \text{(Equation V.1-1)}$$

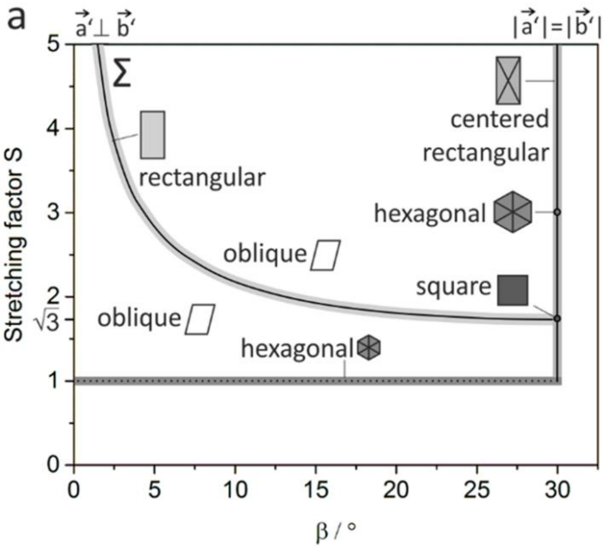


Figure V-2. Phase diagram of the transfer. The phase diagram misses the centered rectangular lattice at 0° for all stretching factors S. Additionally, the phase diagram should be mirrored at 30° and then it repeats itself every 60°. Adapted with permission from Hummel et al. [139]. Copyright © 2019 American Chemical Society.

For better usage, the angular description in this thesis is changed to a starting lattice of particles touching each other in x-axis direction (not in y-axis as with Hummel et al. [139]). The angle (α) is the same as depicted above but is now enclosed by x-axis and the stretching vector (Figure V-3). The above-mentioned phase diagram shifts by 30° ($\alpha = 90° - \beta_{Hummel}$) when plotting α. In this thesis, the transfer/stretching vector $\vec{S}$ and transfer/stretching factor S are named transfer vector $\vec{T}$ and transfer factor T to avoid confusion with the stretching method of particle manufacturing.

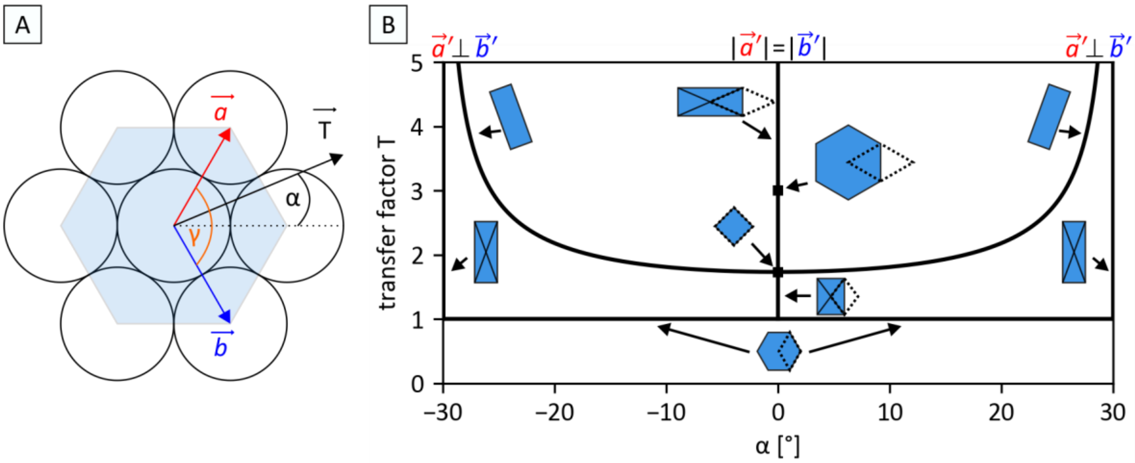


Figure V-3. A: Sketch of hcp particles with lattice vectors $\vec{a}$ (red) and $\vec{b}$ (blue), transfer vector $\vec{T}$, transfer angle α, and intrinsic angle γ (orange). B: Phase diagram of transfer factor T to angle α for hcp as starting lattice. New vectors $\vec{a}'$ and $\vec{b}'$ (after transfer) are orthogonal at curved plot ($\vec{a}' \perp \vec{b}'$) and have the same length on the vertical line at 0° ($|\vec{a}'|=|\vec{b}'|$). Dashed lines show the transformation of the rhombus spanned by $\vec{a}$ and $\vec{b}$. Diagram repeats at ± 30°.

A more general approach with an easier equation depicting the same solution of Equation V.1-1 will be given in Chapter V.1.4.2.

V.1.2.1 Square Lattices

Every 60° (starting from 30° at Hummel et al.), it is possible to gain access to a squared lattice (D_4) by transfer with the factor $T = \sqrt{3} = 1.73$ directly in the direction of the next particle contact (α = 0° or β_{Hummel} = 30°). The intrinsic angle γ becomes 90°. The vector sizes for the resulting square lattice are $\vec{a}' = \vec{b}' = r \cdot \sqrt{6}$ with r as the radius of the particles.

V.1.2.2 Hexagonal Lattices

Further transfer in the direction of the next particle contact gives access to another extraordinary point in the phase diagram. Every 60°, it is possible to gain access to an open-packed hexagonal lattice (D_6) by a transfer with factor $T = 3$. The intrinsic angle γ becomes 60° and the distance of the particle centers grow to $\vec{a}' = \vec{b}' = r \cdot \sqrt{12}$.

V.1.2.3 Rectangular Lattices – Primitive and Centered

With the intrinsic angle γ of 90°, a defined possibility to get rectangular shaped lattices can be plotted (see Equation V.1-1). The equation for the transfer factor T has limits at 0° and 60°. At those angles the possible primitive-rectangular lattice becomes a centered-rectangular lattice. Other possibilities for centered-rectangular lattices are transferring the lattice at 0° (containing the square and the open-hexagonal lattice points). The lattice vectors depend on the transfer parameters.

V.1.2.4 Oblique Lattices

Oblique lattices are found at the other transfer parameters.

V.1.3 Close-Packed Oriented Ellipsoidal Layers

While the theory of how to generate multiple *BRAVAIS* lattices was now implemented for 1D transfer of spherical hcp to open-packed lattices, the same can be applied to gain close-packed lattices of oriented ellipsoids.

To do so, one starts again with an hcp lattice of spherical particles, but then changes the motif itself in regards to the transfer vector. Thus, the underlying lattice got the same transfer as above (same lattice phase diagram), but the shape of the motif changes from a sphere to an ellipsoid. As the stretching is one-dimensionally, the radius in the direction perpendicular to the transfer direction will stay the same.

$$r_{long} = T \cdot r_{original} \qquad \text{(Equation V.1-2)}$$

$$r_{short} = r_{original} \qquad \text{(Equation V.1-3)}$$

The resulting lattices with the ellipsoids are sketched in Figure V-4.

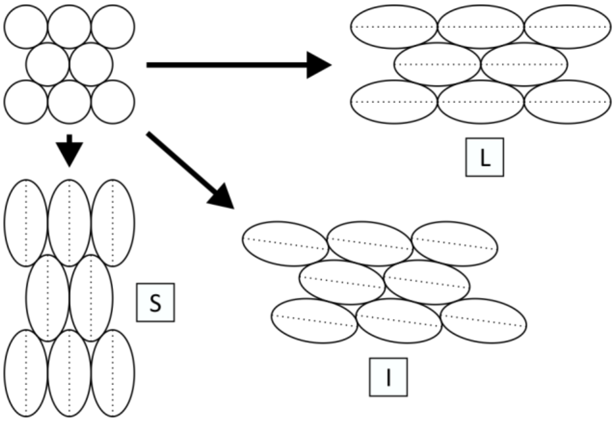


Figure V-4. Sketch of possible close-packed lattices of ellipses. L and S configuration have centered-rectangular lattices, while intermediate configuration I shows oblique or primitive rectangular lattices. Dotted lines depict the long axes of the ellipses.

Close-packed lattices are generated with short axis particle contact (S configuration, “transfer“ of hcp at ± 30°, repeating at ± $n \cdot 60°$), long axis contact (L configuration, “transfer“ of hcp at 0° ± $n \cdot 60°$), and intermediate configurations (I-configuration, oblique or primitive rectangular lattices).

For the mathematical description, L and S configuration will be discussed in the following (needed for Chapter V.4.4.2).

V.1.3.1 Description of L and S Configuration in Ellipsoidal CP Layers

Due to the centered-rectangular lattice, the distance between particle centers is half the rectangle diagonal (Figure V-5). The distance from the ellipsoid centers D can, therefore, be calculated by Equation V.1-4.

$$D = \frac{\sqrt{X^2+Y^2}}{2} \qquad \text{(Equation V.1-4)}$$

Furthermore, X and Y must meet the criteria of the following equation (Equation V.1-5) of two ellipsoids being in contact:

$$\frac{X^2}{(r_x \cdot 4)^2} + \frac{Y^2}{\left(r_y \cdot 4\right)^2} = 1 \qquad \text{(Equation V.1-5)}$$

The radius r_x is the center ellipse radius in x-direction while r_y is the radius in y-direction. X is the dimension of the particle center distance in x-direction, Y in y-direction. For the following two lattices, the center distance and the angle β to x-axis can be calculated by basic math.

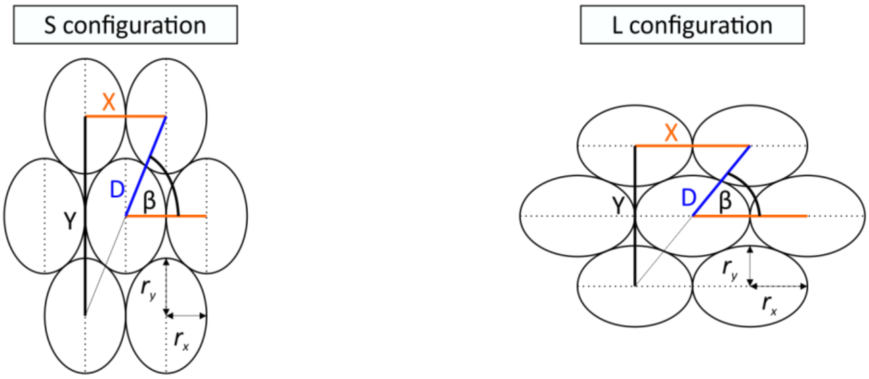


Figure V-5. Sketches of S and L configuration of close-packed ellipses with AR 1.42. The long axes are marked by dashed lines. Note that r_x and r_y are not the same ellipse axes, but interchangeable in the two configurations.

The short axis of the ellipsoid is s (short axis radius) while the long axis is l (long axis radius). As the two configurations just have different orientations for the axes by 90°, r_x or r_y are interchangeable short or long axis. The equations containing the AR are, therefore, as well interchangeable by inversing the AR.

Table V-2. Equations to describe the ellipsoids cp monolayers with short axis (S) and long axis (L) configuration.

S configuration	**L configuration**
With $r_x = s, r_y = l$ and $X = 2s$ $Y = 2l\sqrt{3}$ Diagonal particle center distance: $D = \frac{\sqrt{(2s)^2 + 12l^2}}{2} = \sqrt{s^2 + 3l^2}$ Using the AR: $AR = l/s$ $D = s\sqrt{1 + 3AR^2}$ The particle contact angle can be calculated to $\beta = \cos^{-1}\left(\frac{X/2}{D}\right) = \cos^{-1}\left(\frac{1}{\sqrt{1 + 3AR^2}}\right)$	With $r_x = l, r_y = s$ and $X = 2l$ $Y = 2s\sqrt{3}$ Diagonal particle center distance: $D = \frac{\sqrt{(2l)^2 + 12s^2}}{2} = \sqrt{l^2 + 3s^2}$ Again using the AR: $AR = l/s$ $D = s\sqrt{3 + AR^2}$ The particle contact angle is then calculated to $\beta = \sin^{-1}\left(\frac{Y/2}{D}\right) = \sin^{-1}\left(\frac{\sqrt{3}}{\sqrt{3 + AR^2}}\right)$

V.1.4 Transfer Diagram to Open-Packed Ellipsoidal Monolayer

The close-packed ellipsoidal particle layer can be transformed to an open-packed ellipsoidal monolayer. As the starting lattice of the monolayer does not have the same properties as an hcp particle monolayer (no six fold symmetry), the phase diagram for transfer needs to be adjusted.

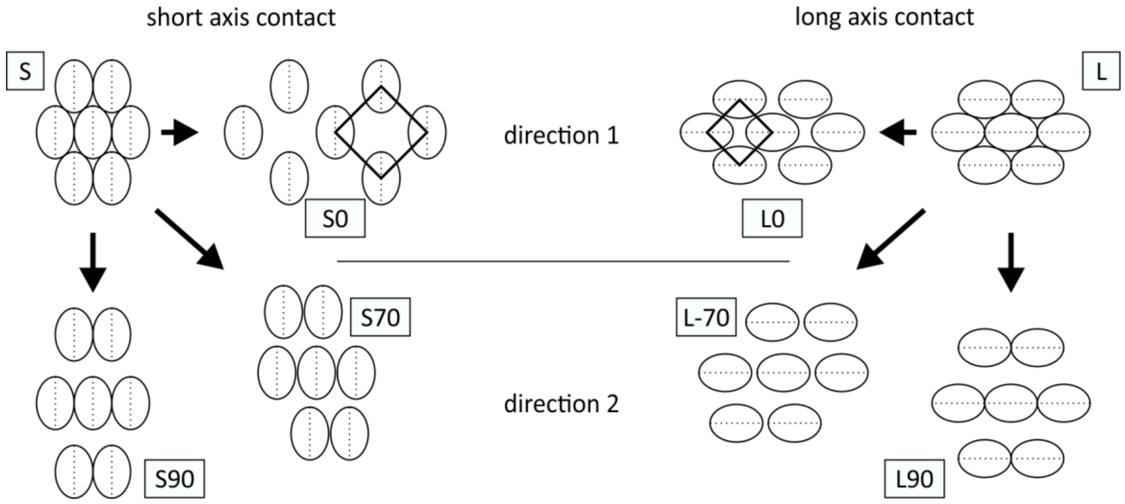


Figure V-6. Sketch of lattice transfer. The D_6 symmetry of hcp became D_2 after the deformation to ellipsoids. Therefore, the transfer has to be divided into two general directions for the description. Examples for the two directions were α= 0° for direction 1, and 70° (or -70°) and 90° for direction 2 (angle indicated with number after letter). Black squares at 0° show possible square lattices. Note that S0 and L0 have same particle orientation at square lattice, but S0 has larger lattice vectors and, therefore, a more open lattice.

From Table IV-2, the particle center distances of the S and L configurations are known. Analogous to Chapter V.1.2, the start lattice can be spanned open by lattice vectors $\vec{a}$ and $\vec{b}$. The angle between those vectors should be γ. The transfer vector $\vec{T}$ has a transfer angle α and transfer factor T. The projection of $\vec{a}$ and $\vec{b}$ on $\vec{T}$ enclose angles σ and ε.

For the stretched close-packed layers S and L, the symmetry degenerated to D_2. The lattice repeats just every 180° with a mirrored repetition every 90°. To render the phase diagram of T and α, two different directions need to be investigated.

The two vectors $\vec{a}$ and $\vec{b}$ with the same length D span the lattices of direction 1, while the two vectors $\vec{c}$ and $\vec{d}$ with length D and X span the lattices of direction 2. The angle β is the angle between the spanning vector $\vec{a}$ and the x-axis. The angle where the two directions meet is defined by 90° to the respective spanning vectors.

Direction 1 (Chapter V.1.4.1): $0° \leq \alpha \leq 90° - \beta$

Direction 2 (Chapter V.1.4.3): $90° - \beta \leq \alpha \leq 90°$

The combined transfer diagram of the two directions can be found in Chapter V.1.4.4.

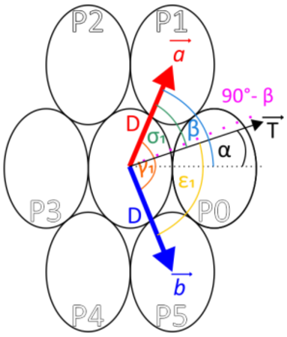
V.1.4.1 Transfer in Direction 1

The lattices of direction 1 are spanned by vectors of equal length ($|\vec{a}| = |\vec{b}| = D$), thus, the equations will describe the hcp lattice transfer as well.

The Equation V.1-6 for direction 1 is valid at $0° \leq \alpha \leq 90° - \beta$.

The projection of $\vec{a}$ (to P1) on the transfer vector encloses angle σ_1 while $\vec{b}$ (to P5) encloses angle ε_1.

$\sigma_1 = \beta - \alpha$ and $\varepsilon_1 = \beta + \alpha$



The projections are vectors $\overrightarrow{S_{Px}}$:

$\overrightarrow{S_{P1}} = \cos\sigma_1 \cdot D$ and $\overrightarrow{S_{P5}} = \cos\varepsilon_1 \cdot D$

The vectors orthogonal to the projection are $\overrightarrow{H_{Px}}$.

$\overrightarrow{H_{P1}} = \sin\sigma_1 \cdot D$ and $\overrightarrow{H_{P5}} = \sin\varepsilon_1 \cdot D$

Per definition, the orthogonal vector will retain its length while the projection will be transferred by the transfer factor T. Note that cot is used, as H ($H > 0$) is then below the fraction bar.

$$\overrightarrow{H_{Pxnew}} = \overrightarrow{H_{Px}} \quad \text{and} \quad \overrightarrow{S_{Pxnew}} = T \cdot \overrightarrow{S_{Px}}$$

$$\sigma_{1,new} = \cot^{-1}\frac{\overrightarrow{S_{P1new}}}{\overrightarrow{H_{P1new}}} = \cot^{-1}(T \cdot \cot\sigma_1)$$

$$\varepsilon_{1,new} = \cot^{-1}\frac{\overrightarrow{S_{P5new}}}{\overrightarrow{H_{P5new}}} = \cot^{-1}(T \cdot \cot\varepsilon_1)$$

With the opening angle γ it is:

$$\sigma_{1,new} + \varepsilon_{1,new} = \gamma_1$$

Using sum theorem of $\cot^{-1}$:

$$\cot^{-1}x + \cot^{-1}y = \cot^{-1}\frac{xy-1}{x+y} \qquad , \text{for } x + y > 0$$

$$\text{and } \cot^{-1}x + \cot^{-1}y = \pi + \cot^{-1}\frac{xy-1}{x+y} \quad , \text{for } x + y < 0$$

$$\gamma_1 = \cot^{-1}\frac{T \cdot \cot\sigma_1 \cdot T \cdot \cot\varepsilon_1 - 1}{T \cdot \cot\sigma_1 + T \cdot \cot\varepsilon_1}$$

Back substitution to α and β and using multiple sin/cos substitutions, leads to equation:

$$T_{1,2} = \frac{\cot(\gamma_1)\cdot\sin(2\beta)\pm\sqrt{\cot^2(\gamma_1)\cdot\sin^2(2\beta)+\cos^2(2\alpha)-\cos^2(2\beta)}}{\cos(2\alpha)+\cos(2\beta)} \qquad \text{(Equation V.1-6)}$$

$$\text{with } \gamma_1 = \gamma_1 - n\pi \qquad , n \in [0; 1]$$

V.1.4.2 Narrow Phase Diagram of Transfer in Direction 1

The mathematical description for the pronounced graph of $\vec{a}' \perp \vec{b}'$, target angle $\gamma_1 \; = \; 90°$, (see Figure V-3) is derived as follows:

$$T_{1,2} = \frac{\cot(\gamma_1)\cdot\sin(2\beta)\pm\sqrt{\cot^2(\gamma_1)\cdot\sin^2(2\beta)+\cos^2(2\alpha)-\cos^2(2\beta)}}{cos(2\alpha)+cos(2\beta)} \qquad \text{(Equation V.1-6)}$$

With $cot(90°) \; = \; 0$ and transfer factor $T > 0$, the equation simplifies to:

$$T_{1,2} = \pm\frac{\sqrt{\cos^2(2\alpha)-\cos^2(2\beta)}}{\cos(2\alpha)+\cos(2\beta)} = \sqrt{\frac{(\cos(2\alpha)-\cos(2\beta))}{(\cos(2\alpha)+\cos(2\beta))}} \qquad \text{(Equation V.1-7)}$$

In the phase diagram, the graph of the vectors being orthogonal to each other ($\vec{a}' \perp \vec{b}'$) is moving to higher values (larger T necessary) and becomes narrow while starting angle β gets larger (Figure V-7). The valid region for angle α gets smaller ($0° \leq \alpha \leq 90° - \beta$). Square lattices are found at higher transfer factors (intersections with 0°) for higher starting angle β (larger AR). All other lattice types with orthogonal vectors are rectangular.

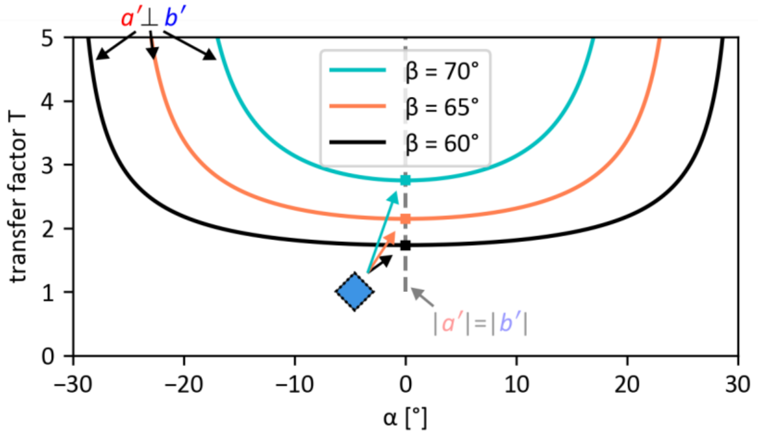


Figure V-7. Transfer factor dependency of the orthogonal spanning-vector graph on starting angle β. A square lattice is just present when the length of the vectors becomes equal (points at 0°). All other angles on the graphs resemble rectangular lattices.

On a hexagonal lattice with starting angle of $\beta = 60°$ the equation simplifies further to:

$$T = \sqrt{\frac{\cos(2\alpha)+0.5}{\cos(2\alpha)-0.5}} \qquad \text{(Equation V.1-8)}$$

The value for $\alpha = 0$ gives the known transfer factor $T = \sqrt{3}$ for square lattices. The derived equation here is much simpler than the one of Hummel et al. (compare to Equation V.1-1, Chapter V.1.2). A mathematical substitution was tried but was not successful. However, plotting the curve with both equations shows the same line (just shifted by 30°).

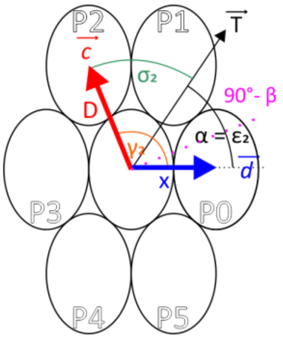
With the general Equation V.1-6, other target angles can be defined. A test of the equation with target angle $\gamma_1 = 60°$ and $\alpha = 0$ gives the known solution of $T = 3$ for open-hexagonal transfer with $T > 0$ and $cot(60°) = \sqrt{3}/3$.

V.1.4.3 Transfer in Direction 2

The Equation V.1-9 for direction 2 is valid at $90° - \beta \leq \alpha \leq 90°$.

For direction 2, the spanning vectors $\vec{c}$ (to P2) and $\vec{d}$ (to P0) have lengths D and x, and the angle δ between P1 and P2

$$\delta = 180° - 2\beta$$



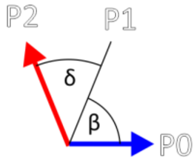
The projection angles can be described again with angle α and β:

$$\sigma_2 = \beta + \delta - \alpha = 180° - \beta - \alpha$$

$$\varepsilon_2 = \alpha$$

The projections are vectors $\overrightarrow{S_{Px}}$:

$\overrightarrow{S_{P0}} = \cos\varepsilon_2 \cdot x$ and $\overrightarrow{S_{P2}} = \cos\sigma_2 \cdot D$



The vectors orthogonal to the projection are $\overrightarrow{H_{Px}}$.

$\overrightarrow{H_{P0}} = \sin\varepsilon_2 \cdot x$ and $\overrightarrow{H_{P2}} = \sin\sigma_2 \cdot D$

Transfer equations (same as above):

$$\overrightarrow{H_{Pxnew}} = \overrightarrow{H_{Px}} \quad \text{and} \quad \overrightarrow{S_{Pxnew}} = T \cdot \overrightarrow{S_{Px}}$$

$$\sigma_{2,new} = \cot^{-1}\frac{\overrightarrow{S_{P2new}}}{\overrightarrow{H_{P2new}}} = \cot^{-1}(T \cdot \cot\sigma_2)$$

$$\varepsilon_{2,new} = \cot^{-1}\frac{\overrightarrow{S_{P0new}}}{\overrightarrow{H_{P0new}}} = \cot^{-1}(T \cdot \cot\varepsilon_2)$$

Again the new opening angle γ_2 of $\vec{c}$' and $\vec{d}$' can be described by:

$$\sigma_{2,new} + \varepsilon_{2,new} = \gamma_2$$

Using sum theorem of $\cot^{-1}$:

$$\cot^{-1} x + \cot^{-1} y = \cot^{-1}\frac{xy-1}{x+y} \quad \text{, for x + y > 0}$$

$$\gamma_2 = \cot^{-1}\frac{T \cdot \cot\sigma_2 \cdot T \cdot \cot\varepsilon_2 - 1}{T \cdot \cot\sigma_2 + T \cdot \cot\varepsilon_2}$$

By back substitution to α and β and using multiple sin/cos conversions, leads to equation:

$$T_{3,4} = \frac{\cot(\gamma_2)\cdot\sin(\beta) \pm \sqrt{\cot^2(\gamma_2)\cdot\sin^2(\beta) - \sin(2\alpha)\cdot\sin(2\alpha+2\beta)}}{-2\cdot\cos(\alpha)\cdot\cos(\alpha+\beta)} \qquad \text{(Equation V.1-9)}$$

V.1.4.4 Combined Transfer Phase Diagram of Directions 1 and 2

The resulting equation for the transfer factor can be used to complete the phase diagram.

$$T_{3,4} = \frac{\cot(\gamma_2)\cdot\sin(\beta)\pm\sqrt{\cot^2(\gamma_2)\cdot\sin^2(\beta)-\sin(2\alpha)\cdot\sin(2\alpha+2\beta)}}{-2\cdot\cos(\alpha)\cdot\cos(\alpha+\beta)} \qquad \text{(Equation V.1-9)}$$

The curve of orthogonal vectors ($\vec{c}\,' \perp \vec{d}\,'$), target angle $\gamma_2 = 90°$, is derived with $cot(90°) = 0$ and transfer factor $T > 0$. The equation simplifies to:

$$T_{3,4} = \frac{\pm\sqrt{-\sin(2\alpha)\cdot\sin(2\alpha+2\beta)}}{-2\cdot\cos(\alpha)\cdot\cos(\alpha+\beta)} \qquad \text{(Equation V.1-10)}$$

As direction 2 is mirrored on x-axis, it follows $\alpha = -\alpha$.

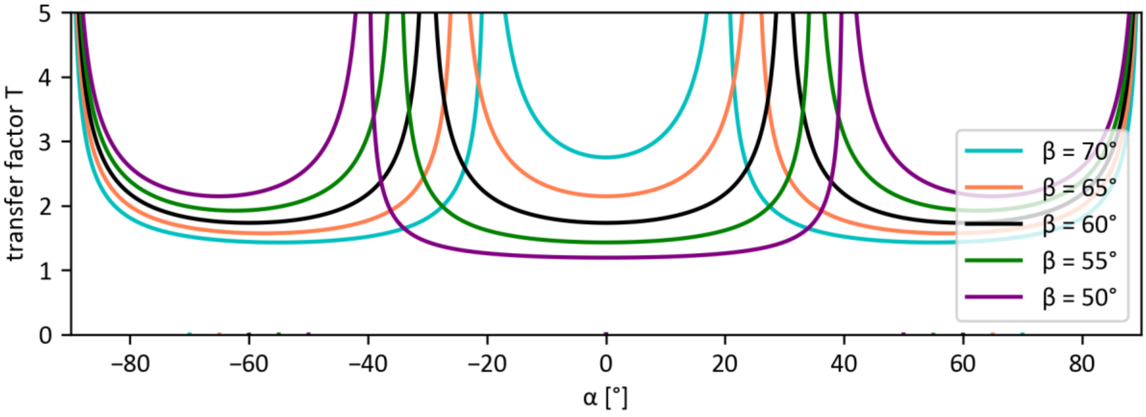


Figure V-8. Graph of transfer factor T respective to the transfer angle α for perpendicular vectors (square and rectangular BRAVAIS lattices) at different starting angle β. Note that 3 directions (mirrored direction 2, direction 1, and direction 2) with different spanning vectors but same β are depicted here in one diagram.

For hexagonal lattice of $\beta = 60°$ the graph -30° to 30° is the same as 30° to 60° (Figure V-8, black line). For larger and lower β, the two directions with their individual equation differ. For growing β, the direction-2-graph moves down and widens up into the angle regime of previous direction 1. It is not widened to > 90° as the graph mirrors there. For descending β, the direction-1-graph (Figure V-8 middle part) moves down and widens while direction-2-graph narrows.

For plotting the whole phase diagram, it is also necessary to have a look at the dependency of equal sides.

For hcp (spherical) lattices the equal sides are just possible for 0° (± $n \cdot 60°$) and at $T = 1$.

The ellipsoidal lattice starts in direction 2 already with uneven lattice vectors, leaving the possibility to transfer the vectors back to equal lattice vectors.

The graph that gives equal lattice vectors can be calculated by:

$$D_{new} = x_{new} \qquad \text{(Equation V.1-11)}$$

$$D_{new} = \sqrt{(\overrightarrow{S_{P3,new}})^2 + \left(\overrightarrow{H_{P3,new}}\right)^2} = D \cdot \sqrt{T^2 \cdot \cos^2 \sigma_2 + \sin^2 \sigma_2} \qquad \text{(Equation V.1-12)}$$

$$x_{new} = \sqrt{(\overrightarrow{S_{P0,new}})^2 + \left(\overrightarrow{H_{P0,new}}\right)^2} = x \cdot \sqrt{T^2 \cdot \cos^2 \varepsilon_2 + \sin^2 \varepsilon_2} \qquad \text{(Equation V.1-13)}$$

With $\sigma_2 = 180° - \beta - \alpha$ and $\varepsilon_2 = \alpha$ from Chapter V.1.4.3, the equation for T is calculated:

$$T = \sqrt{\frac{\left(\frac{x}{D}\right)^2 \cdot sin^2(\alpha) - sin^2(\alpha+\beta)}{cos^2(\alpha+\beta) - \left(\frac{x}{D}\right)^2 \cdot cos^2(\alpha)}} \qquad \text{(Equation V.1-14)}$$

With $\cos(\beta) = \frac{x}{2D}$ the equation can be written as:

$$T = \sqrt{\frac{4 \cdot sin^2(\alpha) \cdot \cos^2(\beta) - sin^2(\alpha+\beta)}{cos^2(\alpha+\beta) - 4 \cdot cos^2(\alpha) \cdot \cos^2(\beta)}} \qquad \text{(Equation V.1-15)}$$

For hcp ($\beta = 60°$), $D = x$ initially and the Equation V.1-15 simplifies to the line at $T = 1$ for all α. For other angles of β, $D \neq x$ initially and the Equation V.1-15 results in a non-linear graph for T resembling equal sized vectors for different α. Note that the equation is just defined at angles of direction 2 ($90° - \beta \leq \alpha \leq 90°$).

The intersection of the Equations V.1-15 and V.1-10 will then give square lattice positions.

V.1.4.5 Transfer Phase Diagram of CP Ellipses with AR 1.42

The three directions (direction 1, direction 2, and mirrored direction 2) can be drawn in the same phase diagram for a given AR when taking into account that the lattice spanning vectors change with angles (AR 1.42, Figure V-9). Note that the vectors of direction 1 ($\vec{a}$ and $\vec{b}$) will stay constant for every α when not transferred at all ($T = 1$). However, the graphs are just plotted on the region of angles where the respective vectors span the lattice.

The diagram was developed with S configuration, but it is as well valid for the L configuration, as one has just to change the AR to the inverse. The L configuration phase diagram for AR 1.42 can be found in Figure V-9 as well. Further diagrams with different ARs and starting lattices S and L can be found in the supporting information V.8.1. Note here that at L configuration the graph of direction 1 will “vanish“ when the AR exceeds 1.73 and it will be not possible to have an intersection giving a squared lattice as the angle γ in between $\vec{a}$ and $\vec{b}$ is lower than 90°.

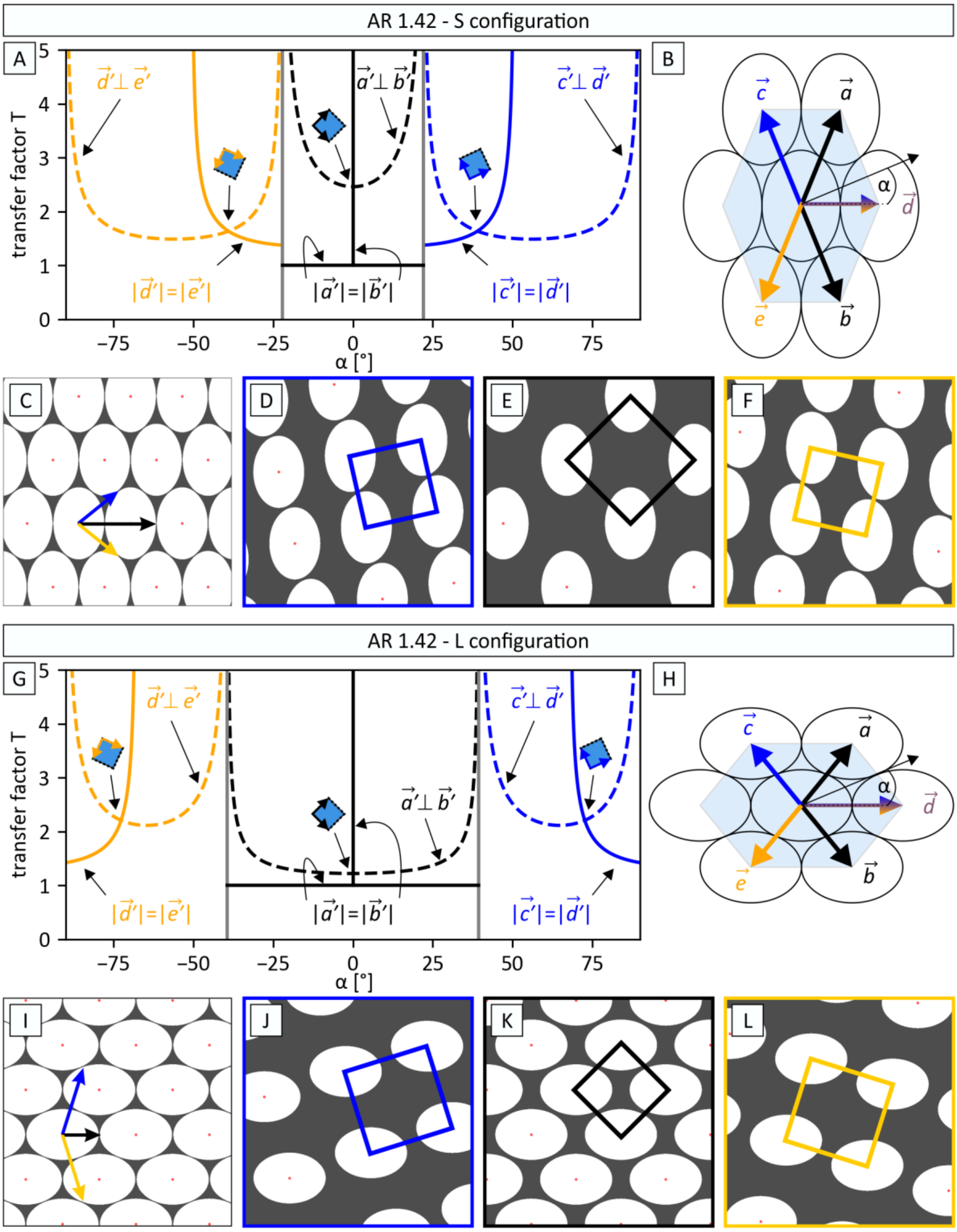


Figure V-9. A, G: Transfer phase diagrams of lattices with AR 1.42 starting from S or L configuration. Grey lines separate the 3 directions of different spanning vectors and have no further meaning. B, H: Sketches showing the different vectors that span the lattices. C – F: simulated data of S configuration. C: close-packed lattice and D – F: after transfer to the square lattice points shown in A. I – L: Same as C – F but for L configuration. See supporting information V.8.1 for phase diagrams of other AR. Simulation of data, see next sub-chapter.

The phase diagrams reveal three points where the respective spanning vectors become orthogonal to each other and have the same length. Thus, square lattices are accessible at those intersection points for the AR 1.42 lattices (Figure V-9 D – F and J – K).

V.1.5 Simulated Datasets of Particle Lattices Made in Python

Test datasets and images of ellipses with different ARs and different orientations were generated by python (code see Appendix VIII.6.3). For visualization and getting the lattice positions, an hcp lattice undergoes an affine transformation of rotating the start lattice by an angle 0° ≤ α < 60°, followed by a 1D transfer by factor T along the x-axis, and deploying a back rotation. The angular range does not need to exceed 60° as the hexagonal lattice repeats itself then.

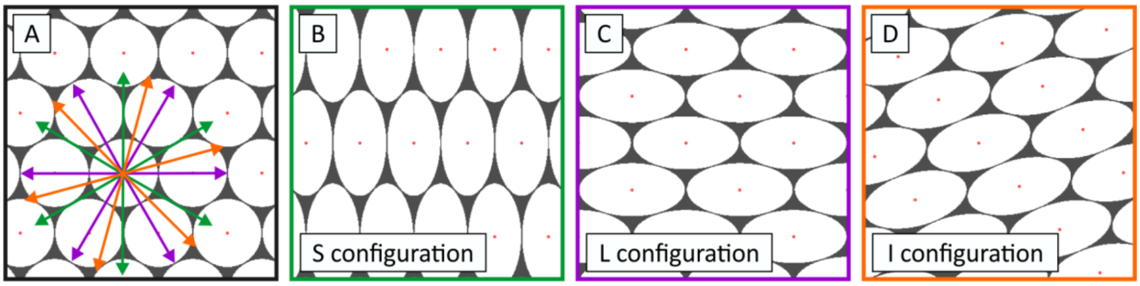


Figure V-10. Computer created particle positions (red dots) for the $g(r)$ calculation. A: Spheres (hcp), B, C and D: By affine transformation received ellipses with AR 2. B: S configuration and C: L configuration were simulated. Additionally, further possible structures with other transfer angles e.g. for D: 15° (intermediate configuration I) can be simulated as well. At an intermediate configuration, the next ellipse does not touch directly in the direction of an ellipse axis.

The previously introduced lattices with short and long axis contact of the ellipses (S and L configuration, Figure V-10 B and C) and the intermediate oriented ellipses lattices (I configuration, Figure V-10 D) can all be reproduced with the python script by affine transformation of a hexagonal close-packed lattice (Figure V-10 A). Further implementations gave the possibility to start with other lattices, like square lattice or already-transferred lattices (simulating transfer of close-packed ellipses). An example of the simulated transfer of cp ellipse lattices of AR 1.42 using the parameters of square lattice points of the respective phase diagram can be found in Figure V-9. Square lattices with other AR (1, 2, and 3) are depicted in the supporting information V.8.2.

V.1.6 Pair Correlation Function $g(r)$

Particle lattices can be investigated by their spherical distribution function, which is also called the radial distribution or pair correlation function $g(r)$. In this chapter, the spherical distribution function is implemented to investigate the particle position data of colloidal monolayers in the following chapters (data from python program of Chapter III.3).

The spherical distribution function shows the possibility to find particles at a certain distance (r) around a center particle [152]. The principle of how to evaluate $g(r)$ is that at certain distances from the particle center the particle density rises. Those positions can be referred to the lattice vectors. Crystalline lattices have distinct particle positions and distances and, therefore, a very distinct $g(r)$ (lines in diagram at Figure V-11).

The $g(r)$ for 2D is calculated analogous to the paper of Kopera and Retsch [153]. The number of particles in the sample ($\#P_{sample}$) and the sample imaged area (A_{sample}) give the average particle density of the sample (ρ_{sample}). The radial particle density is defined in the area by counting the number of particles ($\#P(r, \Delta r)$) in defined bins (bin size Δr around given radius) and the area of the ring spanning the bin ($A_{ring}(r, \Delta r)$).

$$A_{ring}(r, \Delta r) = \pi \cdot \left[\left(r + \frac{\Delta r}{2}\right)^2 - \left(r - \frac{\Delta r}{2}\right)^2\right] \qquad \text{(Equation V.1-16)}$$

The local $g(r)$ is given by the following Equation V.1-17:

$$g_{local}(r, \Delta r) = \frac{\rho_r}{\rho_{sample}} = \frac{\frac{\#P(r,\Delta r)}{A_{ring}(r,\Delta r)}}{\frac{\#P_{sample}}{A_{sample}}} \qquad \text{(Equation V.1-17)}$$

This local $g(r)$ is averaged over all particles in the sample. The calculations were performed adapting the python script of my former colleague Dr. Bernd A.F. Kopera.

V.1.6.1 Python Implementation and Program Adjustments

The main function and python implementation of the $g(r)$ calculation was extracted from the work of my former colleague Dr. Bernd A.F. Kopera. He published his calculation on a 3D particle assembly [153] but also implemented the algorithm in python for the use in 2D (single images). His code was then adjusted and enhanced by myself, to use it with the resulting particle tables of my own python program (code snippet of the calculation see Appendix VIII.6.4).

Adjustments of the code were e.g. that particles in bins with small sizes (< 5 pxl², fractions of circles) were filtered off because their $g(r)$ would be high (as the particle number is divided through area) and then overlaid the other data. As the $g(r)$ converge to 1 for real samples very fast and the positions of the first few peaks are the most important ones in terms of comparing the samples, I defined to calculated the $g(r)$ just for circles with maximum radius 10 times the minor diameter of the particle. This shortened the computing time for the calculation drastically. Lastly, I set the standard bin size to 1 pxl.

V.1.6.2 *g(r)* of Square and Hexagonal Lattices

The resulting pair distribution function can be compared to reference lattices like square and hexagonal lattices. With square lattice, the next neighbor distance is 4 neighbors with distance $1d$ (d: smallest distance from center to center; at monodisperse spheres it is resembled by the diameter of the particles) and 4 neighbors with distance $\sqrt{2}d$. Then another 4 with distance $2d$. As the radius grows, and with it the area, the peak height decreases. This is the same with any other lattice. In a hexagonal lattice, the center particle has 6 neighbors with distances of $1d$, followed by 6 with $\sqrt{12}d$ (Figure V-11).

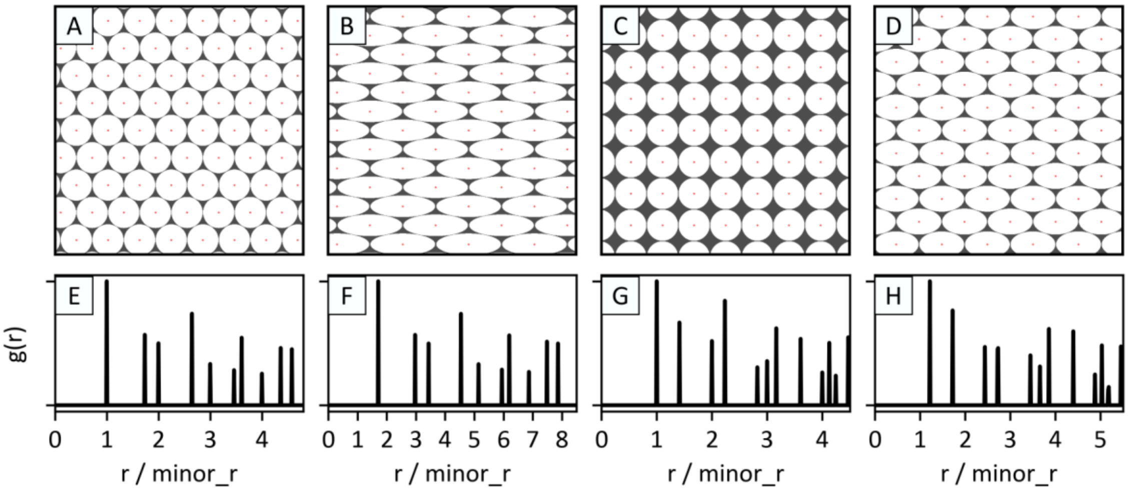


Figure V-11. A – D: Simulated test data images with E – H: corresponding pair correlation function of hexagonal lattices (A: AR 1, B: AR 3) and square lattices (C: AR 1, D: AR 1.73).

As expected from the phase diagrams of the previous chapters, a transfer of an hcp lattice by $T = 3$ (= AR 3) in the direction of particle contact lattice (Figure V-11 A to B) delivers the same $g(r)$ as for the original hcp lattice with $T = 1$. Hence, they have both a hexagonal lattice (compare Figure V-11 E and F).

Consequently, a transfer of an hcp lattice by $T = \sqrt{3}$ (= AR 1.73) in the direction of particle contact (Figure V-11 D) show square lattice positions (compare Figure V-11 G and H).

V.1.6.3 Height of Peaks

The height of the single peaks of the $g(r)$ function is dependent on the amount of data points in the dataset as $g(r)$ is divided through the particle density of the sample. Additionally the image pixel resolution and bin size used for the $g(r)$ calculation defines the height of the peaks as well. To give an example, it is adjustable by these parameters if data with $r = 1.00$ and $r = 1.01$ will be counted to the same bin, or not. Therefore, the absolute peak values of the $g(r)$ function will not be evaluated.

V.1.6.4 Normalized Peak Positions

As the distances of the peak positions differ in terms of absolute values (for small particles the distance from center to center, while in contact, might be 350 nm, for big particles e.g. 1300 nm) the $g(r)$ is normally divided through the smallest center-to-center distance possible (parameter in python code "minor_r"). At spheres, this would be the particle diameter. To compare different lattices, the $g(r)$ can further be normalized by dividing through the r-value of the first peak (smallest center-to-center distance of particles in the lattice). Then the peak positions fall on the same values as known lattice distances and the lattice can be identified easily. In addition to this benefit, the normalization corrected a possible influence of the resolution from the given image (low resolution led to broad peaks) on the pair correlation function (Figure V-12).

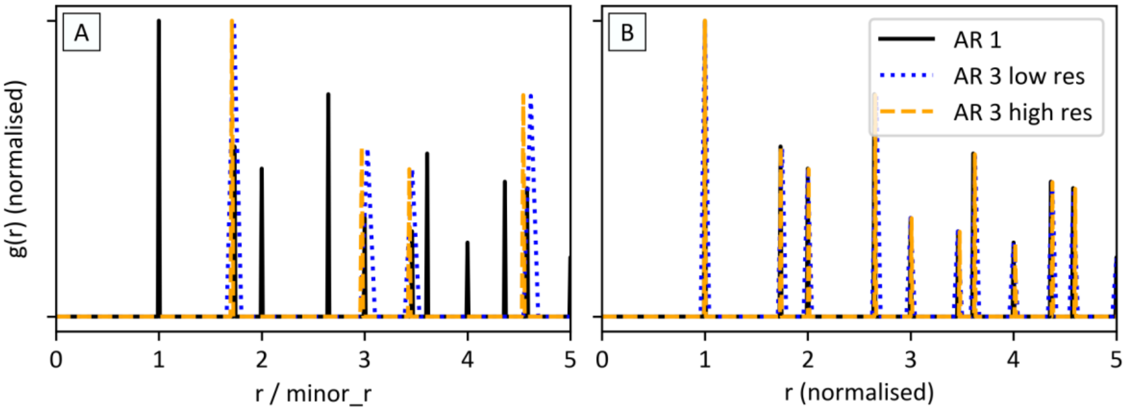


Figure V-12. Pair distribution functions for hexagonal lattices and different ARs or resolutions (low res test data: 800 pxl × 800 pxl, minor_r ≈ 14 pxl; high res test data: 4000 pxl × 4000 pxl, minor_r ≈ 71 pxl; all datasets had around 1200 positions). A: $g(r)$ plot of hexagonal lattice with r divided through minor_r. Deviation of peaks by different pixel sizes/resolutions. Low resolution results in wider peaks. B: To compare the lattice, the graphs were normalized to the first peak. The other peaks come all to the same positions; hence, they describe hexagonal lattices.

V.1.7 Angular Correlation Function $g_2(r)$

To describe ellipsoid packing, Zheng and Han [105] not just used the position pair correlation function, but an angle dependent correlation function (Equation V.1-18). Thereby, the orientation angle of the particle at distance r ($\theta(r)$) is subtracted from the orientation of the center particle ($\theta(0)$). The term is then “normalized“ to the range of -1 to 1.

$$g_2^{\theta}(r) = \cos(2 \cdot [\theta(0) - \theta(r)]) \qquad \text{(Equation V.1-18)}$$

Thereby, results 1, 0, and -1 refer to parallel, random and perpendicular arrangements of ellipsoid-to-ellipsoid. The values are binned to mean or median values to get a curve.

For parallel particle layers, like in the described transformed particle layers, the graph would become 1 over the whole sample (Figure V-13 A). As $g_2^{\theta}(r)$ cannot become -1 (perpendicular) for the whole sample, because it is impossible that more than 2 particles can be all orthogonal to each other in 2D lattices, just a few positions can become -1. At the same time, this would lead to particles of parallel orientations and a zigzag graph between 1 and -1 evolves (Figure V-13 B). Binning with large bins would then lead to mean values between -1 to 1. For random oriented particles, the plot fluctuates around zero (Figure V-13 C). Thus, the angle dependent correlation function is best to be used to compare parallelly-ordered to unordered regions.

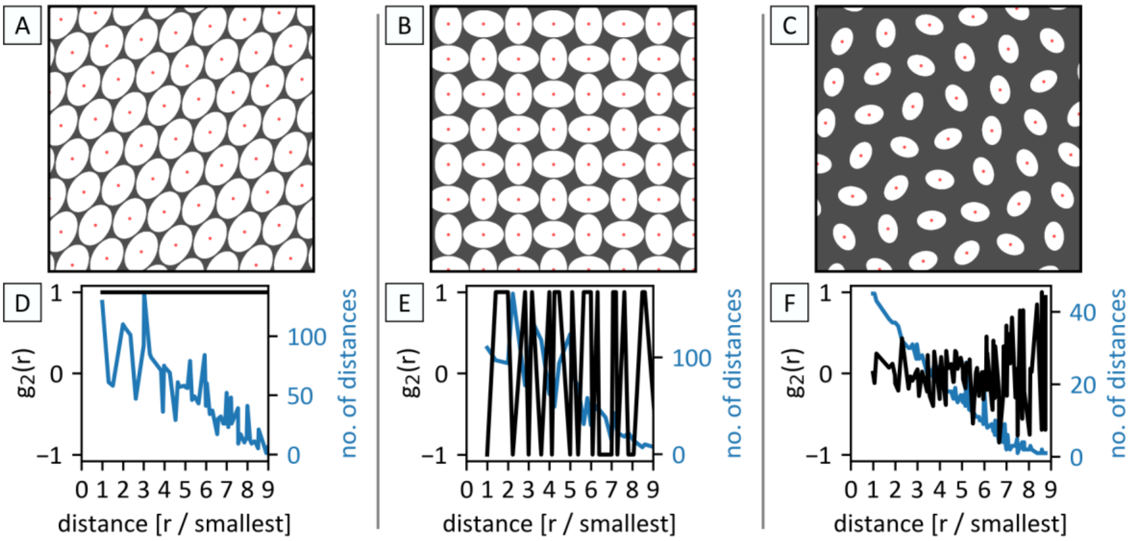


Figure V-13. A – C: Test data and D – F: angular pair correlation functions $g_2^{\theta}(r)$ (black) for different particle orientations and numbers of compared particles (blue). A: parallel (1), B: orthogonal alternating (1 or -1), and C: randomly oriented. The number of compared particle distances decreases with distance due to sample size.

For displaying the angle dependent correlation function, a separate python script extracts the particle distances and orientations from the data of the ellipse detection program, calculates the $g_2^{\theta}(r)$ and plots the graph (code see Appendix VIII.6.5).

V.1.8 *FOURIER* Transformation of Images

Apart from the calculations of pair correlation functions, there is the possibility to investigate a sample by making a *FOURIER* transformation of the given image by Fiji/ImageJ. The resulting image in frequency domain, either displays a point pattern for crystalline or a blurred region for non-crystalline lattices. The distinct peaks give the lattice spacing in reciprocal dimensions. Thus, the further out the peak from the center, the narrower the spacing. The values for lattice spacing can be recorded by using the point value tool in Fiji/ImageJ. At hcp layers (particles in contact), the peak closest to the center should be the diameter of the particle.

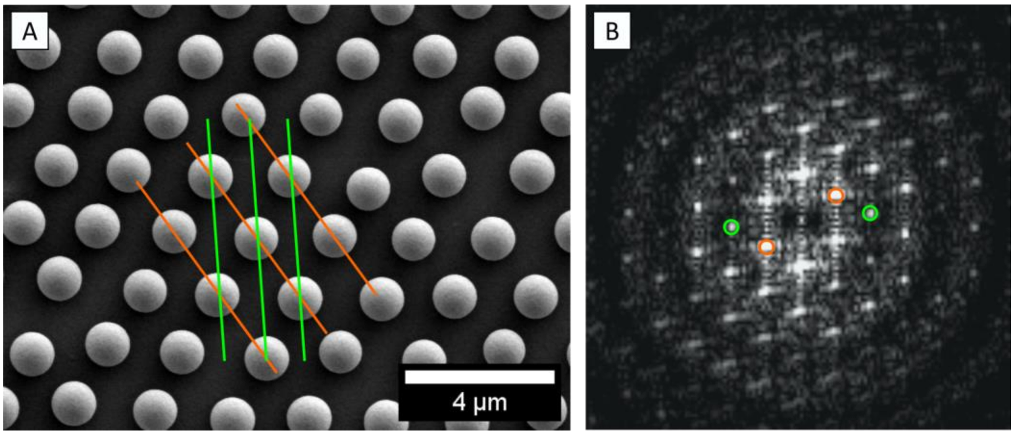


Figure V-14. A: SEM image (transferred lattice of particles DB13, see Chapter V.3) and B: its FOURIER transformation (FFT) by Fiji/ImageJ. The periodic lattice can be seen with bright spots in the FFT image. Lattice planes and their respective FFT representation: 1.67 µm (orange) and 1.09 µm (green).

The *FOURIER* transfer pattern can be compared to the laser diffraction pattern. However, for laser diffraction just the first order diffraction was recorded with the given setup (see Appendix VIII.1.9).

V.2. Floating Monolayers by Self-Assembly

Two-dimensional assemblies of particles can be divided into dense layers and open layers. Both are often used in particle lithography. Monolayers of polymer particles can be made by floating them on water/liquid surfaces. In this sub-chapter, two well-known methods were employed for producing densely-packed floating particle monolayers.

V.2.1 Theory of Floating Particle Monolayer Assembly

One easy approach to fabricate close-packed particle monolayers is the self-assembly of particles on liquid surfaces or at liquid surface contact lines. Thereby, particles are pinned at the air-liquid interface. The assembly tries to gain a local energy minimum balancing the acting repulsive (electrostatic or steric) and attractive (mostly capillary) forces (comparable to *DLVO* theory, see *Physics and Chemistry of Interfaces, Chapter 6.5.3* [154]). An overview on the different forces acting on particles (Figure V-15), can be reviewed e.g. at Maestro and Guzmán [155].

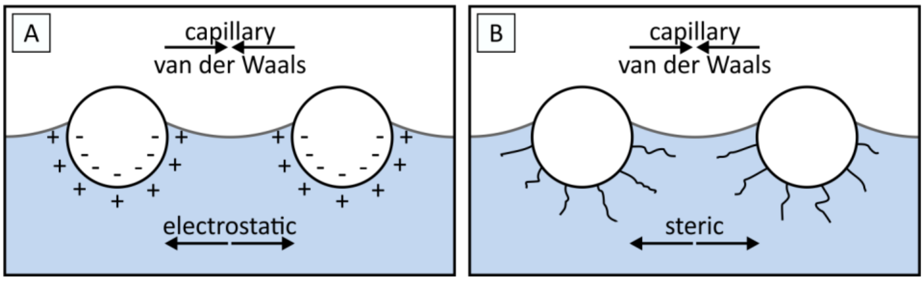


Figure V-15. Overview of repulsive and attractive forces on particles floating on a water surface. A: (Charged) particles have repulsive electrostatic and attractive capillary and VAN DER WAALS forces. B: Ligands on the particles or additives in the sub-phase can give (additional) steric repulsive forces. With possible leftovers of PVA on the particle surface or additions of surfactants to the sub-phase, all kinds of forces (A and B) are relevant for the assembly of the particles used here.

V.2.1.1 Interplay of Forces during Assembly

For a defect-less monolayer assembly, it is necessary that the attractive forces are in balance with the repulsive ones, so that the particles have the possibility to find an energetic minimum position. If the attractive forces are too strong, the particles cluster immediately and are unable to form close-packed layers. However, if the attractive forces are lower than the repulsive ones, the particles form open-packed layers. This is used e.g. to retain open-packed hexagonal lattices by countering capillary forces with an organic phase on top of the particles [135, 136].

The interplay of forces can be adjusted in multiple ways. The repulsive electrostatic forces (Figure V-15 A). can be screened e.g. by adding salt [156, 157]. Raising or lowering the pH value will lead to protonation/deprotonation of surface groups on the particles leading to higher or lower electrostatic forces and different wettability. The wettability alters the capillary force and might raise or lower the particle at the air/water interface [158]. Note that this aspect was investigated with ellipsoids in Chapter VI for a water filter application.

Capillary forces are the main attractive force in particle assemblies at aqueous interfaces. With surfactants, the wettability and, therefore, the capillary force can be altered. Additionally, the surfactant can screen charges and deliver steric interactions (Figure V-15 B). One standard surfactant used in particle monolayer assembly is sodium dodecyl sulfate (SDS) with a sulfate negative-charged polar head group and a non-polar dodecyl tail. An example for a positive polar surfactant is 2-Methacryloxyethyl-trimethylammoniumchloride (MTC) with an amine head group (same as in synthesis Chapter II.1.3.2), while uncharged surfactants can be block-co-polymers with hydrophilic and hydrophobic blocks.

V.2.1.2 Defects of Particle Monolayers

A hexagonally close-packed monolayer (hcp) is the favored lattice for spherical monodisperse particle assemblies in 2D. Typical defects within the hcp monolayer can be point defects like different sized particles (due to synthesis), missing particles (Figure IV-16 orange arrows), or different shaped particles (Figure IV-16 yellow circles). It is known from literature that the frequently observed cylindrical particles are formed during stirring of the polymer particle dispersion (plastic deformation). The cylinders contain voids as Tripathi and Tsavalas investigated [159]. Older particle dispersions like the ones used in Chapter II.3.1 (small particle batches with names FN), display more cylinders. Other typical defects of monolayers are line defects like grain boundaries (shift in lattice orientation).

With the transfer of a floating particle monolayer onto a dry substrate, other defects, like popped out particles, from particles clamped between the monolayer and the substrate (Figure IV-16 C hexagons) or dislocations from drying (no shift in lattice orientation), can occur. However, these two defects can be minimized by carefully transferring the particles to the water (e.g. by glass slide [92]), leaving no/less particles in the water phase beneath the monolayer (see also supporting information V.8.3), followed by drying the particle monolayer under a tilted angle.

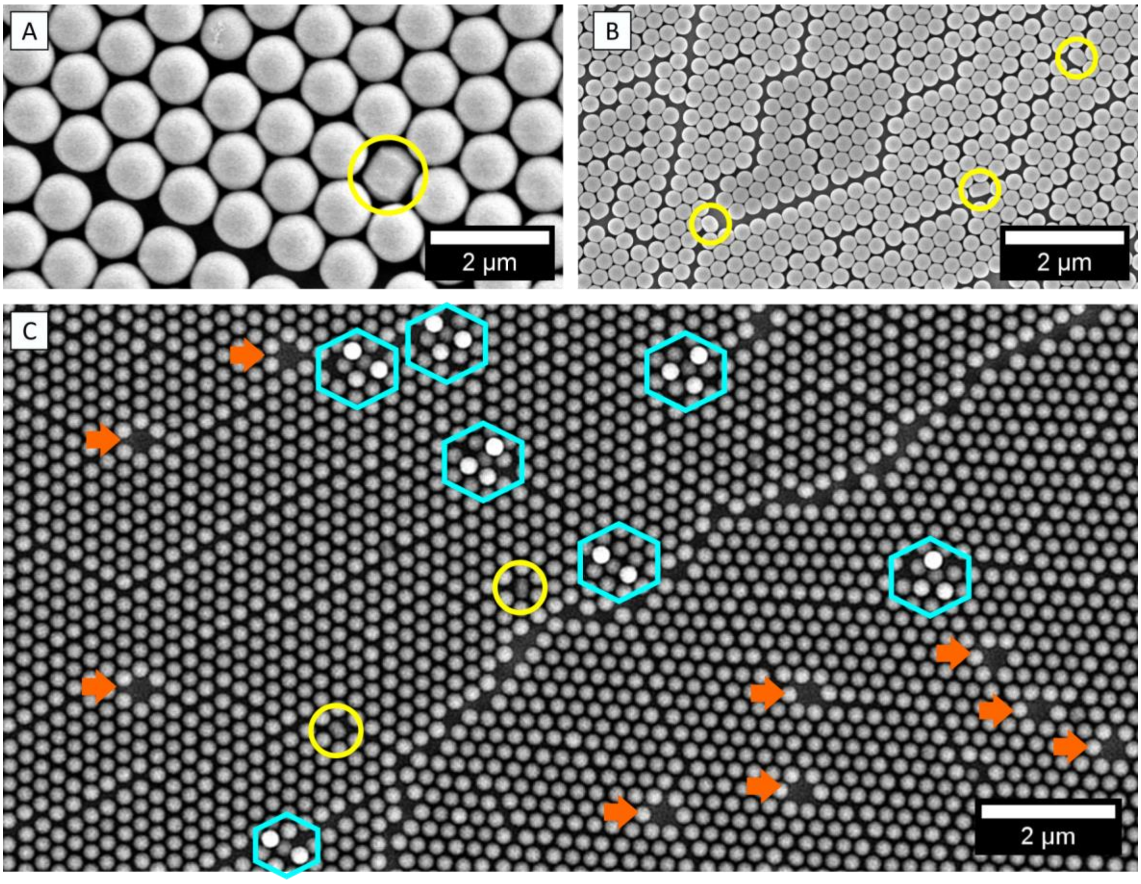


Figure V-16. Floated hcp monolayers of A: big (DB13, 1.15 µm) and B, C: small particles (FN269, 0.33 µm). Monolayers showed point defects like cylindrical particles (yellow circles), popped out particles (cyan hexagons), and missing particles (orange arrows). Dislocations are visible in A, B (no lattice shift), while image C depicts a grain boundary (shift in lattice orientation) and faint dislocations.

V.2.1.3 Consideration for Ellipsoidal Particle Monolayers

The ellipsoidal particles show some specialties at the water surface. The curvature of an ellipsoidal particle influences the air/water interface deformation so that the tips of the particle are wetted less than the sides of the ellipsoid. The ellipsoidal particles display an uneven, saddle-like deformation of the water surface that can lead to different assembly behaviors [143, 144]. Therefore, pair potential investigations of ellipsoids on water-oil interfaces by Luo et al. [145] showed a favored assembly of particles with a high AR in a side-by-side assembly, forming ribbon like structures. Also, patches of pseudo nematic domains might form at high AR as Zheng and Han showed [105].

With enough pressure from surrounding particles (high particle concentration during assembly), the ellipsoids in the monolayer might be pushed onto another one with one of their tips pointing upwards (particle flipping) [160].

V.2.2 Two Applied Methods – *RETSCH* & *VOGEL*

For floating particles in monolayers, two different methods were used within this thesis.

The first method implemented by Retsch et al. [129] (short: *RETSCH* method) worked well with the big particles. As for this method just a few drops of the particle dispersion with low wt% was needed for spin-coating, it was the first choice to make the monolayer assembly with the large ellipsoidal particles (Table V-3).

The second method was the main method used by Vogel et al. [130] (short: *VOGEL* method). For this method, the particle suspensions need to have concentrations of more than 5 wt%. Thus, the method is suited for larger batches of particles. A benefit of monolayers made by the *VOGEL* method is that they are just limited in size by the sub-phase dimension, while with the *RETSCH* method monolayers are limited due to the amount of particles distributed on the spin-coated substrate.

*Table V-3. Particles used for monolayer assembly studies. Methods R (RETSCH) and V (VOGEL) were applied on different batches due to the necessary amount of particles in the given methods. Assemblies of the small FN269 particles (particle type 1, PS-AA, Chapter II.1.3.1) were performed in 2019/2020, and of the large DB07 and DB13 particles (particle type 3, PS-PSS, Chapter II.1.3.3.2) in 2021/2022. Zeta potential measured in 0.9 mM NaCl or *0.9 mM KNO3. Particles were stretched in the oven by hand or in the roll-stretching device. ARs of spherical particles were 1.0 ± 0.0 for all batches, thus noted here as 1 without deviation.*

particles		**method**	**synthesis**	**AR**	**long axis [µm]**	**short axis [µm]**	**zeta potential**
DB07	0 %	R, V	disp. pol.	1	1.06 ± 0.03		-64 mV
	25 %	R	oven	1.7 ± 0.1	1.53 ± 0.08	0.88 ± 0.04	-70 mV
	50 %	R	oven	2.5 ± 0.2	1.97 ± 0.12	0.80 ± 0.05	-67 mV
	75 %	R	oven	3.3 ± 0.4	2.40 ± 0.16	0.74 ± 0.05	-63 mV
DB13	0 %	R, V	disp. pol.	1	1.15 ± 0.02		-70 mV
	15 %_1	R	roll	1.39 ± 0.07	1.41 ± 0.06	1.02 ± 0.03	n/a
	15 %_2	V	roll	1.42 ± 0.07	1.47 ± 0.06	1.03 ± 0.03	n/a
FN269	AR 1	V	emul. pol.	1	326 ± 7 [nm]		-46 mV
	AR 2	V	oven	2.2 ± 0.1	556 ± 22 [nm]	254 ± 10 [nm]	-37 mV
	AR 3	V	oven	3.2 ± 0.2	718 ± 35 [nm]	225 ± 10 [nm]	-22 mV*
	AR 4	V	oven	4.3 ± 0.4	901 ± 47 [nm]	211 ± 12 [nm]	-25 mV*

V.2.2.1 Method One – *RETSCH*

The particle monolayer preparation method *RETSCH* contained two easy steps for preparing colloidal monolayers floating on a liquid surface. Firstly, the particle dispersion is spin-coated onto a plasma-activated glass substrate. Secondly, the resulting unordered open-packed two-dimensional layer of particles on the glass substrate is immersed into a liquid. The particles detach from the glass and assemble into a close-packed monolayer at the 3-phase contact line. The resulting monolayer floats on the liquid surface and can be lifted off. If the particles do not detach from the spin-coated glass (mostly with small particles), a functionalization of the glass substrate prior spin-coating e.g. by liquid phase silanization as explained by Retsch et al. [129] needs to be done.

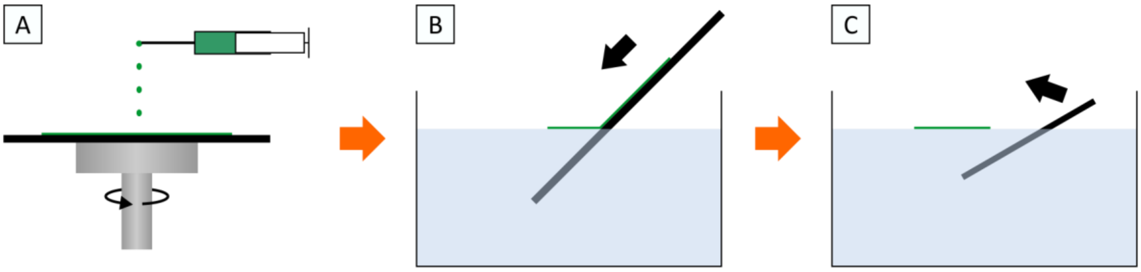


Figure V-17. Schematic drawing of the RETSCH method. A: The particle dispersion (green) is spin-coated onto the substrate (black) to form an open-packed two-dimensional particle layer. B: The substrate is immersed into a liquid and the particles detach and order at the 3-phase contact line. C: The ordered monolayer can be scooped from the liquid surface by another substrate. The later discussed results are from particle layers scooped always in the same direction as the spin coated substrate was immersed.

V.2.2.1.1 Standard Protocol *RETSCH* Method

100 – 200 µl of a 1 wt% containing particle dispersion (wt% and amount not fixed, more particles gave a larger monolayer) was spun on a freshly-plasma-activated glass slide (one-half of a Hellmanex-III-cleaned microscopy glass slide). The here used spin-coater program was either just 1000 rpm for 30 – 60 seconds or a 4-step program (step1: 200 rpm, ramp 20 s; step 2: 300 rpm, ramp 20 s; step 3: 400 rpm, ramp 20 s, dwell 40 s; step 4: 4000 rpm, ramp 0 s, dwell 10 s). The coated glass slide exhibited a turbid appearance.

The glass slide was inserted into a MilliQ water basin by hand with a tweezer at an angle around 45° – 80°. For the large PS-PSS particles (e.g. DB batches, Chapter II.1.3.3), some HCl in the sub-phase helped to detach and stabilize the generated monolayer. For the PS-AA particles (e.g. small particles from FN, Chapter II.1.3.1), some SDS (0.1 mM) in the sub-phase was beneficial.

V.2.2.1.2 Increasing Monolayer Size

The amount of particles on the to-be-immersed glass slide defined the area of the resulting monolayer and was, therefore, limited. As the particles on the glass slide were pushed together during assembly, the area covered by the resulting monolayer was just a fraction of the size of the glass slide. To enlarge the liquid area covered by monolayers, the monolayer forming process could be repeated with multiple samples onto the same sub-phase. The new monolayer could be deposited directly next to the formerly prepared layer or even in contact with this layer. However, with the *RETSCH* method, the grainsize was generally not large (Figure V-18).

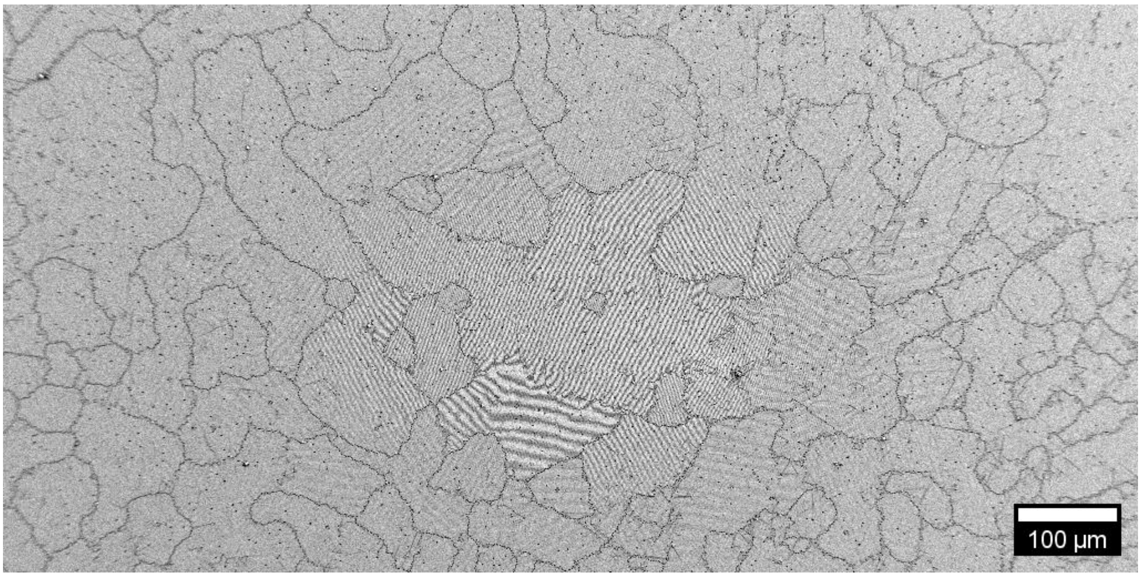


Figure V-18. SEM image as example of a grainy monolayer from the RETSCH method. Interference lines (Moiré-patterns) show the different orientations of the lattice inside the grains. Particle type: DB13_0 % spherical.

V.2.2.2 Method Two – *VOGEL*

Large close-packed monolayers could be created by direct assembly of a dispersion on a 3-phase contact line. Vogel et al. [130] used a glass slide as a ramp, where the dispersion is spreading down towards the water surface (Figure V-19). By contact with the water phase, the particles start to assemble while lifted onto the water surface by the spreading agent. Without a spreading agent (mostly ethanol), the particles tend to submerge. To improve the packing, the right ratio of particle content and spreading agent is necessary. Here, it was found that with larger particles, the content of ethanol has to be increased.

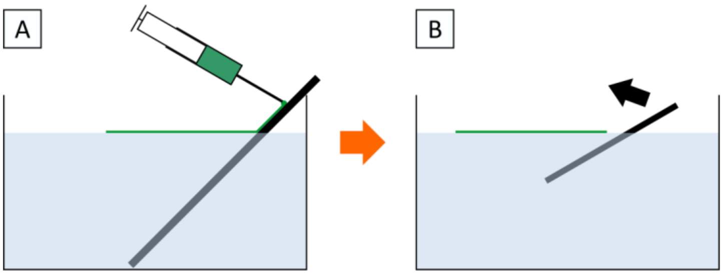


Figure V-19. Schematic drawing of the VOGEL method. A: The dispersion (green) is directly applied on the ramp (black) and spreads while moving downwards. On the water contact line, the particles assemble into monolayers. Monolayer size is just dependent on the amount of dispersion used. B: The monolayer (green) can be scooped from the liquid surface.

Surfactants like sodium dodecyl sulfate (SDS) within the sub-phase can favor dense hexagonal packing. Especially for smaller particles, SDS was favored. However, in my findings with the ~1 µm sized particles with sulfate groups (PS-PSS, Chapter II.1.3.3), the assembly on water containing SDS was not possible. A lowering of the pH of the sub-phase was mostly beneficial here for close-packed particle monolayers [92].

However, especially with the big particles, large-area monolayers could be made on top of pure MilliQ as well, without using surfactants or different pH values.

V.2.2.2.1 Standard Protocol *VOGEL* Method

A plasma activated glass slide (pre-cleaned with Hellmanex III) was mounted above a basin with a sub-phase e.g. a 0.1 mM SDS solution or pure MilliQ in an angle of 45° – 60° to the sub-phase surface. The lower part of the glass slide was brought into contact with the sub-phase. To favor packing, the pH value of the sub-phase was adjusted before applying the dispersion by drops of ammonia solution ($NH3_{(aq)}$) or by using $NaOH_{(aq)}$ (0.1 M) or $HCl_{(aq)}$ (0.5 – 0.1 M) as a sub-phase. For making the spreading dispersion, a particle dispersion with 5 – 16 wt% particles in water was mixed with ethanol in a volume ratio of 2:1 (small 326 nm particles 5 wt%, big 1 µm particles 5 – 16 wt%). This spreading dispersion was then transferred onto the glass slide by a syringe (Braun, needle diameter 0.4 mm or smaller). By applying a quasi-constant flow by hand (syringe pump possible, but needed much dispersion volume), the monolayer started to spread on the contact line in a straight fashion away from the glass ramp. To counter cross flow, which resulted in a turning of the monolayer, e.g. glass slide shields on both sides of the ramp could be used (see supporting information V.8.3). The area of the basin could be covered completely (Figure V-20) when moving the prepared monolayer out of the way of the to-be-prepared monolayer. As the monolayer was normally quite stiff, this could be achieved mechanically by a scalpel or via gas flow by blowing gently.

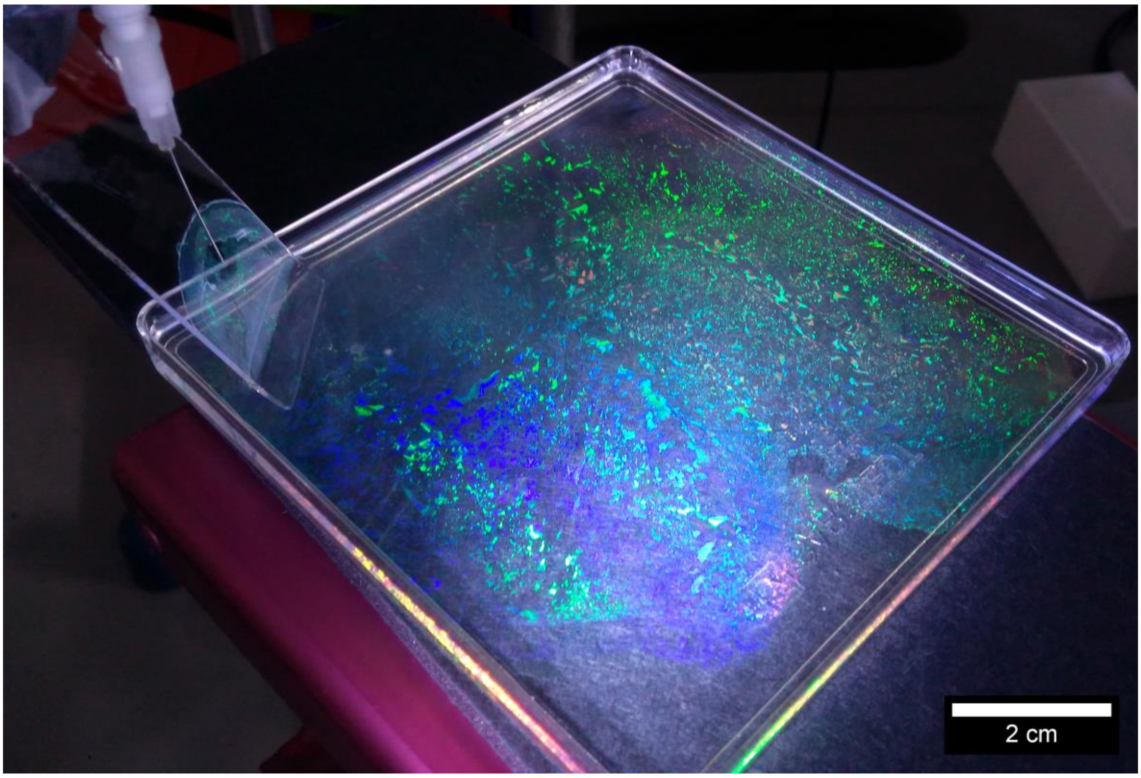


Figure V-20. Photograph of the applied VOGEL method for monolayer assembly of spherical 326 nm particles (FN269 AR 1). Same colored areas indicate large ordered areas of same lattice orientation. The basin was a 10 cm × 10 cm plastic petri dish and the ramp a microscopy glass slide. A torch from side-bellow illuminates the monolayer. Scale bar estimated by diagonal dimension of the basin in the center of the picture.

The Monolayer could then be scooped by a plasma activated glass slide or silicon wafer piece and dried at an angle of approximately 60°. During deposition, additional particles located below the monolayer pushed particles out of the cp layer. It is, therefore, beneficial to avoid submersing particles or to transfer a floating layer from a turbid sub-phase to a clean sub-phase prior deposition (see supporting information V.8.3).

V.2.2.2.2 Amount of Spreading Agent and Particle Concentration

A regular monolayer normally spread in a slow to medium speed (depending on the input volume; around 2 cm/s).

However, if the ethanol content of the dispersion was too high, the spreading was much faster and the particle rushed over the surface. No monolayer or just flakes of monolayers (swirling around) formed. However, when the ethanol content was too low, the monolayer was not transferred to the water surface, which resulted in a turbid sub-phase.

If the particle concentration was too low, the resulting monolayer did not show regular packing or a very instable ribbon-like ordered area occurred. This area grew slowly. If the concentration was too high, white streaks appeared where multiple particle were on top of each other.

V.2.2.2.3 Enlarging Grain Size (Big Spherical Particles)

The resulting large-size monolayer of spherical particles was polycrystalline with distinct grains (Figure V-20). A simple possibility to enlarge the grainsize with big particle monolayers was found by blowing gently directly perpendicular onto the center of the monolayer with an air-pistol and moving the air stream back and forth (for grain growth by air stream see also Meng et al. [161]). The grains grew as the area of same reflexes became larger and laser diffraction showed distinct hexagonal patterns with big particle monolayers. This was, however, just a hands-on finding and needs to be investigated further. Small area monolayers could not be grain grown by the air-pistol as they fractured in smaller pieces. Other possibilities to grow grains at particle monolayers might be the use of sound waves [162] or barrier sways on *LANGMUIR* troughs [163] (not used within this thesis).

V.2.2.2.4 Transfer a Monolayer to Other Sub-Phases

The monolayer preparation by the *VOGEL* method might lead to a turbid sub-phase. A transfer from the turbid to a clean new sub-phase helped to improve the dried monolayer quality (less popped out particles, Figure V-16).

To achieve this, a turbid-sub-phase monolayer of big PS-PSS particles on 0.5 M $HCl_{(aq)}$ without SDS was scooped from the preparation container by a plasma-activated glass slide and was then immediately transferred (without drying) analogous to the *RETSCH* method to a storage container with a clean sub-phase containing a 0.025 M SDS solution.

The monolayer remained densely-packed and could be further used. An additional advantage of this transfer process was the possibility to change the sub-phases to different pH values (possibility to adjust immersion depth of particles, see Chapter VI.4).

The clean sub-phase monolayer of big spherical particles was then grain grown by air stream and finally used in the study of Chapter V.6.

V.2.3 Results of Ellipsoidal Monolayers

V.2.3.1 Results of *RETSCH* Method

The *RETSCH* method was highly reliable when using large particles, as it delivered densely packed monolayers every time regardless of the particles' AR (Figure V-21, photographs of the samples are displayed in the supporting information V.8.4). Just a few particles were pushed onto one another, leaving large areas of truly one-particle-height monolayers.

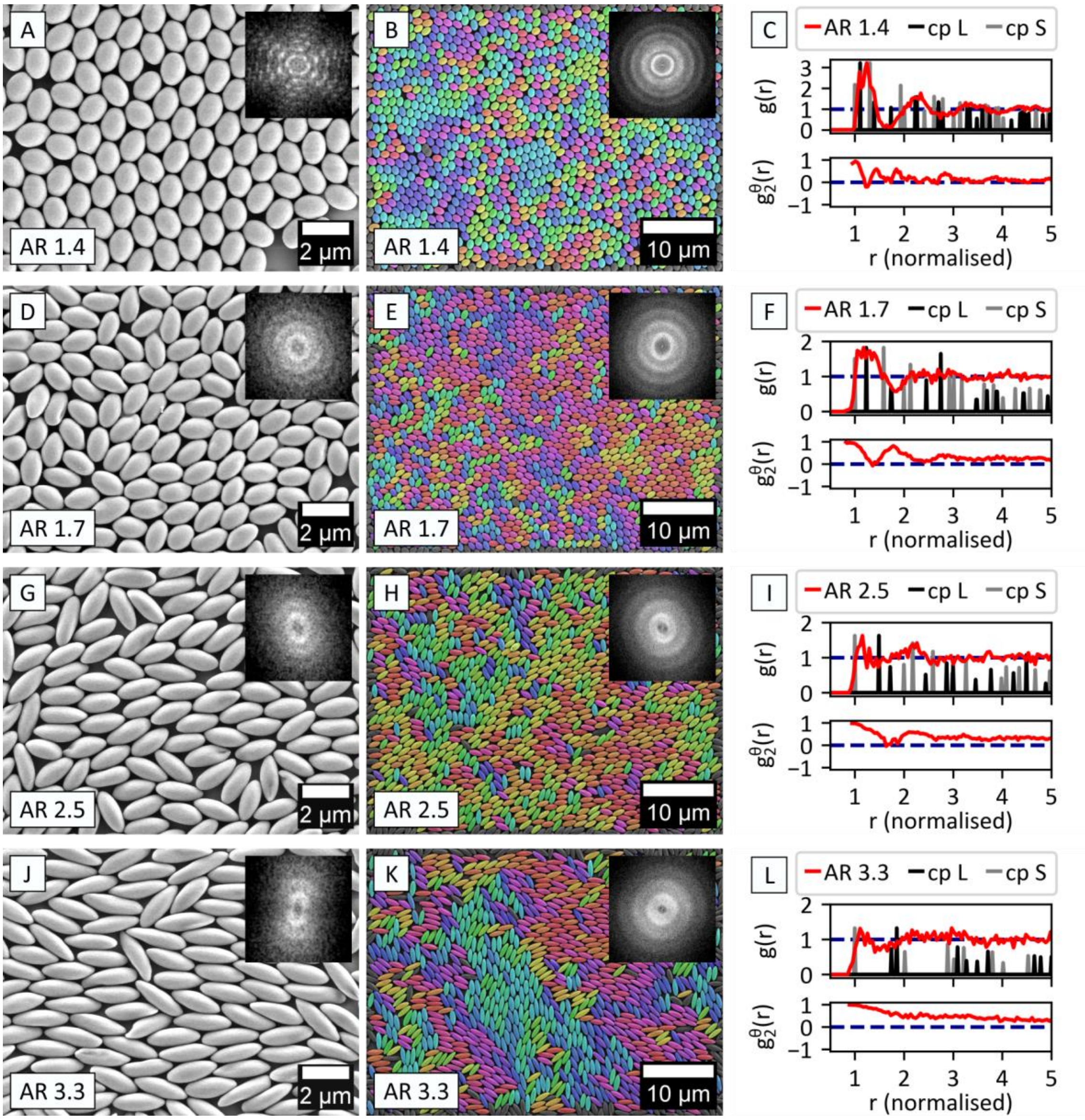


Figure V-21. Monolayers assembled with the RETSCH method on pure MilliQ. First column A, D, G, J: Zoomed-in SEM images on one "domain" of the respective monolayer. The particles showed small areas of order. Second column B, E, H, K: Zoomed-out SEM images with color-coded orientations (same color = same orientation; horizontal: red, vertical: cyan). Insets show the FFTs of the monolayer images (by Fiji/ImageJ). The crystalline domain at image A gave distinct peaks. Third column C, F, I, L: Pair distribution and angular correlation functions of the second column images in comparison to close-packed (cp) test data with same AR and configuration L and S (see Chapter V.1.5). All r normalized to minor diameter (particle width) of SEM image evaluation (see Table V-3). Samples from top to bottom: DB13 15 %_1 AR 1.39, DB07 25 % AR 1.74, DB07 50 % AR 2.45, DB07 75 % AR 3.26.

Distinct crystalline domains of close-packed order and same particle orientations were visible at ellipsoidal monolayers with a near spherical, low AR (1.4 ± 0.1) (Figure V-21 A, zoomed-in on a domain, and B orientation color-coded zoomed-out image). The FFT showed the high crystallinity by distinct peaks (Figure V-21 A inset). The lattice distance/particle width by the FFT peak (0.90 – 1.08 µm, spread due to image resolution) fit well to the SEM results (particle width 1.02 ± 0.03 µm).

The domains of crystalline order became smaller at medium ARs, resulting in blurred FFTs (Figure V-21 D, AR 1.7 ± 0.1, and G, AR 2.5 ± 0.2). However, domains of particles with similar orientation were still visible (although not crystalline) and became larger with high AR (Figure V-21 J, AR 3.3 ± 0.4).

The ordered domains of the samples were not large enough for laser diffraction investigation, thus leaving just blurred rings similar to the FFTs (see supporting information V.8.4).

In the particle monolayer images, S and L configurations next to each other were visible. The transition from S to L was often smooth, so clear grain boundaries were not visible. The smooth transition gave additional peaks in the radial distribution functions (Figure V-21 third column). They were located in-between the peaks of the L or S configuration of the respective AR (calculated with test data, see Chapter V.1.5).

V.2.3.1.1 Results of $g(r)$

The shown radial distribution datasets here were normalized to the minor diameter defined by the general SEM particle evaluation (see particle data in Table V-3). However, this normalization did not work perfectly, as the particles in the monolayers had some spacing in-between and the particle minor diameters were distributions. Thus, the $g(r)$ functions did not fit exactly to the peaks of the simulated data. As there were just small domains and, therefore, no long-range order, the functions were investigated on their first few peaks.

The $g(r)$ of the low AR 1.4 particles showed distinct peaks (Figure V-21 C). Hence, the monolayer had some crystallinity. The peaks seem to fit better to the L configuration (visible in Figure V-21 A). However, the peaks of S configuration were at least also included in the first broad peak of the function.

For the medium AR 1.7, there was a mixture of S and L configuration and lattices in-between. Thus, there was a wide peak over the first peaks of the simulated data. No preferred lattice could be identified.

At AR 2.5 and AR 3.3 the distribution functions converge to value 1 very fast as the next particle center density becomes 1 at any direction. From the SEM images, one would expect a peak at slightly higher values than with S configuration as the particles are not directly touching along S-direction. This peak was present but is minimized because just a maximum of 2 distances can have this distance per particle (compared to 6 at hcp of spheres).

In general, the comparison of the lattices to S and L configuration of ideally packed ellipsoids was not successful for *RETSCH* assembled monolayers with higher AR particles. The monolayer samples showed a too low domain size and the domains' transition was smooth. Thus the $g(r)$ had no distinct peaks.

At low AR particles the $g(r)$ contained distinct peaks and the L configuration was the best fit on this sample.

V.2.3.1.2 Result of $\mathbf{g}_2^{\boldsymbol{\theta}}(\mathbf{r})$

The angular dependency $\mathrm{g}_2^{\theta}(\mathrm{r})$ (Figure V-21 C, F, I, L lower diagrams) showed that particles directly at the minor distance had always a parallel orientation (near 1). This is in line with the idea that the smallest ellipsoids' distance can just occur when particles are in close contact by their short axis.

The overall parallel orientation in the images, evaluated by $\mathrm{g}_2^{\theta}(\mathrm{r})$, seems to grow for large ARs. This is, however, a false friend, as the lowest distance for orthogonal orientation is growing with the AR as well (approx. $r = 1.2$ at AR 1.4 grows to $r = 1.75$ at AR 2.5). The orthogonal contributions added to the graphs at higher values and then mixed with more parallel contributions. Thus, the graphs stood at larger values above 0.

In the graphs for AR 1.4 – 2.5 (Figure V-21 C, F, I), the first dip can be referred to the smallest orthogonal distance for the respective AR.

As there was no long-range crystallinity, the orientation of the particles changed smoothly. Just at AR 1.4 some kind of a crystallinity was present and the orthogonal particles were influencing the diagram at a specific small r, leading to fluctuations (compare to Figure V-13).

V.2.3.2 Results of *VOGEL* Method

For the *VOGEL* method, a spreading agent (ethanol) has to be added to the dispersion. The amount of ethanol to use was dependent on the particle size and on the room/sub-phase temperature (different spreading from winter to summer). Additionally, the application of the dispersion onto the glass ramp by hand was challenging. One has to avoid too much or too few dispersion on the ramp. Therefore, the *VOGEL* method was more difficult to handle.

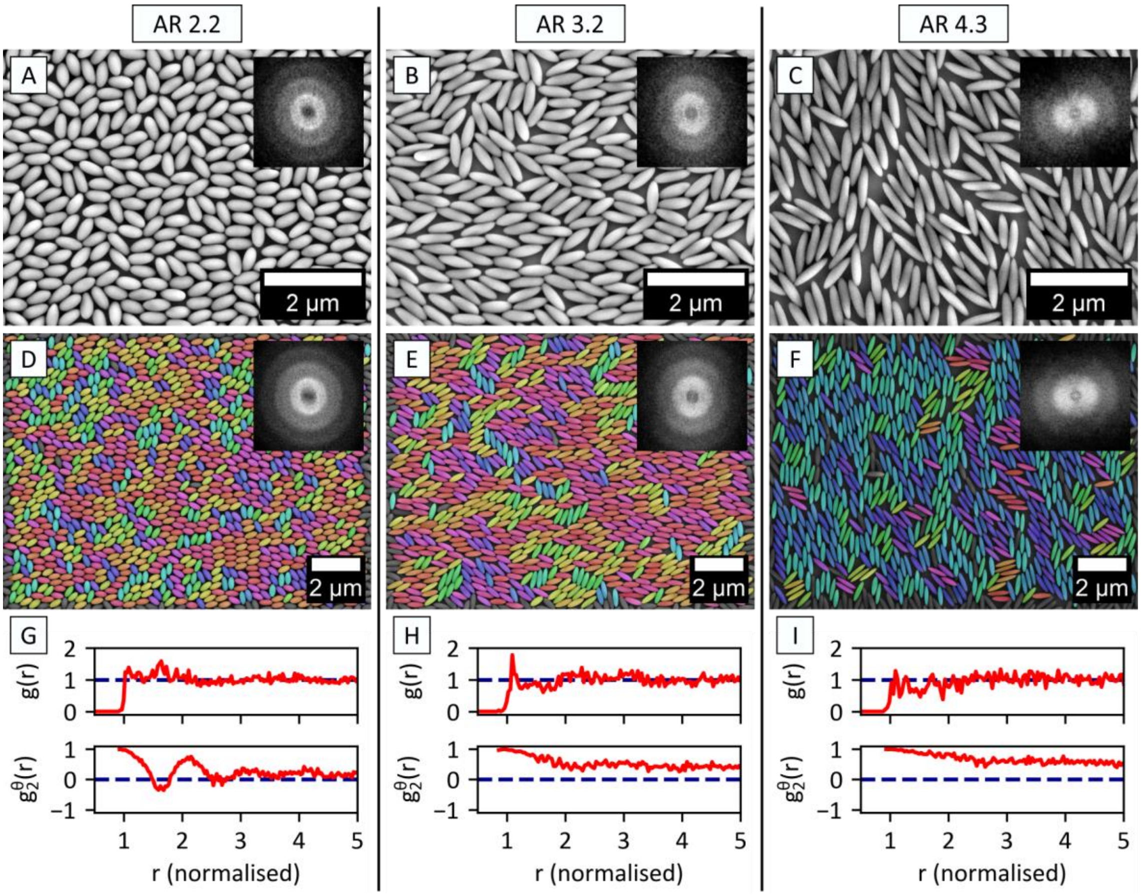


Figure V-22. Monolayers of small particles prepared by the VOGEL method. A – C: Zoomed-in SEM images with FFT insets, D – F: color-coded SEM images with lower magnification showing the overall orientation (with FFT). G – I: Respective $g(r)$ and $g_2^\theta(r)$ plots of D – F. Ellipsoids with AR 2.19 (1st column), AR 3.19 (2nd column), and AR 4.28 (3rd column). Diameter used to normalize $g(r)$ see Table V-3.

With small particles, the *VOGEL* method showed dense but random packed monolayers (Figure V-22 A – C). The monolayers of those ellipsoids were not colorful. With higher AR, the particles tended to pack less densely and the monolayers were not stable on the water surface. The particles were sometimes pushed onto each other (see supporting information V.8.5). A few short-range ordered areas could be found, but long-range order was not visible. The $g(r)$ did converge to 1 very fast without many peaks and is, therefore, not compared to crystalline S and L configurations here (Figure V-22 G – H).

The particle monolayers exhibited a primary orientation (few dominant colors in Figure V-22 D, E and F). The fraction of misoriented particles became lower with higher AR. While at AR 2.2 the angular distribution fluctuated, the parallel orientation of the particles at AR 3 and AR 4 was clear as the function stood all the time above 1 there. At AR 4, the FFT was ellipsoidal which indicates again a primary ellipse orientation. Note that due to the size of the particles, laser diffraction was not possible.

The *VOGEL* method could be applied on big particles as well. However, as the method needed dispersions of several wt% the only AR used here was 1.4 (particles DB13 15 %_2 made by roll-stretching, Table V-3, batch different from the particles used with the *RETSCH* method of Chapter V.2.3.1).

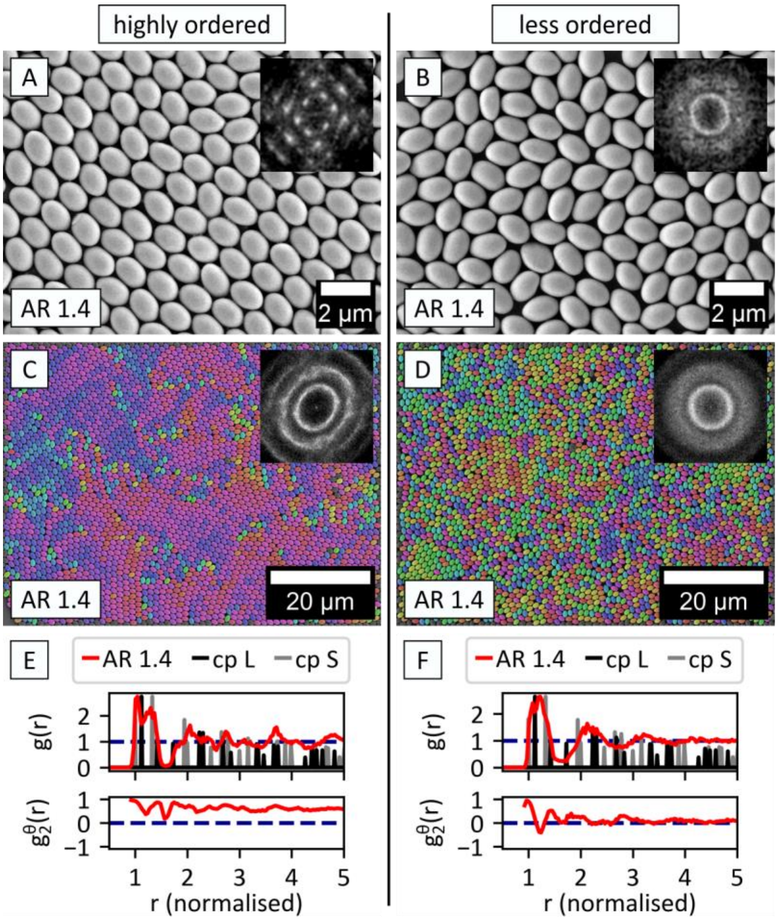


Figure V-23. Big ellipsoidal particles (AR 1.42, DB13 15 %_2) ordered in monolayers. A, C: Some large areas were close-packed and well oriented, while B, D: other monolayers were not oriented, thus giving a blurred FFT. A, C: The highly ordered monolayer was from the application ramp (glass slide) of the VOGEL method, where a monolayer got deposited (after finishing spreading) while moving the lower part of the ramp out of the sub-phase. B, D was made by the normal deposition of a floating monolayer on a glass slide. Both times the sub-phase was 0.5 M HCl. E, F: The radial distribution functions of the color-coded images show peaks that are more distinct at the higher ordered sample. The first peaks were next to the first peaks of L or S configuration of the close-packed test datasets. E: The angular distribution had a higher value (same orientation) for the highly ordered monolayer.

The big AR 1.4 particles displayed some large, nearly crystalline, ordered regions, but also regions with no/low crystallinity (Figure V-23). In the highly-ordered region, the crystal lattice displayed a smeared transition of the lattice (no clear grain boundary/orientation change). Laser diffraction shows similar patterns as the FFT for the highly ordered and the less ordered regions (see supporting information Figure SI V-6). L and S configurations were found next to each other and in the $g(r)$ diagrams. As expected, in highly ordered regions the orientation was parallel ($\mathrm{g}_2^\theta(\mathrm{r}) > 0$) while it was random at low crystalline regions. Thus, a use in optical applications might be possible at highly ordered regions, if the size of those regions could be increased.

V.2.4 Summary of Sub-Chapter V.2

In this sub-chapter, two well-known methods described by and named after *RETSCH* and *VOGEL* for particle monolayer assembly were introduced and could be successfully applied on ellipsoidal particles for the first time. With both methods, it was possible to create densely packed monolayers from ellipsoidal particles.

The *RETSCH* method, which led to best results using big particles (origin from 1.06 µm and 1.15 µm), did always result in dense monolayers, but displayed just small ordered regions. The best order was gained with a small AR (AR 1.39). Large AR ellipsoids (AR 3.26) showed regions with preferred orientations, but did not pack in a crystalline manner.

The *VOGEL* method was applied firstly on ellipsoids with different ARs originating from small 326-nm-particles and secondly on big AR-1.4-particles (1150 nm origin). Some monolayers displayed non-close-packing after scooping them from the water surface (especially with the small particles with high AR), but had a preferred orientation. The big particle monolayers displayed some large crystalline regions and some non-crystalline regions. However, dependencies answering the question of when the orientation/crystallinity of such ellipsoid monolayers increases could not be investigated within this thesis.

As the displayed *RETSCH* and *VOGEL* methods rarely displayed large crystalline areas, the focus of the ellipsoid monolayer preparation shifted to the rubbing method (see Chapter V.4).

V.3. Open-Packed Monolayers – Lattice Transfer Method

The densely packed particle monolayers of Chapter V.2 can be transferred to an open-packed monolayer by using a method developed by Hummel et al. [139].

Within the method, a hydrophobic, polymer-coated glass slide (substrate) is pushed through a densely packed particle monolayer. The hydrophobic particles attach to the substrate and the lattice is opened up. By fixation of the particle positions with a thermal annealing step, using either low T_g particles or low T_g glass coating, it was possible to retain the open-packed lattice after withdrawal of the particle-loaded substrate from the sub-phase.

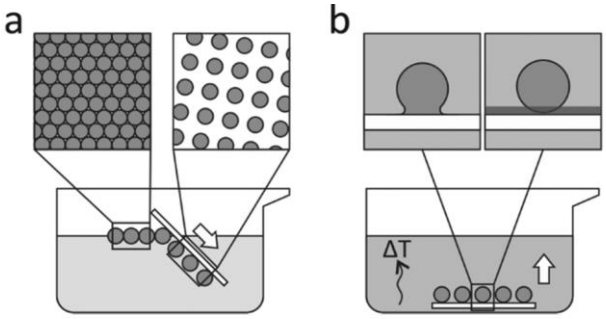


Figure V-24. Schematic description of the lattice transfer process. A: The hcp layer floating on the sub-phase is transferred to open-packed lattice by a polymer coated glass slide. B: The particles were fixed in position either by annealing the particles or the polymer layer prior to withdrawing the glass slide from the sub-phase. Figure reprinted with permission from Hummel et al. [139]. Copyright © 2019 American Chemical Society.

The transfer can be described as a true-one-dimensional lattice stretching giving access to all 2D *BRAVAIS* lattices. The resulting lattice depends on the vector of transfer respective to the pervious monolayer. The theory of lattice transfer can be found in Chapter V.1.2.

V.3.1 Standard Process

Polymer-coated glass substrates (see Chapter V.3.1.1) were pushed upside-down by hand through the monolayers of Chapter V.2, while floating on water. The substrates were held with a tweezer and the coating was pointing towards the particle layer (angle with water approx. -45°, similar to Figure V-24 A). The substrate was moved through the monolayer and the particles got deposited on the substrate. The particle-covered substrate was turned with the tweezer inside the sub-phase so it could lay flat in a glass petri dish, which was sunken inside the assembly container prior to the monolayer preparation. The petri dish was lifted from the assembly container carefully as the substrate needed to be covered with water the whole time. The petri dish was then heated on a heating plate for 15 min to

80 °C. After cooling down, the particle covered substrate was retracted from the water, rinsed with MilliQ water (rinsing with ethanol destroyed the lattice, see supporting information V.8.6) and dried at an angle of 60°.

V.3.1.1 Glass Coating

As the particles used here were all PS particles, a low T_g polymer glass coating was needed for fixation of the particle position. The polymer used was from low T_g co-polymer particles of MMA-nBA (70:30, T_g 50 – 60 °C) made by Marius Schöttle, a colleague within the work group. The particles were dissolved in THF (0.5 g/ml). 120 µl of this solution was spun on plasma-activated glass slides with a 2-step protocol (step1: 500 rpm, 30s; step2: 8000 rpm, 40 s). For a proper polymer-to-glass contact, annealing of the glass at 200 °C for 10 min was crucial. Without this high heating (previously 150 °C was used), the polymer layer detached sometimes and/or showed wrinkles (like from bubbles) after heating for fixation of the particles (Figure V-25).

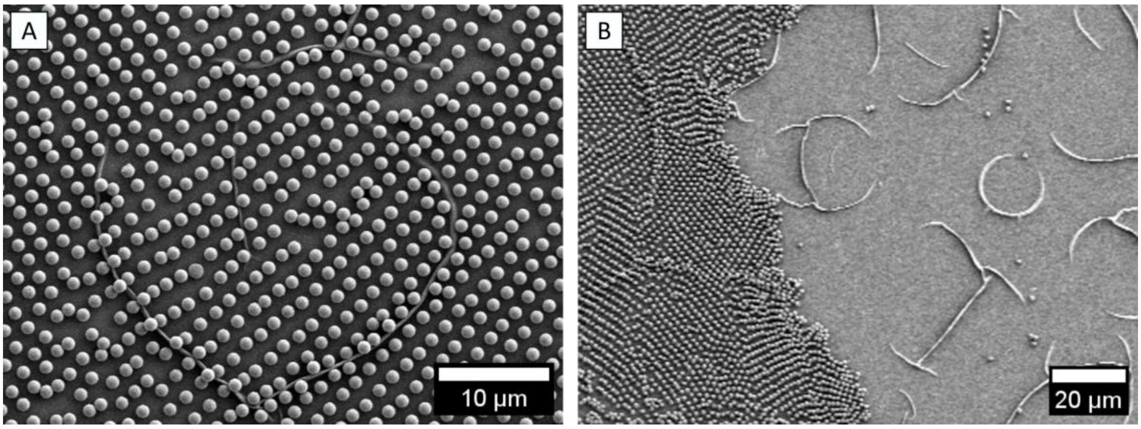


Figure V-25. SEM images of spherical particle monolayer transferred. A: Wrinkles of the fixation layer distort the transferred lattice. B: Zoomed out version shows wrinkles also on areas without particles. Particle type: DB13 0%

The water contact angle for the used polymer coating of the glass was 73° (measurement with 10 µl MilliQ of two spin-coated samples, each 3 positions, after annealing to 150 °C, 10 min). In comparison to the dependency of contact angle and transfer factor found by Hummel et al. [139] a transfer factor of around 3.5 at 73° was expected. Hummel et al. used a computer-aided evaluation of the transfer factor, which could not be used within this thesis. Thus, the transfer factor was measured in Fiji/ImageJ by hand with estimation of previous particle positions (see Figure SI V-8 at supporting information V.8.7). The transfer factor was found to be between 2.3 and 3 for spherical particles. The transfer factor of ellipsoidal particles could not be investigated, as clear lattices were not visible.

V.3.2 General Results

While spherical particles displayed crystalline particle lattices (Figure V-25) with distinct grain boundaries, the transfer of ellipsoidal monolayers resulted mostly in non-crystalline layers (Figure V-26 B, E, H, K).

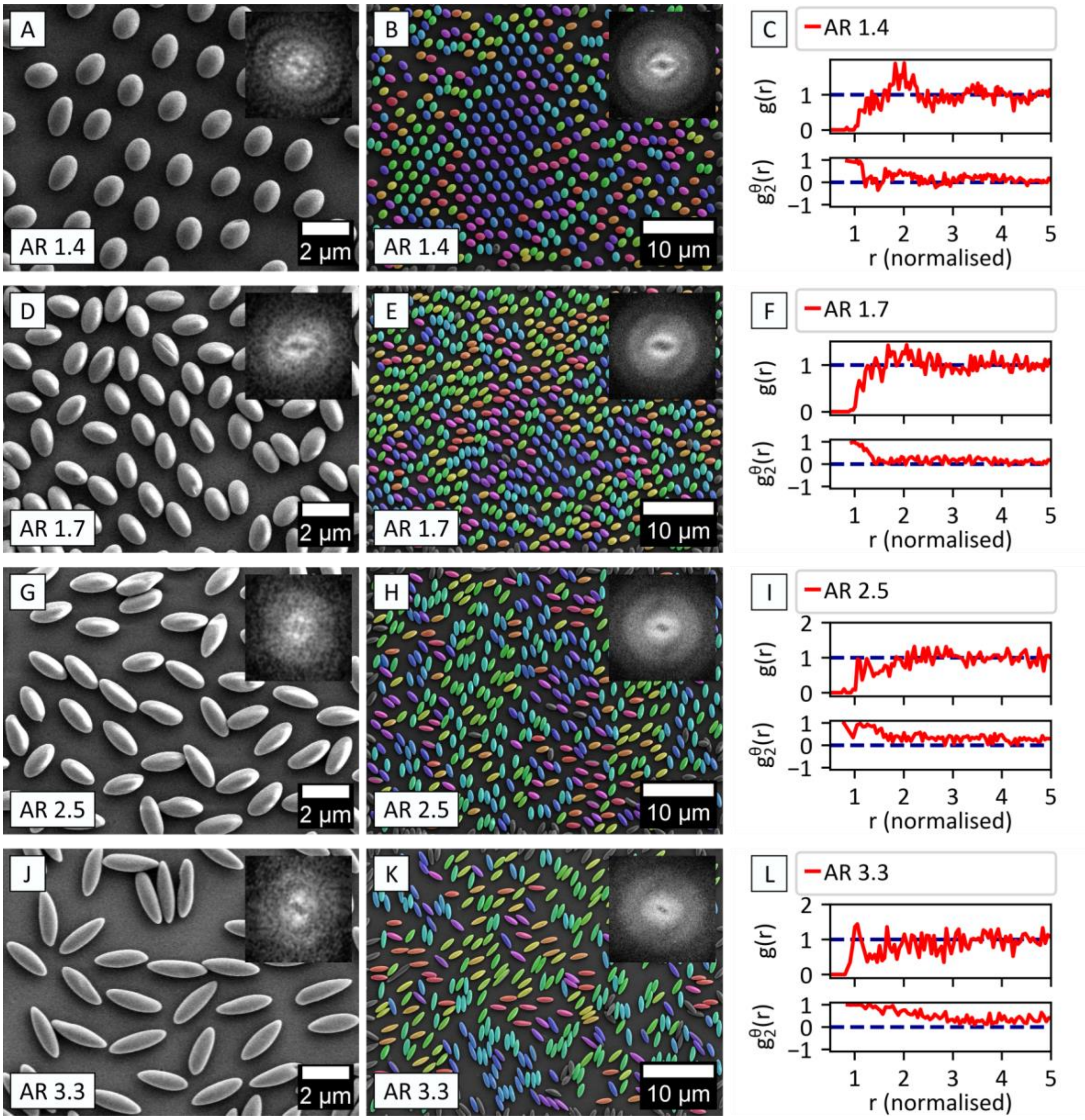


Figure V-26. Lattice transferred monolayers assembled with the RETSCH method on pure MilliQ (from Figure V-21). First column A, D, G, J: Zoomed-in SEM images on one "domain" of the respective monolayer. The particles showed small areas of order. Second column B, E, H, K: Zoomed-out SEM images with color-coded orientations (same color = same orientation; horizontal: red, vertical: cyan). Insets show the FFT of the monolayers images (by Fiji/ImageJ). The crystalline domain at image A gave distinct peaks. Third column C, F, I, L: pair distribution and angular correlation functions of the second column images. All r normalized to minor diameter of particles from SEM image evaluation (see Table V-3). Samples from top to bottom: DB13 15 %_1 AR 1.39, DB07 25 % AR 1.74, DB07 50 % AR 2.45, DB07 75 % AR 3.26.

This finding was not surprising, as the origin monolayer of Chapter V.2.3.1 (*RETSCH* method) did not show large crystalline ordered regions, so the transfer picked up the monolayer particles at different positions. However, small regions of high crystallinity within the origin monolayer, e.g. at low aspect particles (AR 1.4, visible at Figure III-22 A), retained the order also during transfer, thus building small regions with highly ordered lattices (Figure V-26 A).

The laser diffraction patterns of the whole sample showed some broad elongated peaks in the direction of transfer (new diffraction planes after transfer). This was best to be seen on the more ordered sample of the low AR (AR 1.4, DB13 15 %_1) and vanished for the higher AR because of their low order. Accordingly, the scattered-light color of the samples got fainter with higher AR.

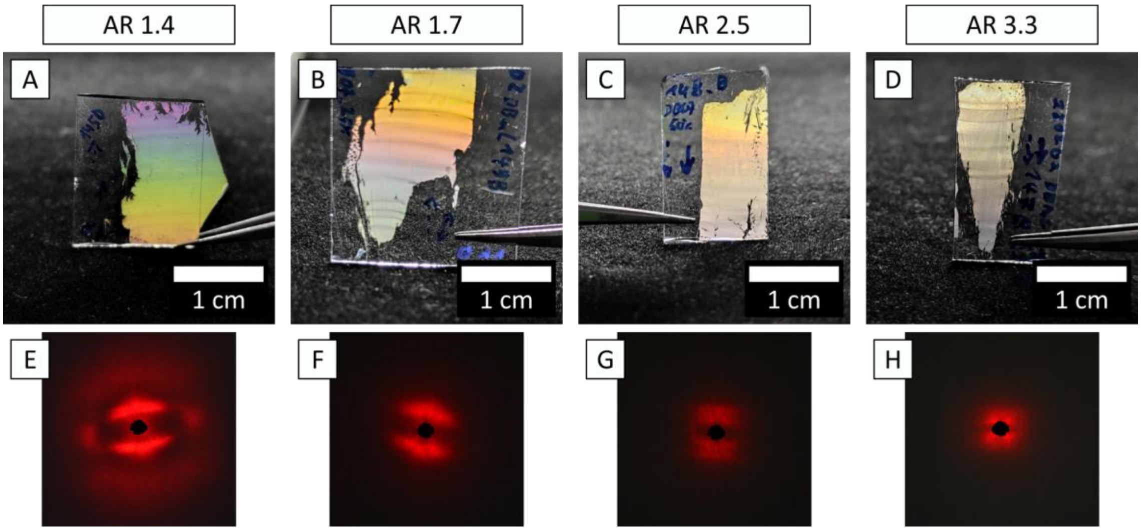


Figure V-27. Transferred monolayers made by the RETSCH method. A – D: Mobile phone pictures with white-light illumination from the back (scale bar estimated) and E – H: the respective laser diffraction patterns (might be slightly turned due to fixation of the sample in the holder of the laser setup). Samples from left to right: DB13 15 %_1 AR 1.39, DB07 25 % AR 1.74, DB07 50 % AR 2.45, DB07 75 % AR 3.26.

Unluckily, it was not possible to refer the distances of the planes (from laser diffraction evaluation) to the transfer factor as the direction of transfer was not fixed and did, therefore, contribute differently to the positions of the particle and hence to the diffraction pattern.

V.3.3 Transfer of Ordered Ellipsoidal Layers

The close-packed monolayer of AR 1.4 particles made with the *VOGEL* method (see Chapter V.2.3.2, particles DB13 15 %_2) was showing crystalline lattice structures in some larger areas. After transfer (Figure V-28 A), the lattice structures were retained and different *BRAVAIS* lattice types could be found on the sample (compare to Hummel et al. [139]). The domains were clearly visible due to drastic changes in the lattice. The dots in the FFT images of the different lattice grains show the high crystallinity (Figure V-28 C and D). Laser diffraction did as well just show lines in the diffraction pattern, as the large spot size of the laser is averaging over multiple grains (Figure V-28 B). Thus, the sample contained many grains with particles in line patterns (Figure V-28 cyan), spanning a rectangular pattern similar to Figure V-28 C (confirmed by zoomed-out image, see supporting information V.8.8, Figure SI V-9 A).

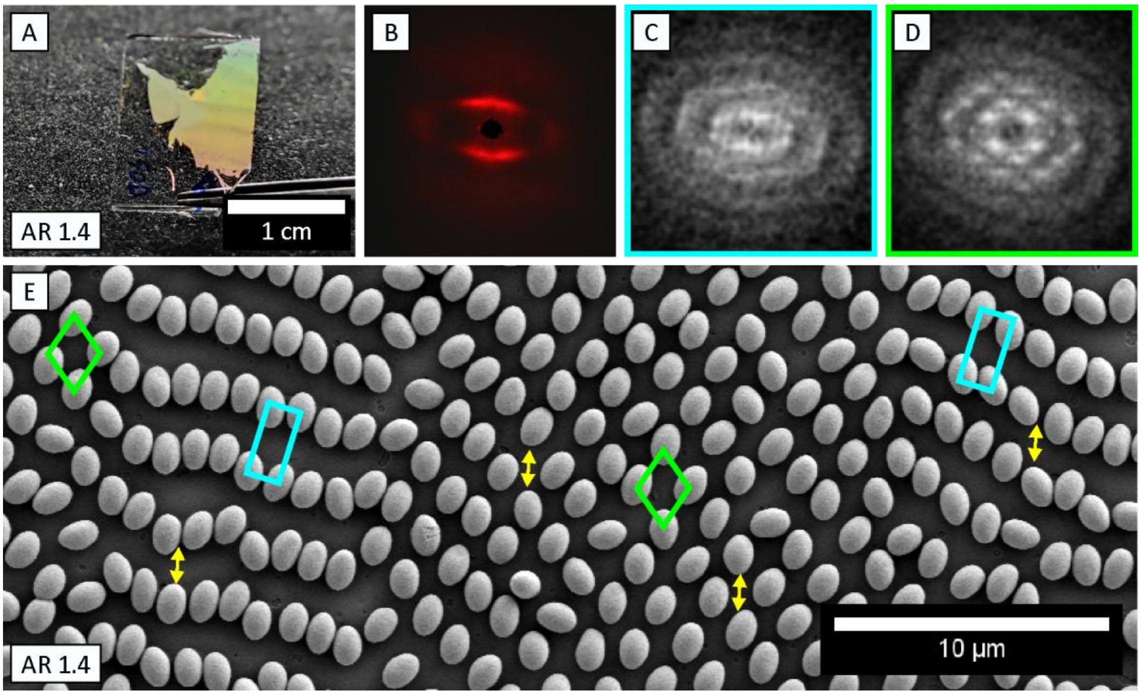


Figure V-28. Transferred monolayer made from a VOGEL-monolayer of AR 1.42 particles (see Chapter V.2.3.2, particles: DB13_15 %_2). A: Mobile phone image with white light illumination from the back, scale bar estimated. B: Laser diffraction pattern of the sample. C, D: FFT images of different grains extracted from the SEM image (E). In the SEM image, individual BRAVAIS lattice types of the different grains are shown as rectangular (cyan) and oblique (green). Yellow arrows indicate the transfer direction.

Due to the small amount of particles in the grain size, pair correlation functions were not reliable (see supporting information Figure SI V-9 B and C). The orientation correlation functions and the orientation histograms showed a preferred orientation in the direction of transfer. This was regardless of the grain (see supporting information Figure SI V-9 D – F).

V.3.4 Preferred Orientation of Ellipsoids after Transfer

During the transfer experiments, it was recognized that the transferred ellipsoids in the monolayer all had a preferred orientation in the direction of transfer. This preferred orientation could have evolved during the formation of the monolayer (e.g. due to assembly forces), during transfer prior fixation (e.g. due to water sheer flow) or when retrieving the sample from the water after fixation (e.g. due to capillary interactions). As the monolayers were taken up in the direction of deposition (analogue to scheme in Figure V-17) and the take up direction was noted down on the sample, the SEM images were all oriented in the direction of take up (vertical at the images, 90°), giving the possibility to re-evaluate the cp monolayers in terms of their particle orientation.

The histograms of the particles in the cp monolayers showed preferred orientations, but surprisingly at individual angles (Figure V-29 A – D). This finding counteracts the idea of having the same orientation after the monolayer formation. The monolayer had, therefore, an unknown preferred orientation before the transfer step. After transfer, a preferred orientation in the direction of transfer was visible for all samples (Figure V-29 E – H).

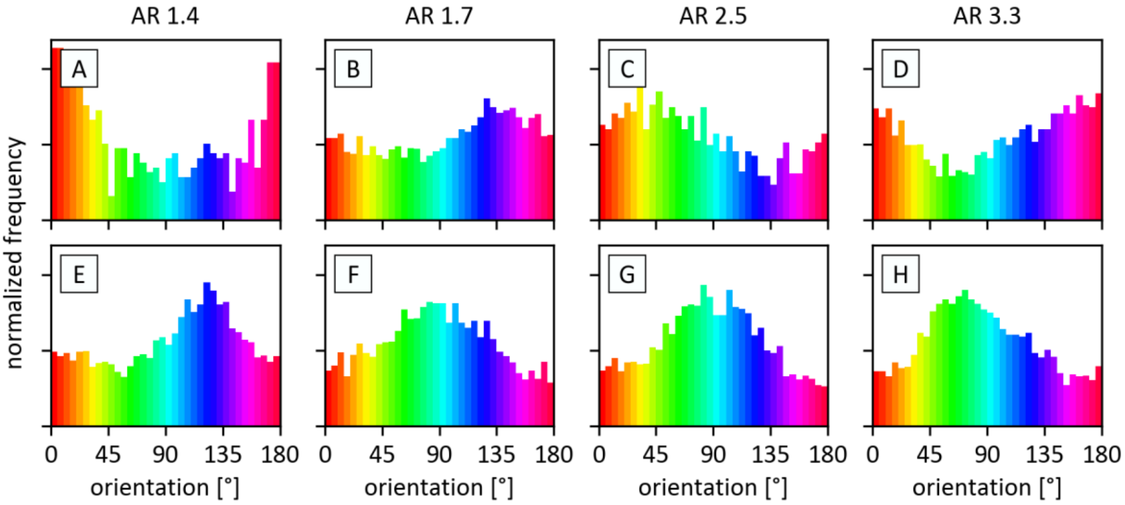


Figure V-29. Orientation-histograms of ellipsoids in monolayers. A – D: Data from dense-packed monolayers made by the RETSCH method. E – H: respective transferred monolayers. For all monolayers, there is a preferred orientation visible. Direction of deposition and transfer on images at 90° ± 20°. For the histograms, 1000 – 10000 particles were evaluated per sample (except at AR 1.4, plot A with just 632 particles). Samples from left to right: DB13 15 %_1 AR 1.39, DB07 25 % AR 1.74, DB07 50 % AR 2.45, DB07 75 % AR 3.26.

Although, these findings could not be further evaluated within this thesis, they might be interesting as they indicate that the transferred ellipsoids turn during transfer before fixation (while diving) or after fixation when the sample is retrieved from the water.

V.3.5 Summary of Sub-Chapter V.3

In this sub-chapter the transfer method of Hummel et al. [139] was successfully applied first time on ellipsoidal particle monolayers.

The resulting transferred ellipsoid monolayers (big ellipsoids AR 1.4 – 3.3 used) did not show large crystallinity due to low crystallinity of the original particle monolayer. With more crystalline samples, the resulting monolayer retained the crystallinity, and grains with differently oriented lattices resulted in very different transferred lattices.

After all, it could be shown that the lattice transfer method is capable to open up dense-packed monolayers of anisotropic particles. The transfer vector direction was the same at different positions. The transfer factor could not be extracted exactly from the particle monolayers due to unknown orientation prior to transfer (estimated factor $T = 2.3 - 3$).

The transferred ellipsoids all showed a preferred orientation in the direction of transfer. As there was already a preferred orientation of the original monolayers, which then changed to be in the direction of transfer, it might be worth to investigate the turning behavior further. Especially until now, it is not clear at which point the particles turn to the transfer direction and if this behavior can be controlled.

However, the transfer method would be more beneficial for optical devices, if the to-be-transferred monolayers show crystallinity. Just then, the resulting transfer lattice has a potential to show crystallinity and well-defined colors (diffraction patterns) as well.

V.4. Large-Area Oriented Monolayers by Dry Rubbing

In the previous sub-chapters, the monolayers were prepared by self-assembly and exploiting hydrodynamic forces. However, particle monolayers can be made as well by forced assemblies. Thus, in this sub-chapter the rubbing method of dry powder particles on PDMS, which was published by Park et al. in 2014 [151], is used to mechanically force particles into monolayer assemblies.

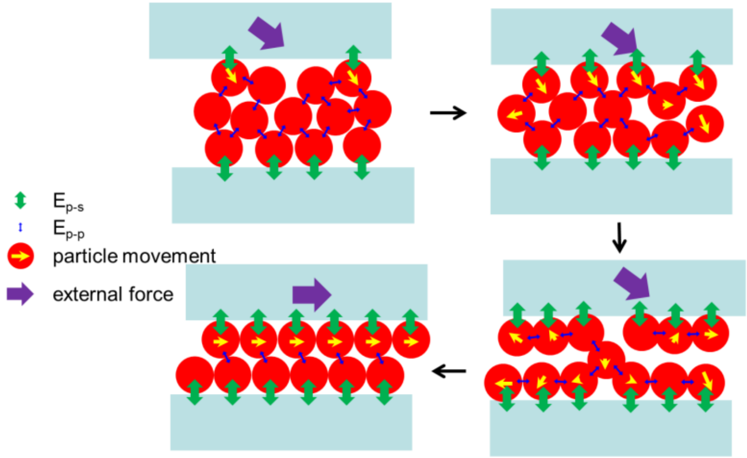


Figure V-30. Schematic drawing of the rubbing process. The dry particle powder is sandwiched between two PDMS sheets. Through movement of the upper sheet, the particles were dispersed and ordered themselves in dense layers at the PDMS surfaces. Reprinted with permission from the supporting information of Park et al. [151] *Copyright © WILEY-VCH Verlag GmbH & Co. KGaA, 69469 Weinheim, Germany, 2014.*

The rubbing method is a very simple, easy-to-apply method and does lead to extraordinary large domains of hexagonally close-packed particle monolayers when using monodisperse spherical particles. For performing the rubbing method, particles are sandwiched between two elastomeric surfaces. The upper surface (applicator) is then moved over the substrate to spread the particles. The method is explained by Park et al. as the rolling of particles on top of an elastomeric, semi-sticking surface (Figure V-30). While the particles roll along in the rubbing direction, the particles interact with the sticky surface. As particles touching each other, they form small regions of close-packed particle assemblies, which need more energy to be moved along. The close-packed layer grows by adding more and more particles. Voids are filled up by add-on particles above the monolayer during rubbing. Excess particles on top of the monolayer are rubbed away as the particle-substrate interaction is stronger than particle-particle attraction, resulting in a particle monolayer. Domains of different orientations are not visible as the preferred domain orientation has a particle contact within rubbing direction due to the rolling nature of the particle assembly. Further, additional rubbing can heal line defects.

Park et al. tested particles of 170 nm to 5 µm and found that the best hexagonal assembly was gained when the particles could roll over the substrate surface. The rolling behavior for differently sized particles is thereby depended on different pressures and speeds. They found that small particles are best assembled with high pressure (e.g. 1.75 kg/cm²) and low rolling speeds (2.1 mm/s), while large particles in the µm-size range are assembled with higher speed (4 mm/s) and lower pressure (0.3 kg/cm²). Note that the values shown here were estimated from a figure in the paper of Park et al. [151]

The optimum for 1 µm particle rubbing, found by Park et al., was 3 mm/s and 2 kg/cm². The adhesion energy of the substrate was monitored by them as well (tack test) and it was found that the assembly failed at low (< 0.01 gf mm, sliding across) or very high (> 10 gf mm, restricted translation) adhesion energies. PDMS made by standard recipe (Sylgard 184, mixing ratio 10:1, 0.05 gf mm) is within the region of optimal adhesion energy and, therefore, well suited to be used for the rubbing method.

Park et al. rubbed the dry particle powder in between two PDMS covered glass substrates with a “reeling machine moving [the upper slide] laterally” [151] with fixed loads. For optimal packing of 1 µm particles the back-and-forth rubbing was done 10 times at 3.5 mm/s and 1.25 kg/cm². Another researcher (Nathan ten Napel [164]), who used the rubbing method within his PhD thesis, constructed a special robotic rubbing device. His applicator for the particles was a PDMS cylinder resulting in a defined contact area.

While literature shows the rubbing of spherical particles [151, 165] or near-spherical supra-particles [164], nothing has been published on the rubbing of ellipsoidal particles yet.

The research of this sub-chapter was done with the help of my student assistant Justus Sanders in 2021/22.

V.4.1 Setup and Standard Process

I developed an easy but defined setup for the rubbing process. The main focus was to increase reproducibility and to start screening the multiple parameters influencing the rubbing process.

The process was basically the same as used by Park et al. in 2014 [151]. Dried polymeric particles were placed onto a PDMS substrate and rubbed with an applicator. Excess particles could be removed in several ways (Chapter V.4.1.1). The used substrates are described in Chapter V.4.1.2 and the used particles in Chapter V.4.1.3.

The rubbing of particles was firstly done by hand using a PDMS piece moved by a finger on top of a particle powder. Later on, the applicator was a PDMS foil held by a 3D printed device. The applicators are described in Chapter V.4.1.4.

For spherical particles, the applicator was moved in different directions until the particles covered the whole surface and the applicator could move easily. The rolling particles acted as a lubricant. After some streaks, a hexagonal close-packed monolayer formed. To enhance the hcp order, the applicator was moved just back and forth (no circles nor curves) following directions with 60° angles in between (along the hexagonal grid lines).

For ellipsoidal particles, the applicator was moved just back and forth or/and in very low angles of 10°. With low AR particles (see Chapter V.4.4.2), strokes in a close to 60° fashion were as well suitable for enhancing the order (more defined laser diffraction pattern). For medium and large ARs this was, however, not the case.

V.4.1.1 Removing Excess Particles

The rubbed close-packed monolayers sometimes had multiple excess particles on top (Figure V-31). Due to the large particle-substrate-interaction it was possible to remove the excess particles by further rubbing (e.g. with a clean applicator), by blowing gas (used by Park et al. [151] and Koh et al. [140]), by applying liquids, or by pressing something with a lower particle-substrate interaction compared to the rubbing substrate on top of the layer (clean glass or PDMS [140]). When using liquids to wash away the particles, one has to avoid contacting/rubbing the monolayer, as the order is completely destroyed when rubbed in the wet state due to a lowered particle-substrate interaction.

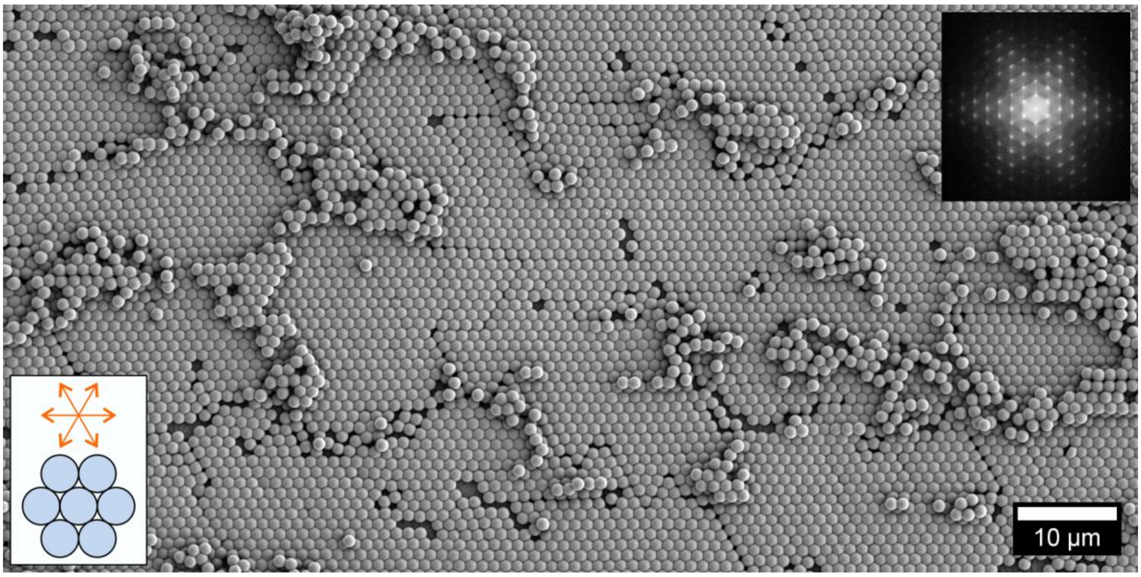


Figure V-31. Rubbed spherical particles with extraordinary long-range ordering exceeding crystal grains/line defects. The layer was made by rubbing in a 60° fashion (orange arrows); however, many excess particles are still on top of the monolayer. These can be removed to a large extend by further rubbing. Inset of FFT shows the very high crystallinity of the rubbed monolayer (first grid plane at 1024 nm). Particles: DB13 0 %.

V.4.1.2 Substrates of PDMS

PDMS (Sylgard 184, DOW Chemicals) in a mixing ratio of 10 : 1 w/w (monomer : initiator) was casted after degassing (40 mbar, 30 min) on a glass plate (20 cm × 30 cm) to a defined wet height of 1 or 1.5 mm with a doctor blade by hand. After curing at 65 °C, substrates in the size of 4 cm × 10 cm for application by hand or 1.5 – 2 cm × 10 cm for application with defined weight were cut out from the PDMS foil. The PDMS substrates were mounted on individual glass slides (for stability and cleaning reasons) prior to rubbing. After rubbing, the PDMS foils with the particle monolayer could be stored without the glass slide and could be used for stretching experiments (Chapter V.5).

V.4.1.3 Particles used in Rubbing Studies

Within this sub-chapter, the used particles come from synthesized polystyrene particles (see Chapter II.1.3.3, DB07 and DB13) in the size of 1 µm (Table V-4). The benefit of using the µm-sized particles was the possibility to investigate the monolayers in an optical laser-scanning microscope (LSM) and to perform laser diffraction. Additionally, a lower pressure was needed for rubbing the particles to highly ordered monolayers (compare to Park et al. [151]). The ellipsoidal particles were prepared using the oven-stretching method of Chapter II.2.1 or the roll-stretching method of Chapter III.2. Both times, the foils for particle stretching were produced by the continuous foil doctor blade (Chapter III.1). The desired particles in dispersion were dried into aluminium pans at elevated temperatures (50 °C). The dense 3D assembled particles were fractured with a spatula to receive a particle "powder" that can be used in the experiments.

Table V-4. Dimensions of the particles used for rubbing experiments in this chapter. The particles come from the monodisperse particle batches DB07 and DB13 (synthesized by dispersion polymerization).

particles		preparation	length [µm]	width [µm]	AR
DB07	0 %	disp. pol.	1.06 ± 0.03		1.02 ± 0.01
	15 %	oven-stretch	1.33 ± 0.06	0.94 ± 0.03	1.43 ± 0.07
	25 %	oven-stretch	1.53 ± 0.08	0.88 ± 0.04	1.74 ± 0.13
	50 %	oven-stretch	1.97 ± 0.12	0.80 ± 0.05	2.45 ± 0.24
	75 %	oven-stretch	2.39 ± 0.16	0.74 ± 0.05	3.26 ± 0.36
DB13	0 %	disp. pol.	1.15 ± 0.02		1.02 ± 0.01
	15 %_2	roll-stretch	1.47 ± 0.06	1.03 ± 0.03	1.42 ± 0.07
	50 %	roll-stretch	1.94 ± 0.09	0.90 ± 0.03	2.17 ± 0.15

V.4.1.4 Applicators

The first applicators used with the rubbing method were simple PDMS foil pieces or thick PDMS pieces of any shape casted into a PE petri dish from the same PDMS mixture (10 : 1 w/w). The size was chosen to cover at least the fingertip with which the applicator was moved.

Later on, 3D printed devices were used. Taking up the idea of the defined contact area of a cylinder which was applied in the device from Nathan ten Napel [164], the 3D printed device was designed that a thin PDMS foil was passed around a half-cylinder. A small part of this half-cylinder was meant to be the only area where the applicator got into contact with the substrate. The contact area was small (ideally a line), so the applied force was not distributed to a large area. This was especially important for rubbing with defined pressure (Chapter V.4.4.3.1). The PDMS foil was clamped into the devices or fixed by scotch tape. The applicator foil of maximum 2 cm width exceeded the substrate width so that the contact area was defined by the substrate width.

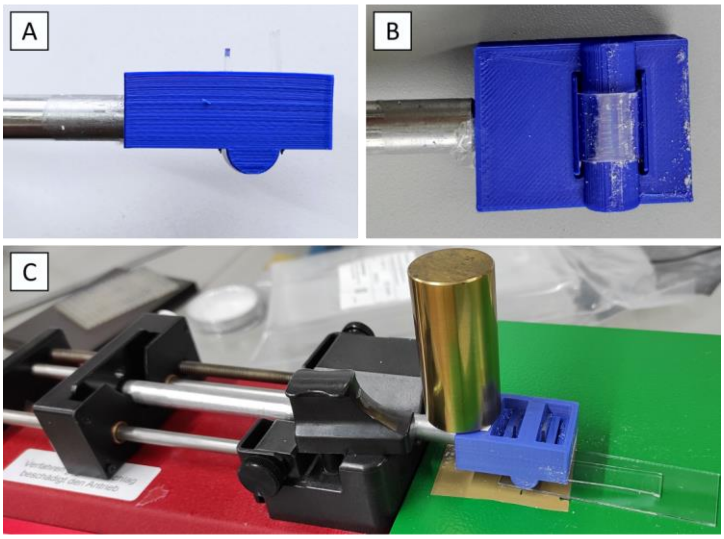


Figure V-32. First design of the applicator for rubbing. A: Side view and B: bottom view with the PDMS applicator stripe. C: A syringe pump with maximum speed of 5 cm/min moved the applicator over the sample. On the applicator a weight could be mounted (brass cylinder, in image shifted to give view to the top of applicator).

The first design (device 1, Figure V-32) contained a stick for pushing and pulling the applicator by a syringe pump (Figure V-32 C). The substrate was taped to a lifting platform. The speed of the syringe pump (Model NE-1000 Multi-Phaser, New Era Pump Systems) was set to the maximum rate (5.1 cm/min). The half-cylinder for rubbing had a radius of 5 mm.

Regular monolayers could be produced. However, this design had the draw back that it needed a second support point of the pushing rod (at syringe pump), so the force of the weight, which was mounted on top, was not fully transported to the substrate. Additionally, the speed of the syringe pump was low (compared to Park et al. [151] and compared to the method of rubbing the monolayer by hand).

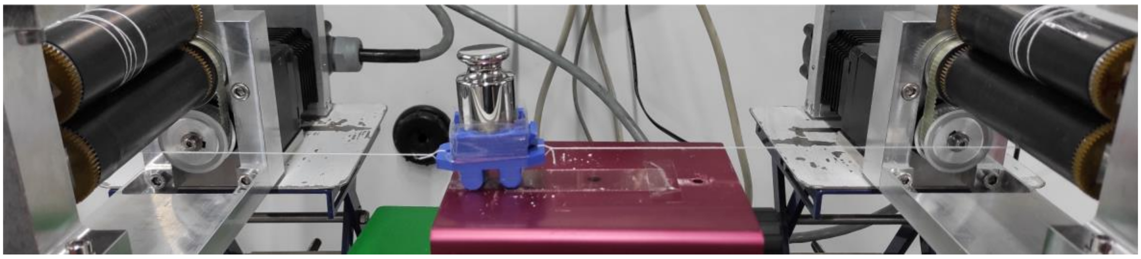


Figure V-33. Second design of an applicator. The wagon-like applicator used the roll-stretch device as a pulley and had two rubbing contact points with the sample. A weight was mounted on the wagon to test the influence of different pressures on the assembly.

The second design (device 2, Figure V-33) used the roll-stretching device to pull a wagon-like applicator (8.67 g) over a substrate. The substrate was fixed by scotch tape to a lifting platform (Figure V-33 pink). The wagon was attached to two strings pulling the wagon to one or the other side. The wagon had two half-cylinders (radius 5 mm) in contact with the sample. The applied speeds of the roll-stretching device were 0.44 cm/s and 1.79 cm/s (26.6 cm/min and 107.4 cm/min, rpm on device: 2.5 and 10). With the second design of the applicator, a problem occurred in the beginning of the rubbing process. When the substrate was not yet covered with particles, the wagon hopped over the PDMS substrate. Another problem was that the wagon was not guided and sway perpendicularly to the moving direction, while moving it across the substrate. These problems were countered by guiding the wagon by hand, carefully paying attention to not giving additional pressure to the wagon. The wagon was designed to add a weight on it whose force should be equally distributed to the front and back half-cylinder.

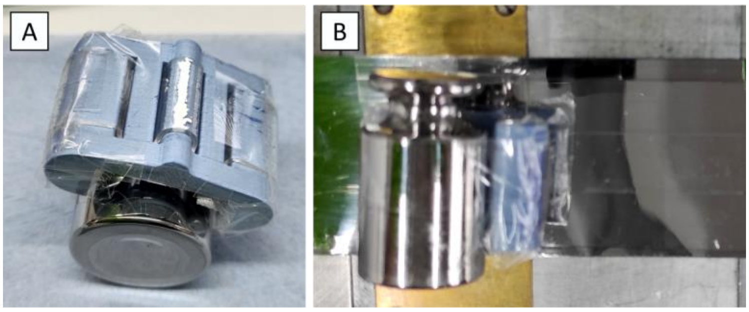


Figure V-34 Final design of the rubbing device. The contact area was minimized to a half-cylinder with a radius of 2.5 mm. A: Bottom view and B: top view with additional weight taped to the applicator. This applicator was used for rubbing by hand or on the belt of the continuous doctor blade device (see B).

With the last design of the applicator device (device 3, Figure V-34), the half-cylinder radius was reduced to 2.5 mm. The device was designed to use the belt of the continuous doctor blade (Chapter III.1.2.1) to move the substrate back and forth, while the applicator is set on the substrate and held in position by hand without putting additional weight on it. The applicator had one contact line (approx. width 1.5 mm). The weight was added by scotch tape to the device. The applied speed was up to 3 cm/s (180 cm/min).

A mechanical fixation in space was not done since then the force during rubbing would not come from the weight but from the uneven thickness of the PDMS or the fixation as well.

V.4.2 Common Defects of Rubbed Monolayers

The rubbed monolayers had large areas of close-packed particles. Nevertheless, several defects occurred (Figure V-35).

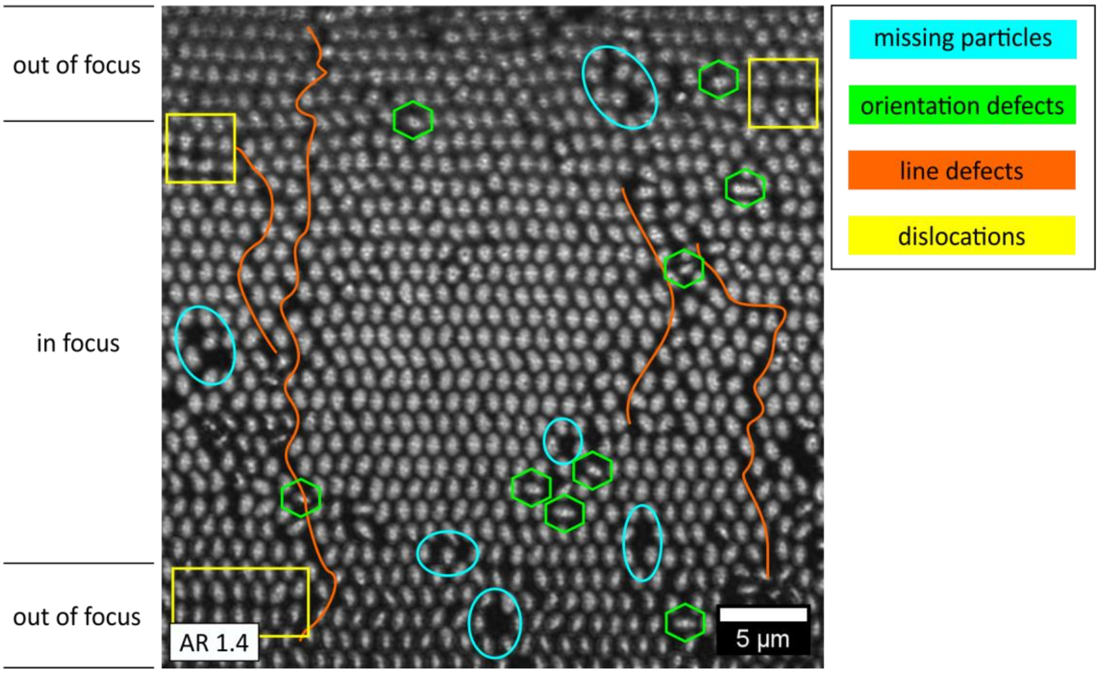


Figure V-35. Examples for different ellipsoidal particle monolayer defects. Dislocations (yellow), point (cyan and green) and line defects (orange) are visible in the image. The used 2D-imaging mode of an optical laser-scanning microscope (LSM2D mode) led to areas in the image being out-of-focus, complicating the evaluation (see next Chapter V.4.3). Sample: AR 1.42 (DB13 15 %_2) rubbed horizontally on PDMS.

The main defects were holes within the monolayer due to missing particles. Rubbing additional particles on the monolayer helped to minimize the occurrence of this defect. However, too many particles led to particles on top. The monolayers depicted domain-like regions separated by line defects (grain border) that origin from holes or turned particles (orientation defect). From domain to domain the orientation of the particles and the lattice did not shift. This was a benefit in contrast to floating particle monolayer domains. Another frequently encountered defect was the dislocation between two adjacent rows leading to slightly open-packed areas (mostly rectangular lattices with long axis particle contact). With higher AR, the particle domains merged into each other without a distinct domain border.

V.4.3 Evaluation Limits

The rubbed particles were imaged while still on the non-conductive PDMS using an optical laser-scanning microscope (LSM, Olympus LEXT). The evaluation of the LSM images was done with the python program of Chapter III.3 by just recognizing the orientation and position of the respective particles (particle dimensions were not evaluated due to low resolution). The LSM was used with the laser-scanning mode in 2D (LSM2D) as the wide-field optical mode gave problems with particles on top (very bright spots). The 3D mode could also not be used for evaluation as the PDMS below the particles was imaged then as bright as the particles, making it impossible for the python program to separate the particles. However, at LSM2D, tilted substrates led to unfocused areas in the image (Figure V-35 top and bottom). The in-focus particles were bright and easy to detect while unfocused particles were darker and blurred, thus, they often gave problems in detection. Particles resting on top of the monolayer left a blurry shade as well. During evaluation, blurred particles were mostly combined to larger particles so the python program filtered them out. Black voids in the overlay were the results. For evaluation of the LSM images in the python program the *global threshold* method (see Chapter III.3.1) was used as the normal *rank gradient* method (as implemented for SEM image evaluation) led to more undetected particles.

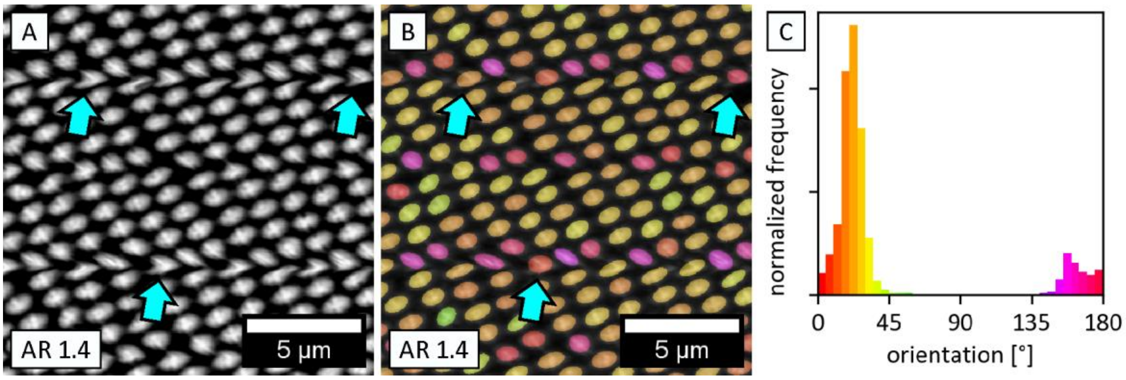


Figure V-36. A: Zoom-in on a LSM2D image (contrast enhanced) and B: orientation-colorized version with line-scanning artefacts (blue arrows indicate same spots). The line-like distortions are moving from top to bottom during scanning. C: orientation histogram of an area double the size of B (of the zoomed out image of B). Sample: AR 1.42 (DB13 15 %_2) rubbed diagonally on PDMS.

With the LSM2D mode, some images depicted line-scanning artefacts resulting in distorted particles (Figure V-36 A and B). The distorted particles were recorded as particles with different orientation (especially when particles were diagonally oriented to the scanning direction). Although it would be best to correct these lines, the here used python program was not designed for that. Therefore, the angular evaluations include those scanning artefacts (Figure V-36, pink particles, angle 170°). The artefacts should not change the particle correlation function, as the position of the particle should stay more or less the same.

V.4.4 Rubbing results

To recapitulate the rubbing method as described in literature, firstly, some spherical particles were rubbed on PDMS substrates. After successful rubbing (Chapter V.4.4.1), anisotropic particles of low AR were used (V.4.4.2) and finally, particles with higher AR (V.4.4.3).

V.4.4.1 Rubbed Spherical Particle Monolayers

The spherical particle monolayer rubbing by hand was highly reproducible. Multiple particle monolayers with extraordinary large domains were made with the described method (see Chapter V.4.1) of 60° rubbing directions (Figure V-37).

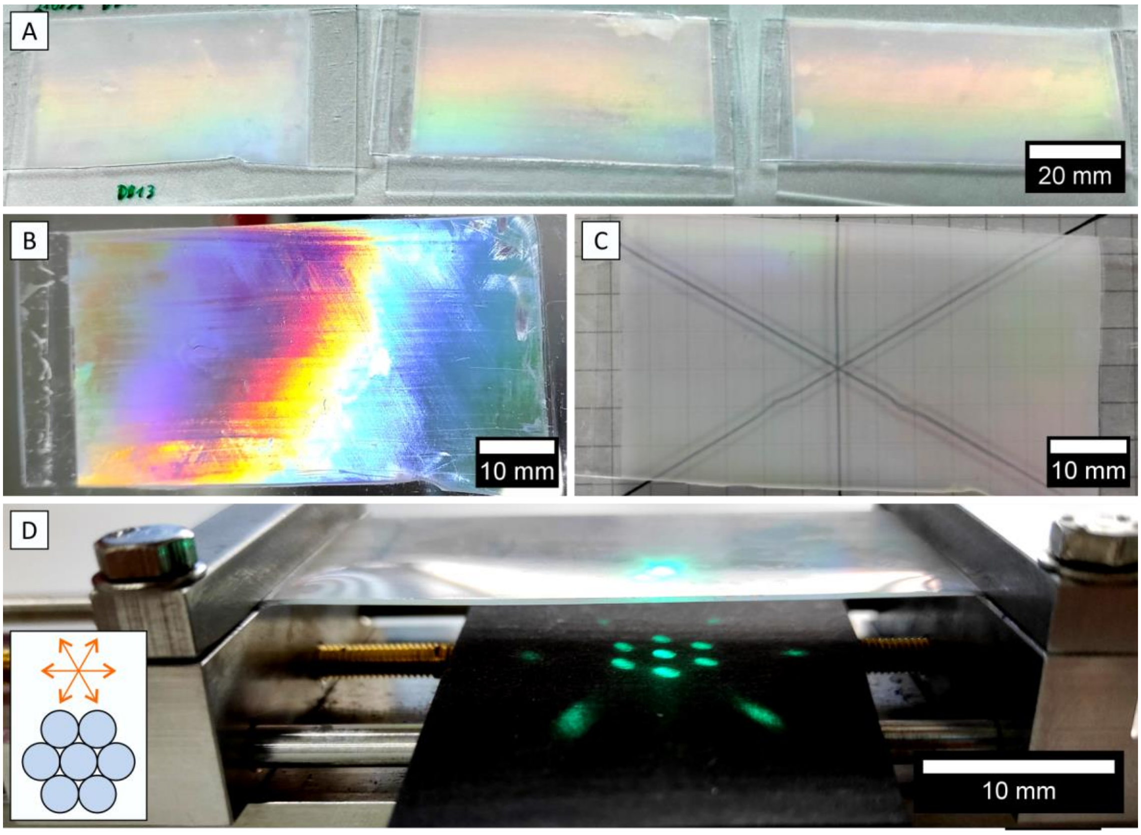


Figure V-37. Spherical particles (DB13 0%) rubbed on A: multiple large PDMS substrates to show the capability of the method. B: Colorful reflexes of the monolayer by illumination with white light from the back. Streaks from rubbing in 60°-fashion were visible. C: Birefringence of the rubbed particle monolayer. D: Laser diffraction showed distinct crystalline peaks with same orientation over the whole surface. Inset indicates the rubbing directions (orange arrows) and the microscopic particle lattice (blue circles). First sample of A used in B – D.

The spherical samples showed colorful reflexes and birefringence (Figure V-37 B and C). The microscope images (Figure V-31) and laser diffraction patterns (e.g. by laser pointer from presenter stick, Figure V-37 D) showed a hexagonal lattice over the whole sample with no orientation changes. The hcp layer was oriented with particle-particle contact in rubbing direction (angle α = 0°).

V.4.4.2 Anisotropic Particles of Low AR 1.4

Small AR particles (AR 1.4, particle types DB07 15 % and DB13 15 %) were easily assembled by the rubbing method into close-packed crystalline lattices. Similar to the spherical particles, very large areas of crystalline particle monolayers were accessible. The preferred orientation of the rubbed particles was perpendicular (90°) to the rubbing direction. Thus, the particle orientation changed with the orientation of the rubbing direction (Figure V-38).

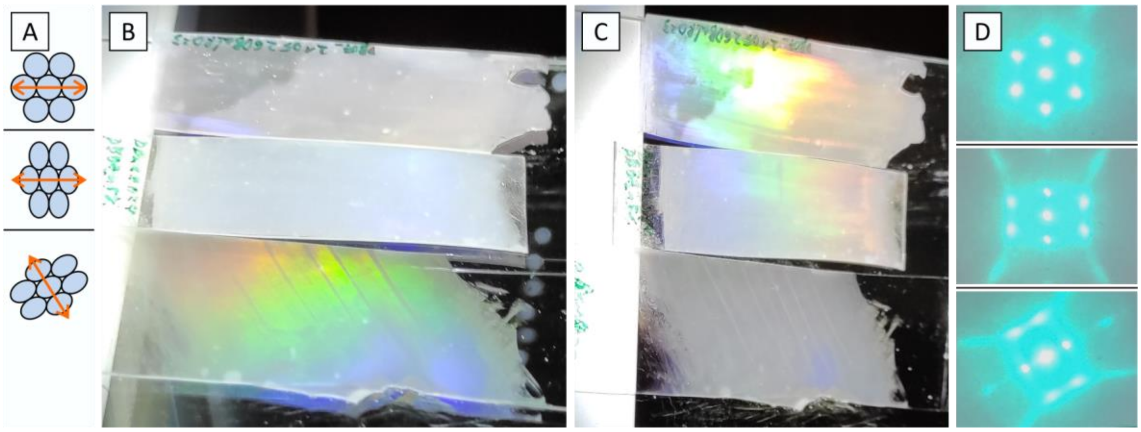


Figure V-38. Comparison of horizontal monolayer rub of spherical particles (DB07 0%), horizontal rub of ellipsoidal particles (DB07 15 %, AR 1.43), and a 60°-rub of ellipsoidal particles (as well DB07 15 %). A: Schemes showing the microscopic arrangement of the particles and the rubbing direction (orange arrows). B, C: monolayers were imaged from different illumination angles (no scale bars on the images as these are perspective views, sample size comparable to Figure V-37). D: The qualitative laser diffraction was performed by the laser pointer of a presenter stick.

The more detailed monolayer investigation (Figure V-39) confirmed that the orientation of the particles was almost perfectly perpendicular to the rubbing direction (tested at multiple samples with different orientation of rubbing, Figure V-39 A – C, J – L and angular correlation function close to 1). The high crystallinity of the particle layers was proven by FFT, laser diffraction and the $g(r)$ calculations of several datasets (Figure V-39 D – I and M – O).

By comparing the $g(r)$ of the samples with the two distinct close-packed lattice configuration S and L (short and long axis particle contact, see Chapter V.1.3, Figure V-4), it could be shown that the S configuration was indeed the dominant lattice type at the particle monolayer. Thus, the main *BRAVAIS* lattice type was centered-rectangular.

In few areas, the L configuration could be found on the rubbed particle monolayer as well. The particles maintained an orientation of 90° to the rubbing direction and the *BRAVAIS* type was centered-rectangular for AR 1.4 (at AR 1.73 it would be square, see Chapter V.1.2.1).

The different crystalline lattices were successfully used to create different kinds of open-packed ellipsoidal particle monolayers in Chapter V.5.

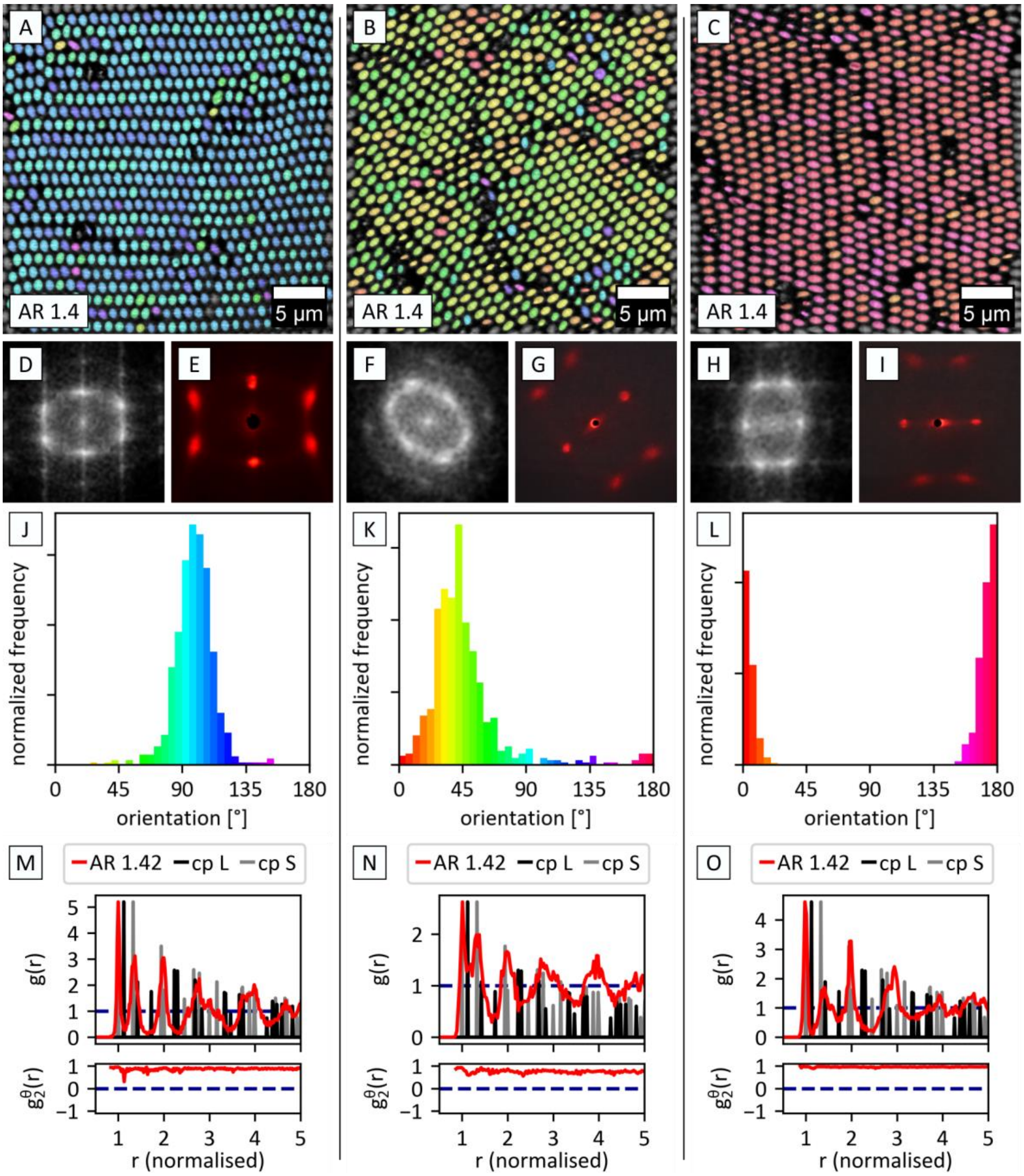


Figure V-39. Detailed investigation of ellipsoids rubbed in different directions: A: Horizontal 0° – particle orientation 90°, B: diagonal 135° – particle orientation 45°, C: vertical 90° – particle orientation 0°. The FFT (D, F, H) of the LSM images were similar to laser diffraction patterns (E, G, I), showing the large area of similar orientation. J, K, L: Histograms of particle orientation of images A – C (coloration the same with A – C). M, N, O: The particle correlation function showed peaks similar to close-packed S configuration (grey, test data with AR 1.42 made according Chapter V.1.5) and no/few L configurations (black). Especially the horizontal rubbed sample showed very distinct peaks. The orientation correlation $g_2^\theta(r)$ was very close to 1 indicating parallel orientation of the particles. All r normalized to minor diameter of particles from SEM image evaluation (see Table V-4). Particle monolayers made from DB13 15 %_2 with AR 1.42 using applicator device 3.

V.4.4.3 Higher ARs (AR > 1.4)

With a low AR of 1.4, the ellipsoids could be ordered in large-domain, crystalline lattices, as seen in the previous chapter. For larger AR of the particles this was, however, not the case. Just with AR 1.7 (particle type DB07 25 %) some crystalline cp domains of few µm were visible on the samples. The cp domains showed both L and S lattice configurations (see Chapter V.1.3, Figure V-4), but there was a smooth transition from lattice to lattice so also intermediate configurations were present. In few areas, the L configuration was dominant, resulting in square lattices (Figure V-40 purple squares). Dislocations of particles forming non-close-packed rectangular lattices with long axis contact were also often seen in these rubbed layers.

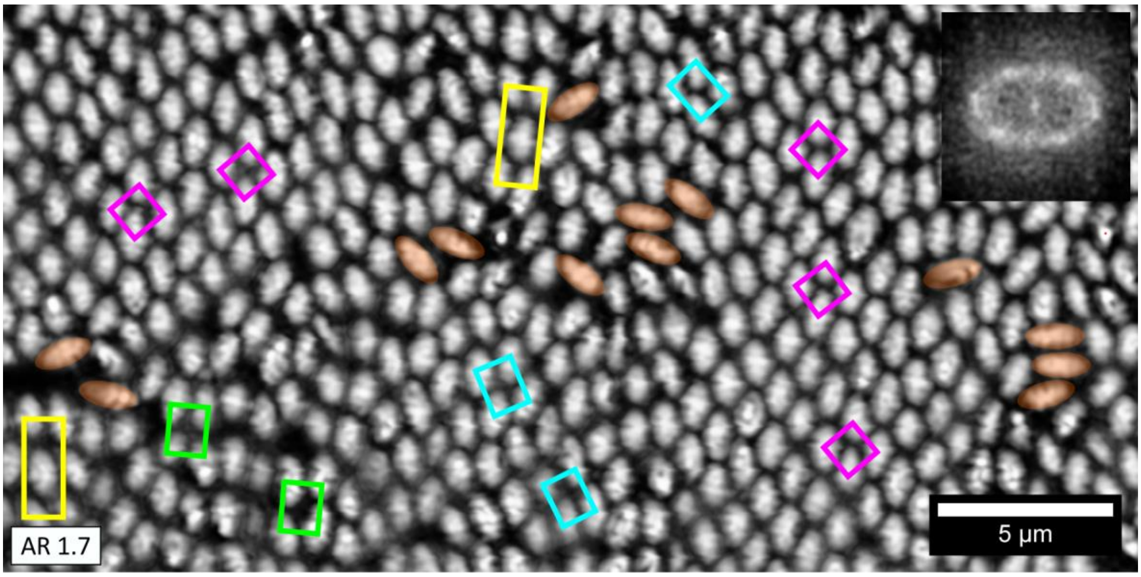


Figure V-40. LSM image of rubbed monolayer with particles of AR 1.7 (DB07 25 %). At some regions, L configuration was strongly present. Due to an AR of 1.7, L configuration led to square lattices (pink squares). S configuration (yellow) still gave centered-rectangular lattices. The transition from lattice to lattice was smooth and intermediate lattices were present (cyan). Dislocations (green) can have a primitive rectangular lattice. Few particles were oriented in the direction of rubbing (orange-colored ellipsoids). Inset shows FFT of image. Sample made with applicator device 1 with syringe pump, speed 5.1 cm/min.

As known from Chapter IV.6.4, the larger the AR the more the two minor axes differ. Additionally, high AR particles have a large contact area with the rubbing substrate. Both makes it harder for larger aspect particles to roll along during rubbing. As with higher AR, more particles were found to be oriented in the direction of rubbing (few visible at AR 1.7, Figure V-40 orange-colored ellipsoids), I propose, that there was a transition from rolling to sliding behavior with increasing AR of the particles (Figure V-41).

Due to many particles shifting their orientation, particles with higher AR were not able to pack in large domains. Thus, an evaluation by pair or angle correlation functions would not be rational. Investigating the overall orientation did clarify the dominant particle movement (rolling/sliding). Particles of AR 1.7 (stretched to 25 %) did mostly roll as they showed a preferred orientation perpendicular to the rubbing direction. This was the same for AR 2.17 (DB13 50 %). However, their orientation distribution was not as narrow as of AR 1.7. With AR 3.26 (DB07 75 %) the preferred orientation changed already by 90°, from 90° to 180° (Figure V-41 G – I). The particles assembled via sliding. Some were even pushed onto each other.

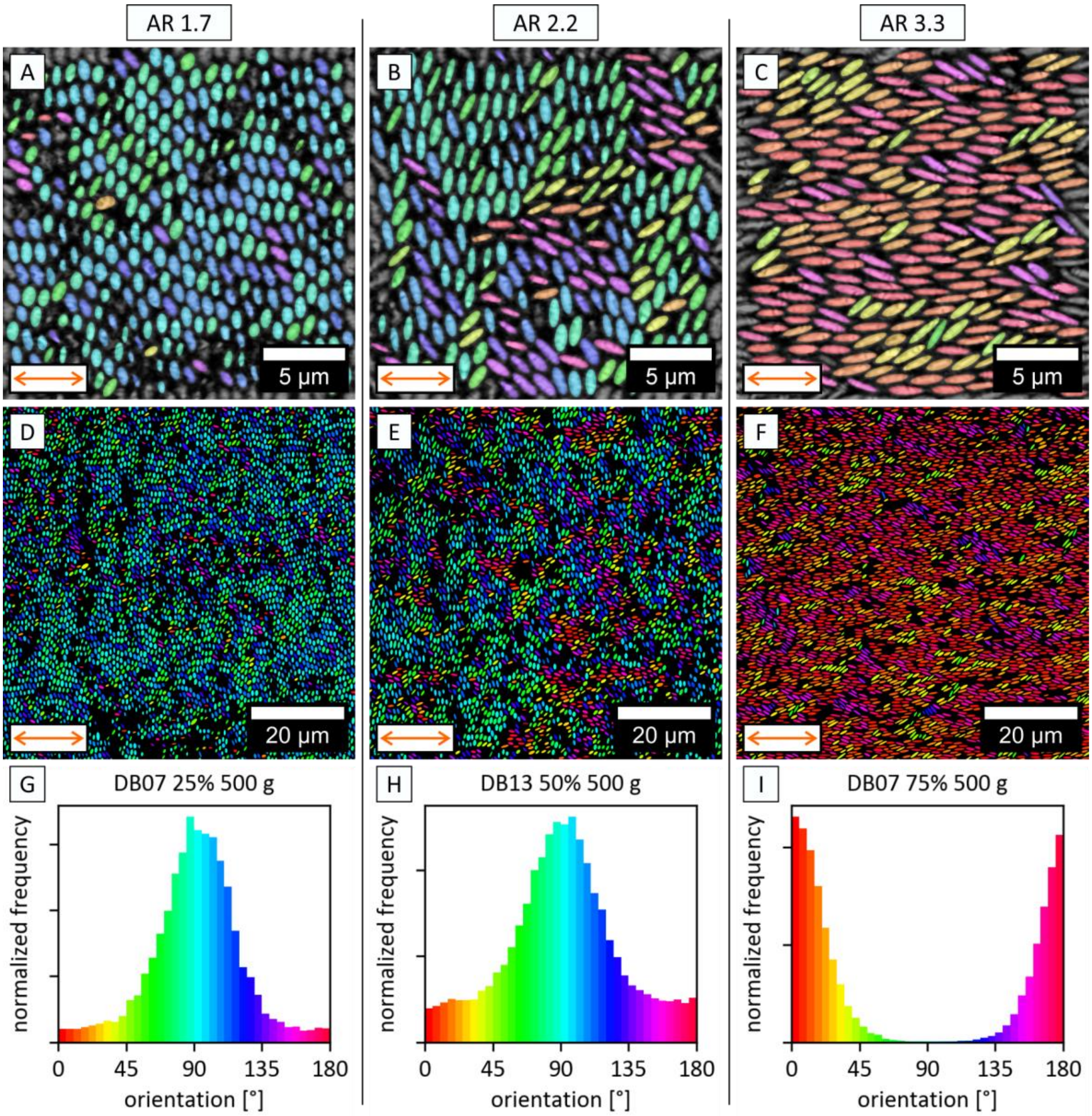


Figure V-41. Comparison of rubbed monolayers of different higher ARs. A – C: Zoom-in on LSM image of rubbed particle monolayers with orientation-colorization. D – F: Identified ellipses/orientations on complete example image (for clarification, no overlay). A – F: Rubbing direction indicated with orange arrows. G – I: The angle comparison histograms (color according to the angle in D – F) are a combination of up to 10 images with more than 3500 particles per image. The histograms are normalized so the area equals 1. Samples from left to right: DB07 25 % AR 1.74, DB13 50 % AR 2.17, DB07 75 % AR 3.26. Samples made with 3 cm/s and 500 g on the semi-automated rubbing device 3, rubbing direction left-to-right and vice versa.

V.4.4.3.1 Influence of the Rubbing Pressure

According to Park et al. [151], the pressure applied during rubbing of the particles can help assembling spherical particles to large grains. Park et al. claimed, that the orientation of the crystalline domains aligned perfectly with higher pressures (up to 2 kg/cm²).

In contrast to studies with spherical particles, in which higher pressure on the applicator led to a domain growth, the ellipsoidal particles did not show larger domains with more pressure. However, with an increased pressure of 2.3 kg/cm² (added weight 500 g, contact area estimated to 0.225 cm² by 1.5 cm × 1.5 mm, applicator device 3, own weight of applicator 10 g), more particles were directed into 180° orientation (Figure V-42, compare histograms of 0 g (A – C) and 500 g (D – F)). The movement of the particles shifted from rolling (90°) to sliding (180°). This was of course not beneficial for ordered domain growth.

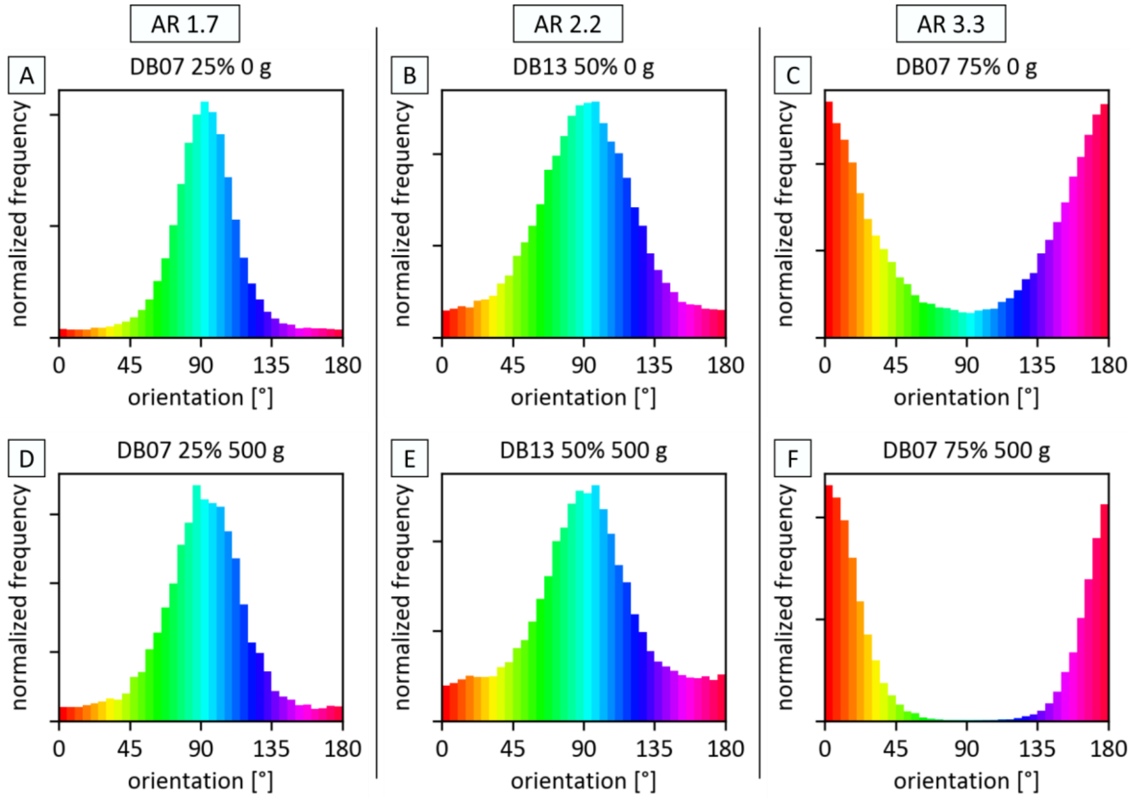


Figure V-42. Combined orientation histograms over multiple LSM images from rubbed particle monolayers made with different loads. A – C: No additional load (0 kg/cm²) and D – F: additional load of 500 g (2.3 kg/cm²; from Figure V-41). Samples made with semi-automated device 3 and 3 cm/s.

Nevertheless, the orientation distribution of the highest investigated AR particles narrowed with higher pressure, as the main movement at 0 kg/cm² was already sliding (180°).

Note that no differences could be identified on the crystalline lattices of AR 1.4 particles (see supporting information V.8.9). Both pressures applied showed 90° oriented particles (rolling) and large crystalline domains over the whole sample images.

V.4.5 Proof-of-Concept: Take-up-Transfer of Rubbed Monolayer

The rubbed monolayer of low AR particles (AR 1.4) showed large domains with crystalline lattices. For further use, it is desirable to relocate such an anisotropic particle monolayer to different substrates. In this case, the developed protocol here might help to achieve the relocation of the monolayer from the sticky PDMS surface to a water surface and further to any other substrate.

As the rubbed particles could not be floated onto water directly from the PDMS rubbing substrate, the rubbed monolayer needed to be taken-up from the sticky PDMS surface. A take-up could be achieved using a PVA foil of arbitrary thickness or a PVA spin coated glass slide (0.5 wt% PVA solution, dry bake 10 min at 180 °C). The PVA-slide or PVA-foil was wetted by holding it above an 80 °C water basin for 30 – 60 seconds. Then it was immediately pushed upside down onto a freshly rubbed particle monolayer. A previous removal of the excess particles on top of the monolayer (V.4.1.1) is recommended for this process. After drying, the particle-covered PVA-glass or -foil was peeled off the PDMS rubbing substrate.

The prepared monolayer on PVA glass showed some PVA filling in between the particles (Figure V-43 A). Additionally, line and orientation defects from rubbing, and voids by missing particles (rubbing and pick up) were visible. Most of the particles were just slightly embedded into the PVA (about 100 nm or less; min 90 % of the height was free standing, measured by 3D mode of LSM). The embedding height depended on the humidity of the PVA surface as well as the pressure applied. However, those parameters were investigated just qualitatively within this proof-of-concept study.

As the particles stick out far from the PVA layer on the glass, it was possible to relocate them in a controlled way to a water surface. To do so, the prepared particle-glass was immersed very slowly into an $HCl_{(aq)}$ filled basin (0.5 M, similar to *RETSCH* method at Chapter V.2.2.1) under a shallow angle (< 30° to water surface). The monolayer detached from the glass, floated on the $HCl_{(aq)}$ surface, and could then be relocated to another glass (Figure V-43) or any substrate capable to withstand low pH values. Higher pH values might be possible but were not tested in this study.

The orientation of the particles inside defect-free domains did not change during the process. However, the monolayer split in smaller pieces on defects (lines or missing particles). Those patches rotated slightly.

The floated particle monolayer had no visible PVA cloaking between the particles and might be used e.g. for particle lithography. A transfer with the lattice-transfer-method, like in Chapter V.3, should also be possible but was not investigated yet.

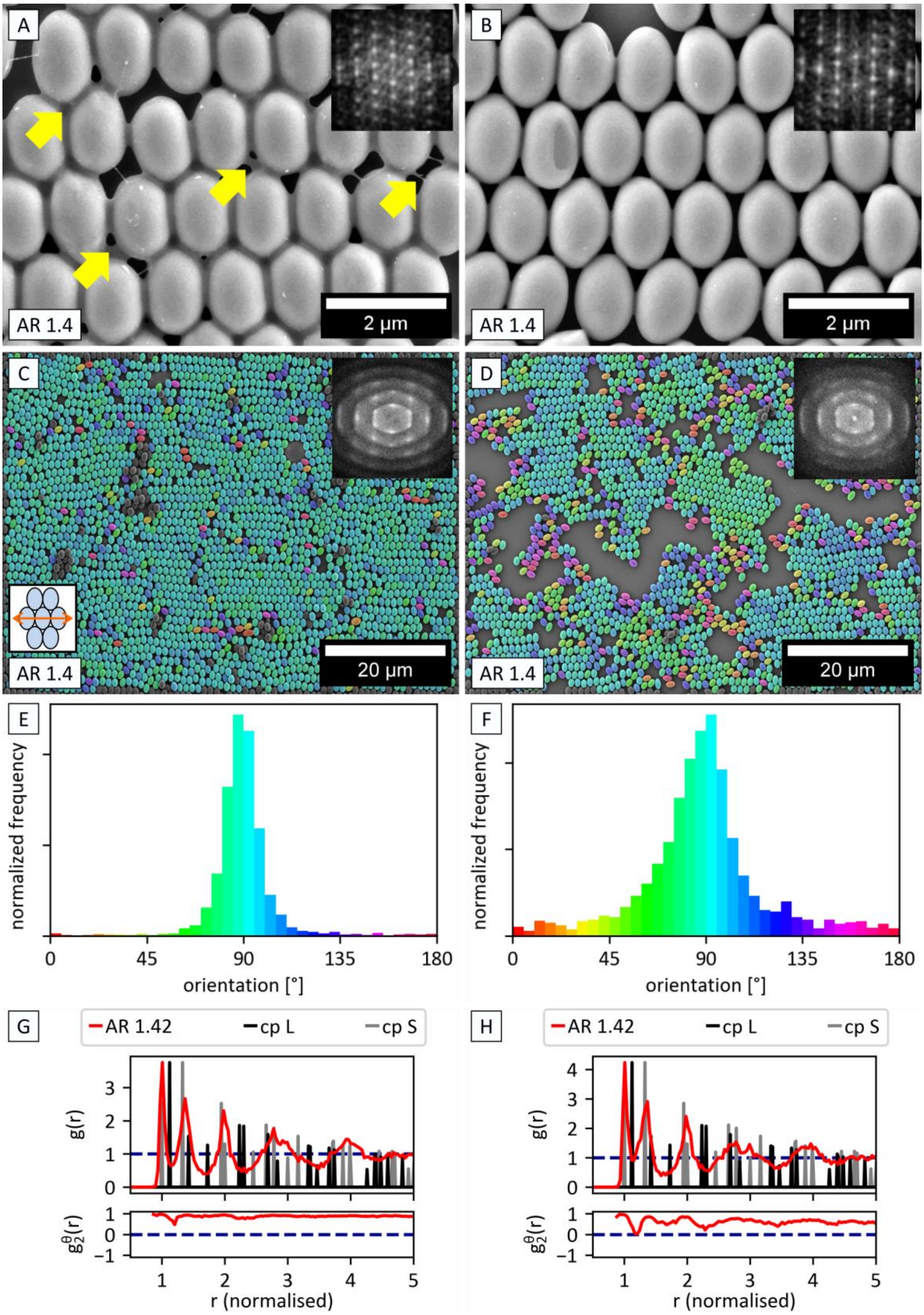


Figure V-43. SEM images of ellipsoidal particle monolayer A, C: on PVA-coated glass slide before floating on water. Some PVA in between the particles was visible (e.g. at yellow arrows). B, D: Particle monolayer after floating on water. No PVA in between the particles was visible. Insets of FFT show high crystallinity. E, F: The orientation distribution was very narrow prior to floating, but widened after floating on water due to turning of domains. G, H: Close-packed S configuration identified by pair correlation functions. All r are normalized to minor diameter of particles from SEM image evaluation (see Table V-4). Particles: DB13 15 %_2 AR 1.42.

V.4.6 Summary of Sub-Chapter V.4

Within this subchapter, the rubbing method of Park et al. [151] was applied on ellipsoidal polymer particles for the first time. The ellipsoidal particles were successfully assembled by the rubbing method into dense, regular crystalline monolayers.

Low AR particles of AR 1.4 did exhibit large crystalline, close-packed domains within the samples, with the dominant particle contact by short axis (S configuration). The laser diffraction pattern showed distinct peaks at anisotropic positions. Therefore, these low AR particle monolayers might be useful for orientation dependent optical devices.

Particle monolayers of AR 1.7 still did show some crystalline domains. Some of these domains had the particle contact along the long axis (L configuration) resulting in a square *BRAVAIS* lattice. Larger AR particles (AR 2.17 and AR 3.26) did not show crystalline domains, but had a preferred particle orientation.

While the particles were mostly oriented perpendicularly to the rubbing direction (90°), the orientation of the largest AR particles changed in a 90° manner. The particles in the monolayer displayed an orientation in the direction of rubbing (0°). I propose that this result was due to a change in the assembly mechanism. With low AR the particles can roll (90°, orientation along short axis) over the surface, while for high AR particles it was easier to slide (0°, orientation along long axis). The sliding or rolling behaviors could be triggered by applying different forces on the applicator (more force less rolling, instead more sliding). Further research on this topic would be beneficial to control these two assembly behaviors.

In the end of this sub-chapter, the high-crystalline large-domain ellipsoidal particle monolayer of AR 1.4 was successfully used in a proof-of-concept study. There, it was possible to take up the rubbed particle monolayer by a PVA covered glass slide and float it on a water surface followed by a take up on another substrate. Although the transfer was not yet fully controlled, the successful proof-of-concept study already enables the use of rubbed monolayers e.g. in colloidal lithography (free voids in between particles).

V.5. Open-Packed Rubbed Monolayers

In this sub-chapter, the method of stretching an elastomeric substrate to open-up close-packed monolayers will be introduced. Although the method itself is self-explaining and very simple, a surprisingly low number of publications was found using the opening of an hcp monolayer due to deformation/stretching of an underlying elastomeric substrate [140]. Moreover, no publication was found in which the method by itself is the scope of the publication and, therefore, investigated in detail.

For example, Yan et al. [141] used the method to open up a hcp silica particle monolayer on PDMS to get specific lattice spacings/positions. They used a process of swelling or stretching PDMS with an hcp monolayer on top and transferring the silica particles with the new open positions to PVA coated glasses by pressure and heating to 100 °C. In 2010, Li et al. [142], with the same co-author B. Yang, evolved this method to a cyclic process as they calcined/burnt away the sticky PVA coating and transferred the particles on their new positions to another PDMS to swell/stretch it again. With repetitions, they managed to open up the particle monolayer to their desired lattice spacing. By using a stretching step instead of swelling, they managed to get different lattice types. Nevertheless, Yan and Li both used an evaporation based hcp particle layer assembly technique following a 2D monolayer transfer to PDMS, limiting their sample size.

With the rubbed monolayers of Chapter V.4 the sample size became extraordinarily large (limit was substrate size of multiple cm²). Moreover, almost the complete rubbed monolayer was one single domain and the orientation of the hcp monolayer could be exactly defined as the particle-particle contact lay within the rubbing direction (see Chapter V.4.4.1). Thus, the rubbed monolayers should have some advantages for using it with an elastomeric substrate stretching method.

However, it is known that the method of elastomeric stretching does not result in a true 1D lattice transfer. The lattice transfer gained from this method is a combination of 1D stretching and compression/shrinking perpendicular to the applied tensile deformation (shear-thinning of the substrate).

In this sub-chapter, the deformation of the close-packed rubbed monolayers of Chapters V.4.4.1 (spheres) and V.4.4.2 (low AR ellipsoids) will be discussed.

V.5.1 Elastomeric Substrate Stretching Process

By using the implemented rubbing technique, the dried spherical or ellipsoidal particles were applied onto 10 cm long PDMS stripe substrates (width: 1.5 cm, thickness: 1 mm) forming close-packed monolayers. The PDMS substrates were mounted into the stretching device (see Chapter II.2.1.2, Figure II-5) in such a way that the particle loaded PDMS strips were sitting on top of the stretching device with their ends entwining the brackets (Figure V-44). This configuration was necessary for microscope investigations. As the mounted samples should also be investigated by laser diffraction (Appendix VIII.1.9), a rectangular window was cut into the ground plate of the stretching device (visible in Figure V-44 B in the back), so the laser could reach the sample.

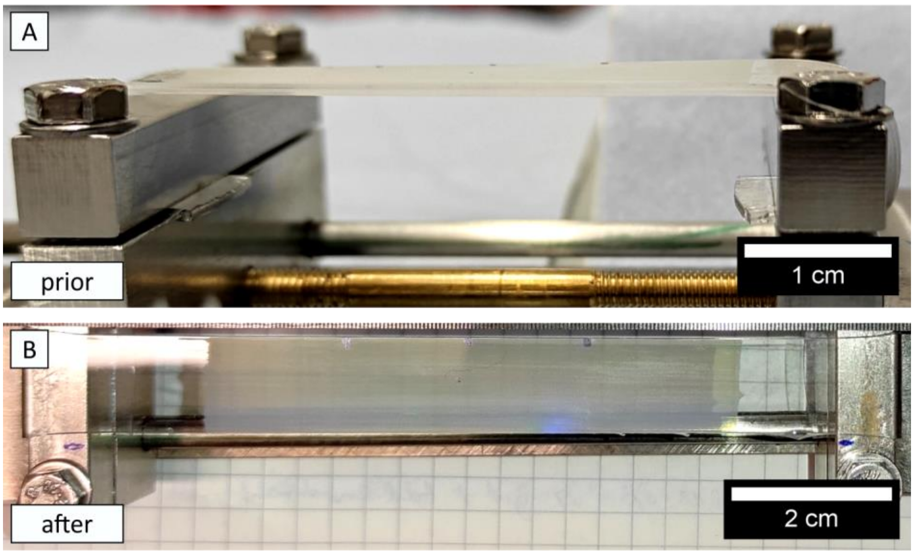


Figure V-44. PDMS sample A: before (side view) and B: after (top view) stretching. Sample was clamped on top of the stretching device entwining the brackets. The clear borders of the rubbed particle monolayer were achieved using sticky tape. B: The shrinking perpendicular to stretching direction (shear-thinning, bone-shape) is visible. Scale bar at A is estimated due to strong perspective view.

The mounted PDMS substrates were stretched without heating.

The stretching ratio was monitored by measuring the elongation and width of the sample area by Fiji/ImageJ, while the orientation of the monolayer and the particles were recorded by LSM images in 2D laser mode.

V.5.2 Substrate Stretching of HCP Spherical Particle Layers

Using the substrate stretching method with spherical particles, two distinct different possible stretching directions need to be described.

First, spherical particle monolayers retaining an ordered lattice with particle-particle contact in stretching direction ($\alpha = 0°$). The diagonally located particles ($\beta = \pm 60°$) could move towards each other due to compression in 90°. The rubbed lattice is stretched open and could be transferred from a hexagonal to a square lattice (Figure V-45 A to B). It was further possible to conserve the large domain square lattice of the particles by PVA take-up (similar to Chapter V.4.5) of the monolayer (Figure V-45 C – E).

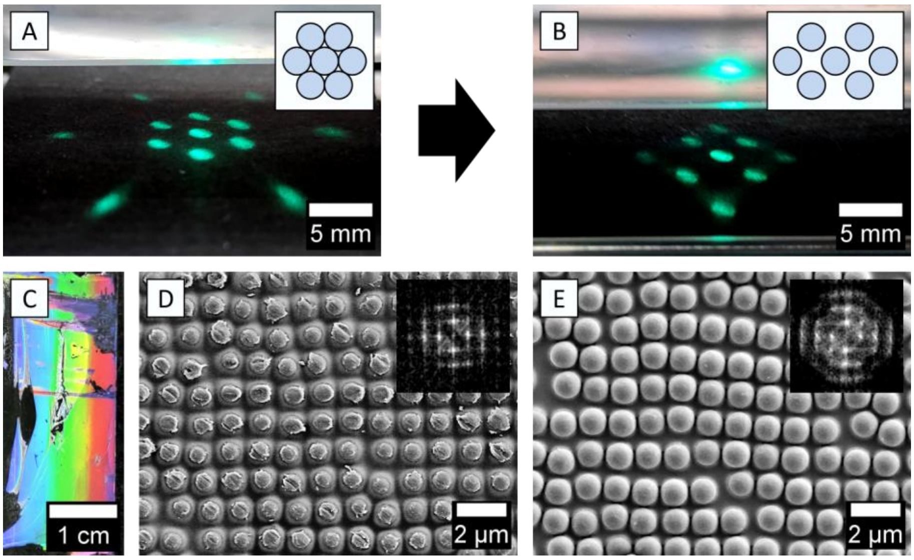


Figure V-45. A rubbed hcp monolayer of spherical particles was stretched to around 45 %. A part of the resulting large square lattice particle monolayer was then transferred to a PVA coated glass slide. A, B: Photographs of the laser diffraction pattern of the PDMS strip mounted in the stretching device (inset shows sketch of particle lattice). C: Photograph of the monolayer on glass slide under incident white light from the back right. The homogeneous color shift (rainbow) over the whole sample shows the ultra-large domain size of the sample. D, E: SEM images of the sample in which the particle were embedded more (D) and less (E) into the PVA layer (FFTs as insets). Note that the SEM images were turned in regard to image B. A zoomed out SEM image can be found in the supporting information V.8.10. Scale bars of photographs A – C estimated. Particles: DB13.

The lattice transformation factor should be 1.73 that means an elongation in one direction of 73 % is necessary to gain a square lattice (compare to Chapter V.1.2). Due to the additional compression in y-direction, a true position transformation factor needed to be calculated by the sum of stretching and compression factors. A stretching in x-direction of just 45 % resulted in a position transformation factor of 1.73 and, therefore, in a squared lattice (Figure V-46).

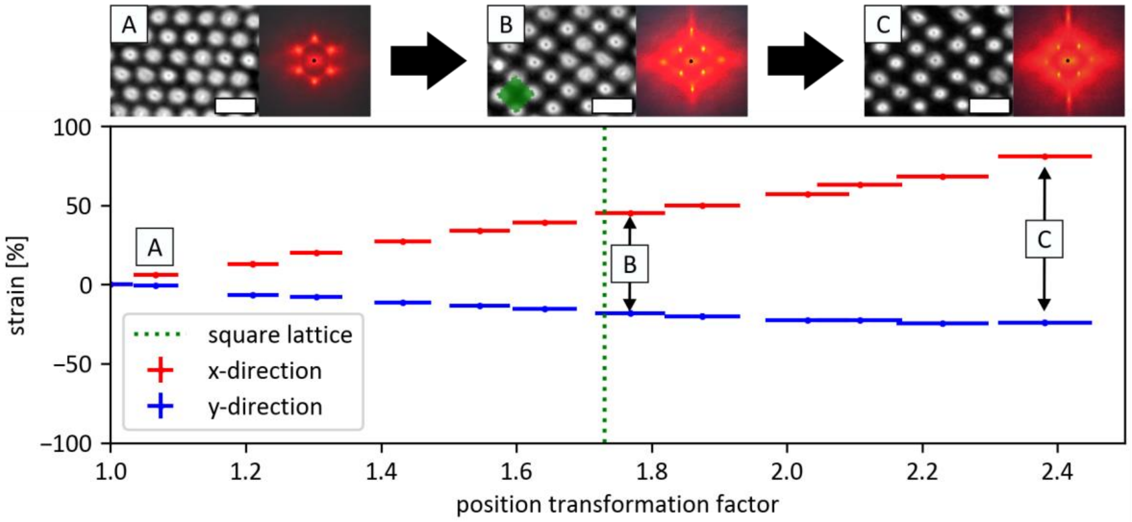


Figure V-46. Diagram of elongation (x-direction) and compression (y-direction) of the PDMS substrate during stretching in x-direction. The resulting position transformation factor defined the amount of stretching necessary to gain e.g. a squared lattice (green dashed line in diagram and green square in B); error propagation with estimated reading error from photographs 0.02 cm. A, B, C: LSM and laser diffraction images of the hcp particle monolayer at the different strains. Scale bars at LSM images are 2 µm.

An open stretched hexagonal lattice with a transformation factor of 3 as calculated in Chapter V.1.2.2 was not possible because the PDMS used here (mixing ratio 10:1) ruptured already at 80 % strain (maximum position transformation factor 2.4). Other PDMS sheets with less cross linker might be stretched further. However, this was not investigated.

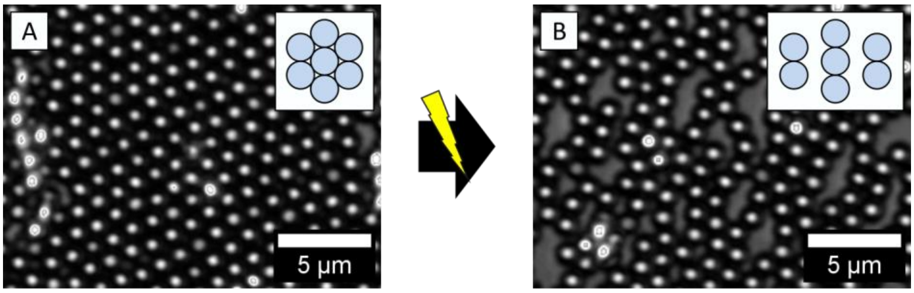


Figure V-47. The stretching of A: hcp spherical particle monolayer rubbed in direction $\alpha = 30°$ resulted in B: wavy lines due to an unable compression of the lattice in particle-particle contact direction.

Second, an hcp direction of particle-particle contact perpendicular to the stretching direction ($\alpha = 30°$ or $90°$) resulted in deformed line patterns (Figure V-47). The in-contact particle positions could not be compressed. The particles gave way by forming wavy lines. Therefore, a defined line pattern with a rectangular centered lattice (Figure V-47 B scheme) was not possible with the elastomer stretching method.

V.5.3 Substrate Stretching of CP AR 1.4 Ellipsoid Layers

Changing the motif of the lattice to ellipsoids, gave two additional parameters to the stretching method – the AR and the orientation of the ellipsoidal particles (see Chapter V.1.4). The orientation of the particles were thereby directly dependent on the direction of rubbing. As known from Chapter V.4.4.2, the dominant lattice for the crystalline AR 1.42 monolayer was the short axis contact lattice structure (S configuration). However, at few positions, long-axis contact with 90° to the rubbing direction could be observed (L configuration). Similar to spherical particles, the close-packed lattice can be stretched in directions along particle contact (S0 and L0, see sketch in Figure V-6) and true square lattices can be obtained by following path S0 or L0 (compare to phase diagram Figure V-9). Stretching lattices with particle contact perpendicular to the stretching direction (S90 and L90) would have the same problem as before at spherical particles by compressing an already close-packed line, leading again to particles giving way and to uneven stripe-like patterns.

V.5.3.1 Theoretic Possibilities for CP AR 1.4 Particle Layers

Square lattices resulting from S0 and L0 should show both the same lattice type (turned 90°, see sketch in Figure V-6). However, the square lattice from L0 has a lower particle distance and, therefore, needs a lower lattice transformation factor (AR 1.42: factor $T = 1.22 \approx 14$ % stretching) compared to S0 (factor $T = 2.46 \approx 83$ % stretching). The connection of the lattice positions with the AR and the needed position transformation factor can be found in Chapter V.1.3 and V.1.4.

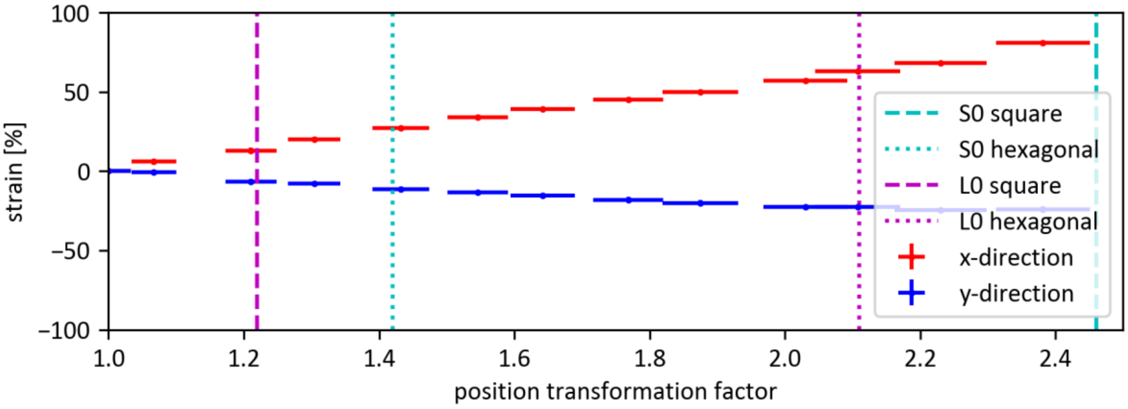


Figure V-48. Strain with position transformation factors needed for different defined lattices with particles of AR 1.42. The S0 square lattice is out of reach with the used PDMS substrate here. Values of defined lattices and dependency of strain to transformation factor T can be found in the supporting information V.8.11.

Due to the PDMS used here, a stretching to a position transformation of 80 % (factor $T = 2.4$) was the maximum possible (often ruptured before).

A square lattice (lattice angle after transfer $\gamma_1 = 90°$, see Chapter V.1.4.1) by starting with S configuration in short axis direction ($\alpha = 0°$, direction 1) was, therefore, not possible ($T = 2.46$, 83 % elongation needed, see Figure V-48 dashed turquoise line on the right). However, by starting from S configuration a slightly open-packed hexagonal lattice ($\gamma_1 = 120°$, was possible, due to the fact that the particles were not on hexagonal lattice positions at their start (Figure V-48 dotted turquoise line, $T = 1.42$, 25 % elongation).

For L configuration, a squared lattice ($\gamma_1 = 90°$, $T = 1.22$, elongation 14 %) and the open-packed hexagonal lattice ($\gamma_1 = 60°$, $T = 2.11$, 64 % elongation) were in the range of the possible position transformation factors from the used PDMS (Figure V-48 purple lines).

V.5.3.2 Results for the Lattice Configurations S0 and L0 of AR 1.4 Layers

The samples for the stretching experiments of S0 and L0 were made by rubbing the long substrate in x-direction (along the stretching direction) for S and in y-direction for L.

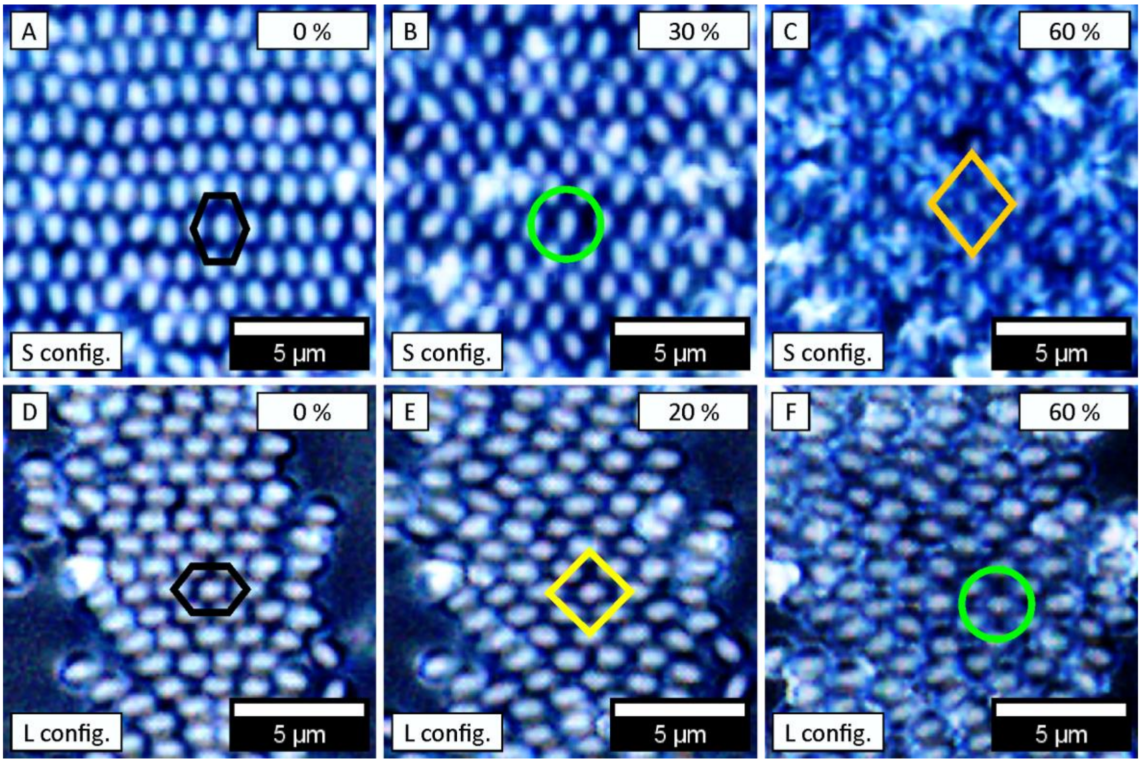


Figure V-49. Ellipsoidal particle monolayers and their stretched layers of AR 1.42 particles (DB13 15 %_2). A – C: S configuration and D – F: L configuration. A, D: Starting lattices (black hexagons). B: At 30 % for S and F: 60 % for L, the PDMS-stretched layers showed hexagonal open-packed lattices (six particles on green circles). A square pattern was visible at 20 % stretching for the L configuration (E, yellow square). For the S configuration a square pattern was not possible with the applied maximum stretching of 60 % (C, orange rhombus). Optical bright field images (color 2D mode in LSM) are shown as scanning artefacts of LSM2D images were frequent at those samples.

Since the L configuration is not the primary lattice of the particles, just a few small spots could be found for investigation (Figure V-49). The particles acted as described by theory. The lattice was opened and the orientation retained. However, the imaging and further orientation evaluation was not possible due to many line-scanning artefacts in the LSM2D images (similar to Chapter V.4.3).

Particle monolayers with AR 1.42 particles and S configuration displayed distinct laser diffraction patterns (Figure V-50). The pattern transformed first to a real hexagonal lattice (stretching to 30 %, theory by linear fit of elongation data, see supporting information V.8.11: 25 %). Then the lattice opened up more but could not be transformed to a true square lattice due to the mentioned restriction of the PDMS. With 60 % stretching, the laser diffraction showed that the lattice was close to a square lattice (Figure V-50 G).

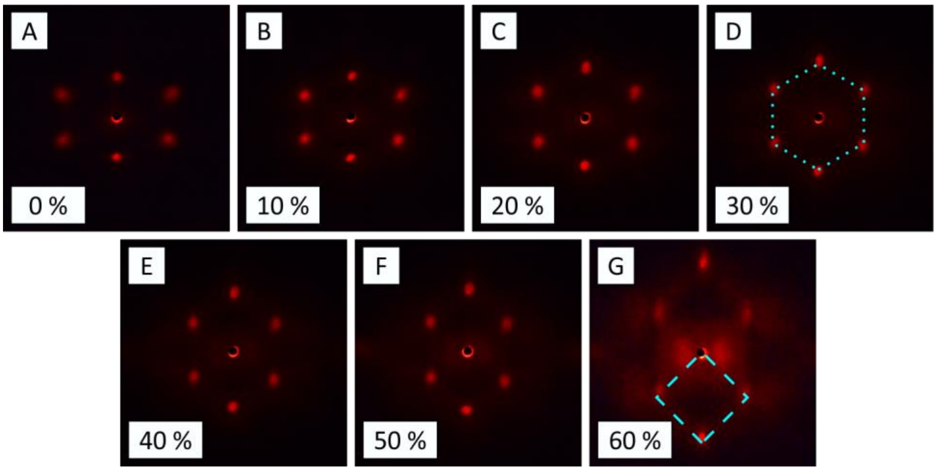


Figure V-50. Laser diffraction patterns of the PDMS stretching experiment (with $\alpha = 0°$) started from S configuration of AR 1.42 particle monolayer. A: 0 %, B: 10 %, C: 20 %, D: 30 % (hexagonal lattice), E: 40 %, F: 50 %, and C: 60 % (not perfect square lattice, PDMS close to rupture).

Regions with L configuration were qualitatively investigated with their optical images. Laser diffraction patterns of these small regions (Figure V-49 D – F) were not possible (less than 100 particles). A square lattice conformation could be reached at around 20 % stretching (theory 14 %) and an open-hexagonal lattice at 60 % (theory 64 %). This was in agreement with the theory (Figure V-48).

V.5.3.3 Twisting of Ellipsoids during Substrate Stretching of AR 1.4 Layers

While theory already showed the large variety of different possible lattices with fixed orientations at different stretching angles, the real lattices found exhibited additional features. Not only could particles hinder each other to move to the right position (e.g. at S90 and L90, see Chapter V.1.4), some particles even started to turn slightly during stretching. The orientation of the particles after PDMS stretching was, therefore, not always their initial orientation. The particles turned in a way that the angle to the stretching direction was lowered (Figure V-51). The orientation distribution became narrower and the mean value shifted to lower angles (Figure V-51 I). Particles with orientations vertical or horizontal to the stretching direction were not affected. Thus, an explanation could be that the ellipsoids had multiple contact points with the PDMS resulting in a twisting force on the particle. The contact points seemed to differ from particle to particle and at different orientation angles. The slight turning was seen best on a diagonally rubbed particle monolayer.

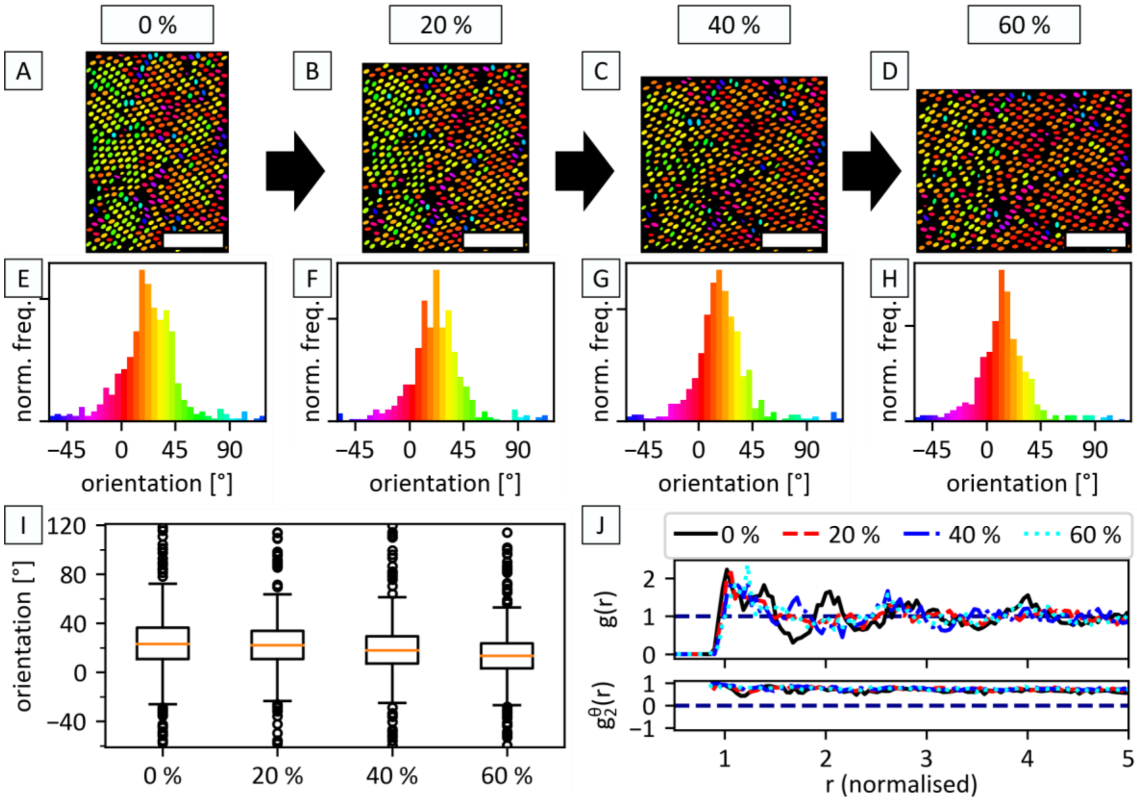


Figure V-51. Same area of a diagonally rubbed AR 1.42 particle monolayer at different stretching of the PDMS substrate. A – D: Ellipsoid positions with angle dependent coloration (from evaluation program). Scale bars 10 µm. E – H: Histograms of the particle orientations of A – D. The histogram of H is narrower than E. I: The boxplots of the orientations show the decreasing median. J: Pair correlation functions. Particles: DB13 15 %_2.

As expected, $g(r)$ did change with the stretching. Unfortunately, it could not be further evaluated due to the poor image resolution (Figure V-51 J). Meanwhile, the angular particle correlation functions ($g_2^\theta(r)$) show the retained parallel orientations of the particles.

V.5.4 Summary of Sub-Chapter V.5

Particle monolayers can be opened up by elastomer substrate stretching. However, for the process, the orientation of the monolayer should be in particle-particle contact towards the stretching direction. If the particle contact is perpendicular to the stretching direction, the lattice will be distorted due to an occurring compression of the elastomer perpendicular to the tensile force.

With the rubbing method, the direction of the monolayer can be well controlled and a particle contact perpendicular to the stretching direction can be avoided. By stretching a hcp spherical particle monolayer to around 45 % elongation, a large-area square lattice particle monolayer was successfully produced. Some square centimeters (> 5 cm²) of this monolayer could be transferred to a PVA covered glass without losing the particle lattice positions, hence, giving the possibility to use it in optical devices.

For ellipsoidal particle monolayers, stretching the lattice displayed challenges. The AR and particle orientation parameter added further parameters to handle. The different starting lattices with short or long axis contact (configurations S and L) resulted in differently open-packed lattices. Square and open-packed hexagonal patterned particle layers were achievable by using the crystalline AR 1.4 particle monolayers of the previous Chapter V.4.4.2.

While L configuration could be brought to display a square pattern (for AR 1.42 at around 20 % PDMS-stretching), the PDMS could not be stretched to the necessary extent to give a true square pattern to particle layers in S configuration (still a bit distorted at 60 %). However, open-packed hexagonal particle layers were produced from S and L configuration of AR 1.42 particles.

Particles that were not oriented in horizontal or vertical direction showed a change in particle orientation after PDMS-stretching. This might be due to multiple contact points of the particles with the substrate thus twisting the particles. As the orientation distribution narrows with this effect and the diffraction planes should change slightly as well, it could be beneficial to do further investigations in regard of opto-mechanical devices like strain sensors.

V.6. Stretching an Open-Packed Particle Monolayer

With the here described, newly developed, three-step process it was possible to obtain open-packed lattices of ellipsoidal-like particles with crystalline order. For the developed process, the methods of the previous chapters were used “upside-down” (Figure V-52).

Firstly, starting with self-assembled hcp spherical particle monolayers from the *VOGEL* (or the *RETSCH*) method (see Chapter V.2), the layers were transferred to new lattice positions by the previously described lattice-transfer method (see Chapter V.3).

Secondly, the open-packed ordered spherical particle layer was embedded in PVA and stretched to a defined AR (analogous to the ellipsoid making of Chapter II.2.1).

Thirdly, the particles in the stretched layer were recovered from the foil without changing their position, hence, maintaining their crystalline order.

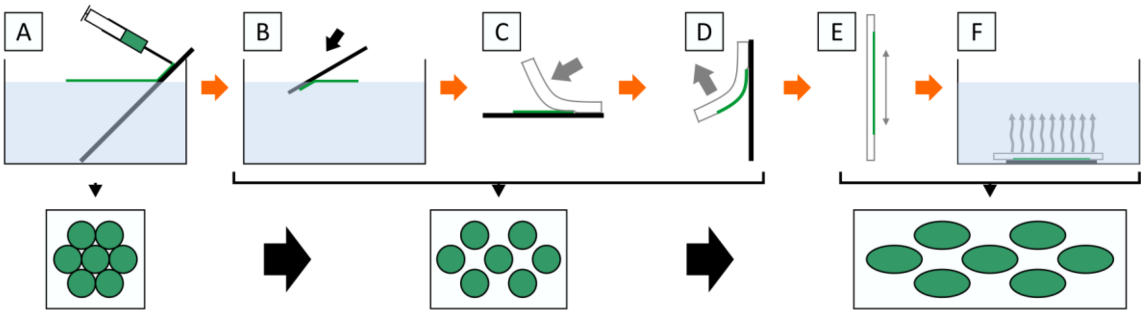


Figure V-52. Scheme of the process for stretching open-packed particle monolayers. A: After preparing hcp particle monolayers by the VOGEL (or the RETSCH) method, the floating particle monolayers are B: transferred to open-packed particle monolayers (according to Chapter V.3). C: The particles are then embedded into PVA by pressing a PVA foil onto the wetted particle monolayer. D: After drying, the particles are incorporated into the foil and can be lifted from the glass substrate. E: The foil is stretched by foil stretching (see Chapter II.2.1). F: The present lattice of the stretched particles is retained during dilution of the PVA foil.

The resulting layers showed particles of different shape depending on the position of nearby particles. Similar looking structures were found by Wang et al. [166] when they made elliptical hemisphere arrays from hcp particle layers with PDMS cast and a micro molding technique.

V.6.1 Process

The particles used in these experiments were self-synthesized PS-PSS particles of 1.06 ± 0.03 µm (DB07) (see Chapter II.1.3.3).

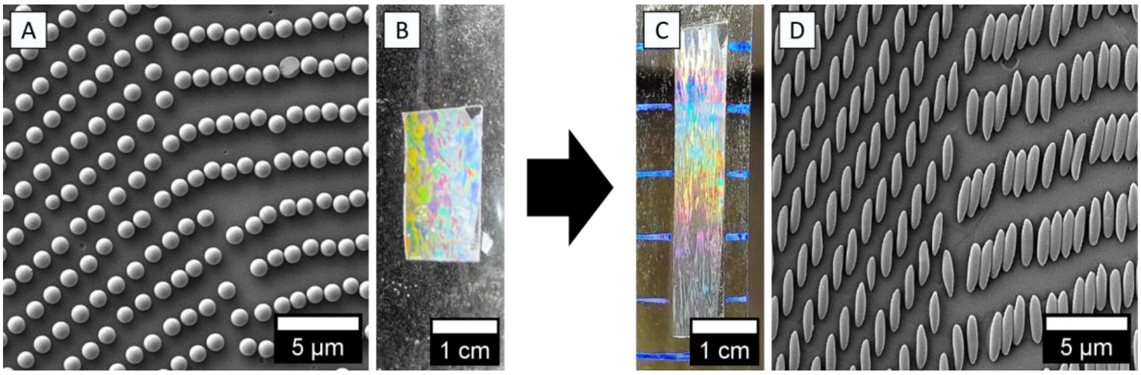


Figure V-53. Example of the process to stretch an open-packed particle monolayer. A: An hcp spherical particle monolayer was transferred to an open-packed state. B: The particle monolayer was embedded into a PVA foil, showing colorful diffraction from illumination with white light from the back. The foil was then stretched to 100 % strain. C: The resulting stretched particles in the monolayer in the PVA foil exhibited still colorful diffraction. E: After recovering the particles without changing the positions, the retained ordered lattice got visible. The stretched particles exhibited different shapes due to hindered stretching by other particles next to them. Used particles: DB07, 1.06 ± 0.03 µm.

Step 1 – open-packed particle layer. The hcp particle monolayers were assembled on 0.5 M $HCl_{(aq)}$ by the *VOGEL* method (Figure V-52 A) and were then stored at 0.1 M SDS solution with grain size enlarged by air stream (Chapter V.2.2.2.3). Applying the transfer method (Chapter V.3, Figure V-52 B) on the spherical particle monolayers resulted in different open-packed lattices. The kind of lattice depended on the orientation of the formally close-packed hexagonal layer to the direction of transfer, as explained in Chapter V.1.2 and at Hummel et al. [139]. The transferred particles were fixed on the glass slide under water by the on the glass coated layer of low T_g polymer (MMA-nBA, T_g ~60 °C) to recover the glass slide from the water without changing the position of the particles (Figure V-53 A).

Step 2 – stretching. The particle monolayer is transferred to a thick PVA foil (30 wt%, 1 mm wet film height) by pressing the plain dry foil on a water-wetted open-packed particle monolayer sitting on the transfer glass substrate (force of around 10.8 g/cm²). The foil-particle-glass-sandwich (Figure V-52 C) needed to dry for several hours. Afterwards, the foil was peeled off the glass (Figure V-52 D), taking the particles with it (Figure V-53 B). The foil showed no mayor disturbance like bubbles or ruptures. The PVA foil was then clamped in the stretching device (see Chapter II.2.1.2) and stretched at 150 °C to the desired elongation (Figure V-52 E). At this proof of concept study, the foil was only stretched to 100 % elongation (see Figure V-53 C).

Step 3 – recovering. The particle loaded, stretched PVA foil was mounted upside down on a low-T_g polymer covered glass slide with a nylon net (part of a 125 µm E-D-Schnellsieb Nylon super-fein) or a 3D printed square-mesh (Polymer PLA, 2 mm hole width, height 0.3 mm) on top. The nylon net or polymer mesh was fixed by sticky tape to the glass slide. The net/mesh had two purposes. It should press the PVA foil onto the glass surface and hinder turbulent water movement directly at the glass surface to destroy the particle lattice. The Mesh-foil-glass sandwich was heated at 80 °C for 10 min so the foil got a better contact with the glass. The sandwich was then stuck to the bottom of a 150 ml glass beaker standing on a switched-off heating plate. Then water was purred slowly into the beaker; firstly by a pipette, later, after the sandwich was covered with water, by a 20 ml syringe. The beaker was filled with 120 ml of water and left standing over night without any movement (Figure V-52 F). Then, the heating plate was switched on to 80 °C for 30 min. After cooling down another 30 min, 100 ml of the water was exchanged carefully to fresh water and left again standing over night. A second water exchange was done. After that, the beaker was emptied completely and the glass substrate with the transferred open-packed particle layer could be recovered (Figure V-53 D).

V.6.2 Results

Depending on the original open-packed lattice, the stretched particle lattice exhibited elongated particles in different patterns. Two major groups could be identified:

The first group was represented by particles having some space around them to deform as individuals, thus showing lattices of individual ellipsoidal-like particles (Figure V-54 A – D). However, mostly, the particles influenced the stretching of neighboring particles by altering the stretching matrix surrounding the particles leaving distorted particles behind.

The second group were particles touching side-by-side already prior stretching, resulting in „fused" line structures (Figure V-54 E and F) when not separated by stretching (Figure V-54 D). The side by side origin of these "fused" structures were traced back to an hcp lattice transfer perpendicular to particle contact (e.g. at $\alpha = 30°$). The "fused" structures still showed individual particles.

By inversing the affine transformation of Chapter V.1.5, a possible origin lattice could be derived from the stretched particles. The SEM images were thereby first turned into the direction of stretching, then ellipses were drawn by CorelDraw at the respective positions and those ellipses were then compressed as a group back to spherical particles (one-dimensional transformation). The result of this inverse affine transformation can be seen at the sketches of Figure V-54. It shows that the resulting stretched particle layers were from different kinds of original lattices (Figure V-54 lattices A – D). Additionally, the inverse affine transformation displays that with a similar original lattice structure very different stretched lattices can be achieved due to different angles of stretching (compare Figure V-54 D – F, all origin from particles in lines).

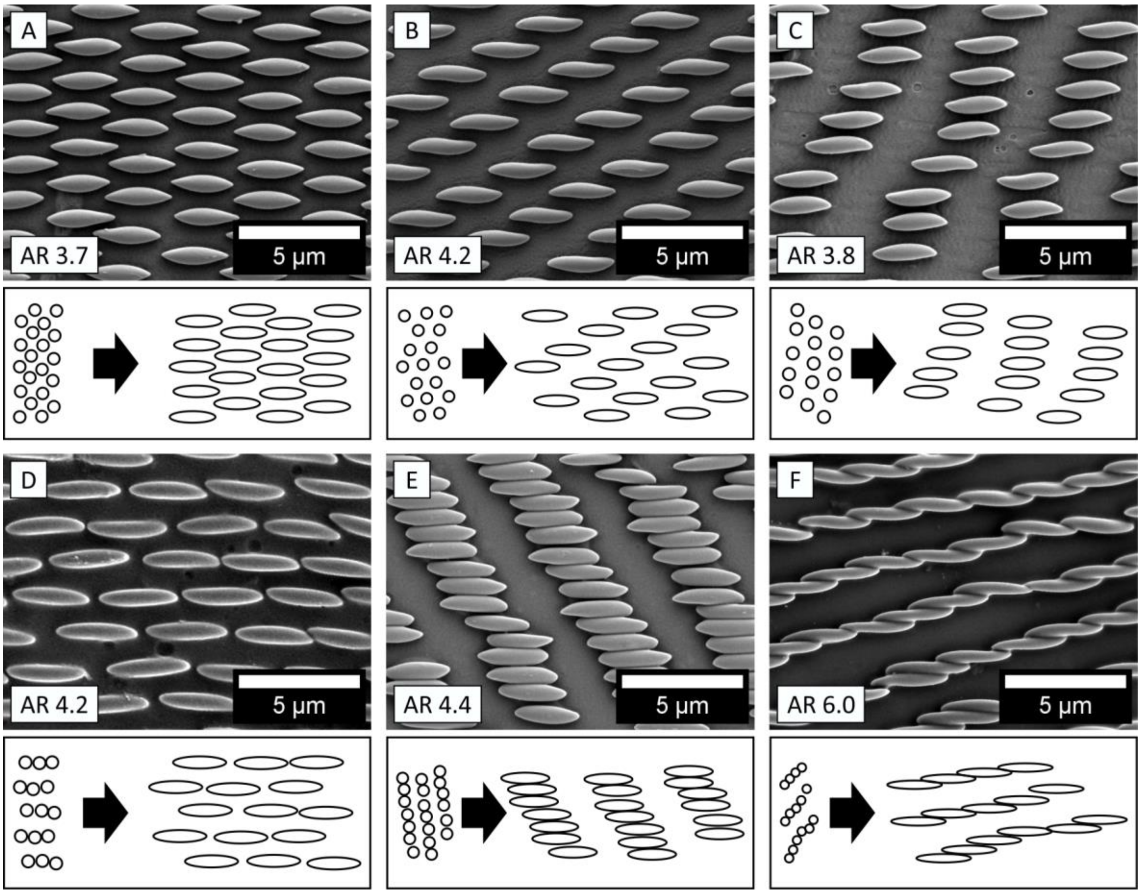


Figure V-54. SEM images of different positions on recovered particle monolayers (images turned so that the stretching direction was in x-direction). A – C: Lattice from a rhombic or square lattice but stretched in different directions. D – F: Lattices resulting from side-by-side particle contact line structures. The lattice was stretched (D) in direction of particle contact, (E) in direction perpendicular to contact/line, and (F) in an intermediate angle to the particle lines. Sketches below SEM images show a reconstruction of possible former lattice positions. A zoomed out image of lattices A can be found in the following Figure V-55 and for B and C in the supporting information V.8.12. ARs of particles in image defined A – E: by python program or F: by hand using Fiji/ImageJ.

Although the complete monolayer sample and thus also the different domains were stretched by 100 %, ARs of the resulting particles were close to the expected AR 4, but differed with the lattice domains. The AR seemed to grow with particles being in close contact (Figure V-54 D – F). However, no clear trend could be identified as sometimes large ARs were also present within non-contact lattices (Figure V-54 B).

The results of these multiple possible lattices show that, especially with the process shown in this chapter V.6, it is crucial for a defined lattice to control both, the monolayer orientation prior to transfer and the direction of stretching.

Depending on the size of the domains from the initial hcp monolayer, the process can result in large areas of same crystalline lattices and orientations (e.g. 150 µm × 150 µm in Figure V-55). With careful control of the origin layer orientation and the followed stretching process, this area should be increasable. Thus, such monolayers might be interesting for optical applications or particle lithography.

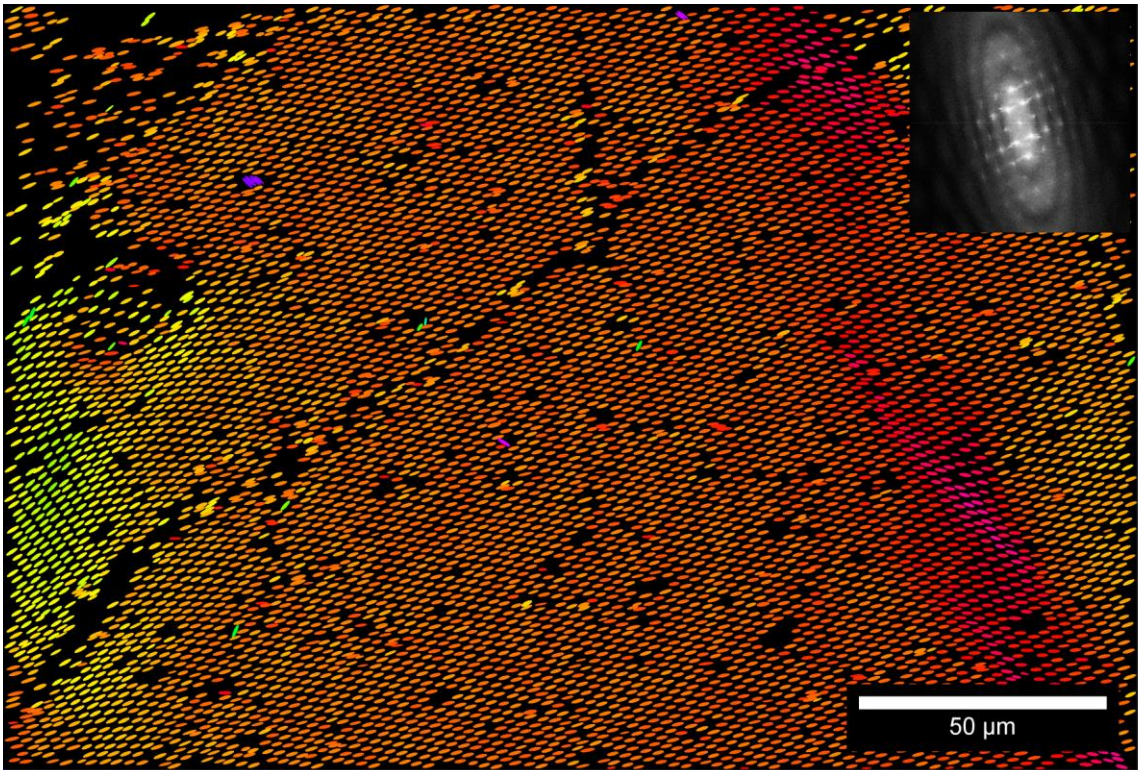


Figure V-55. The angle dependent colorized ellipse image from the lattice of Figure V-54 A (here just overlay), shows a large domain (same orange color), missing particles (black holes), and turned particles (different colors). At some regions (left, yellow, and right, red/pink), the lattice of the layer became deformed (wavy). This was traced back to the insufficient fixation of the stretched foil during dissolving and was found in several other occasions. The distinct spots in the FFT (inset) display as well the high crystallinity of the recovered particle monolayer.

V.7. Summary of Chapter V

Within the chapter, different methods of assembling particle monolayers of spheres and ellipsoids are presented. Mostly, the assemblies of ellipsoids with the different methods displayed were done for the first time in literature.

The chapter started (Chapter V.1) with a theoretical description of particle monolayers and their lattice positions. Further, the sub-chapter implemented the lattice transfer with a transfer factor T, transfer direction angle α and their phase diagram starting from hcp particle layers. With the transfer of an hcp lattice, close-packed ellipsoid lattices could be described as well, by simply changing the motif. Repeating the transfer on such cp ellipsoid lattices resulted in open-packed ellipsoid lattices. The transfer was then described with lattice transfer phase diagrams for different ARs and two different configurations of the starting lattices (L and S).

Further, test datasets with adjustable particle dimensions and positions were constructed via a python script. The position data in the test data was used to implement the pair correlation function $g(r)$ (Equation V.1-17, calculated from particle positions in 2D by access of Dr. Bernd A.F. Koperas code [153]) and the angle correlation function $\mathrm{g}_2^\theta(\mathrm{r})$ (Equation V.1-18 from Zheng et al. [105]).

The next sub-chapter (Chapter V.2), showed the monolayer assemblies of spherical and ellipsoidal particles via self-assembly on water surfaces. Herby two methods (*RETSCH* [129] and *VOGEL* [130]) were applied, both with their own benefits. With both methods, dense-packed particle monolayers of ellipsoids could be made. Few regions of high crystallinity were visible in low AR particles (AR 1.4), while higher AR (up to 4.3) particle monolayers showed patches of same oriented particles without explicit crystallinity.

The dense-packed particle monolayers could be transferred to open-packed lattices (Chapter V.3) utilizing a process from Hummel et al. [139]. However, when the original dense-packed monolayer did not exhibit order, neither did the transferred layer. On the other hand, with the high-crystalline low-AR ellipsoid particle monolayers, the transferred particle monolayers did maintain crystallinity. The process might, therefore, be exploitable for further use if the size of crystalline ellipsoidal particle monolayers could be enhanced.

This monolayer size enhancement was successfully done within the following sub-chapter (Chapter V.4) deploying the dry particle rubbing method of Park et al. [151]. With low AR ellipsoids, a crystalline monolayer of outstanding domain-size (several cm²) was accessible. Higher AR particles did not order to crystallinity but exhibited preferred orientations over the whole sample. This preferred orientation was 90° to the rubbing direction but surprisingly twisted by 90° with the highest tested AR (AR 3.3).

In the next sub-chapter (Chapter V.5), the rubbed, highly ordered layers of spherical and low-AR (AR 1.4) particles were opened-up using the elastomer substrate stretching method. The theoretic calculations that showed the possibility of controlled lattice position transfer for ellipsoidal and spherical particles by calculated phase diagrams (of Chapter V.1.4.5) could, therefore, be backed up with experimental data. Particle monolayers were transferred to square lattices (for spheres and ellipsoids) and hexagonal open-packed lattices (for ellipsoids). Further, it was noted that ellipsoids could turn during stretching when applied not in or perpendicular to the direction of stretching.

The last sub-chapter (Chapter V.6) displayed a process for creating open crystalline ellipsoidal-like particle monolayers. Hereby, different methods of the previous chapters were used in a new order (first open-packing, then stretching). The particles in the layer showed deformation depending on the position of other particles (compare to Chapter IV.8).

V.8. Supporting Information to Chapter V

V.8.1 SI to Chapter V.1.4.5 – Transfer Phase Diagrams

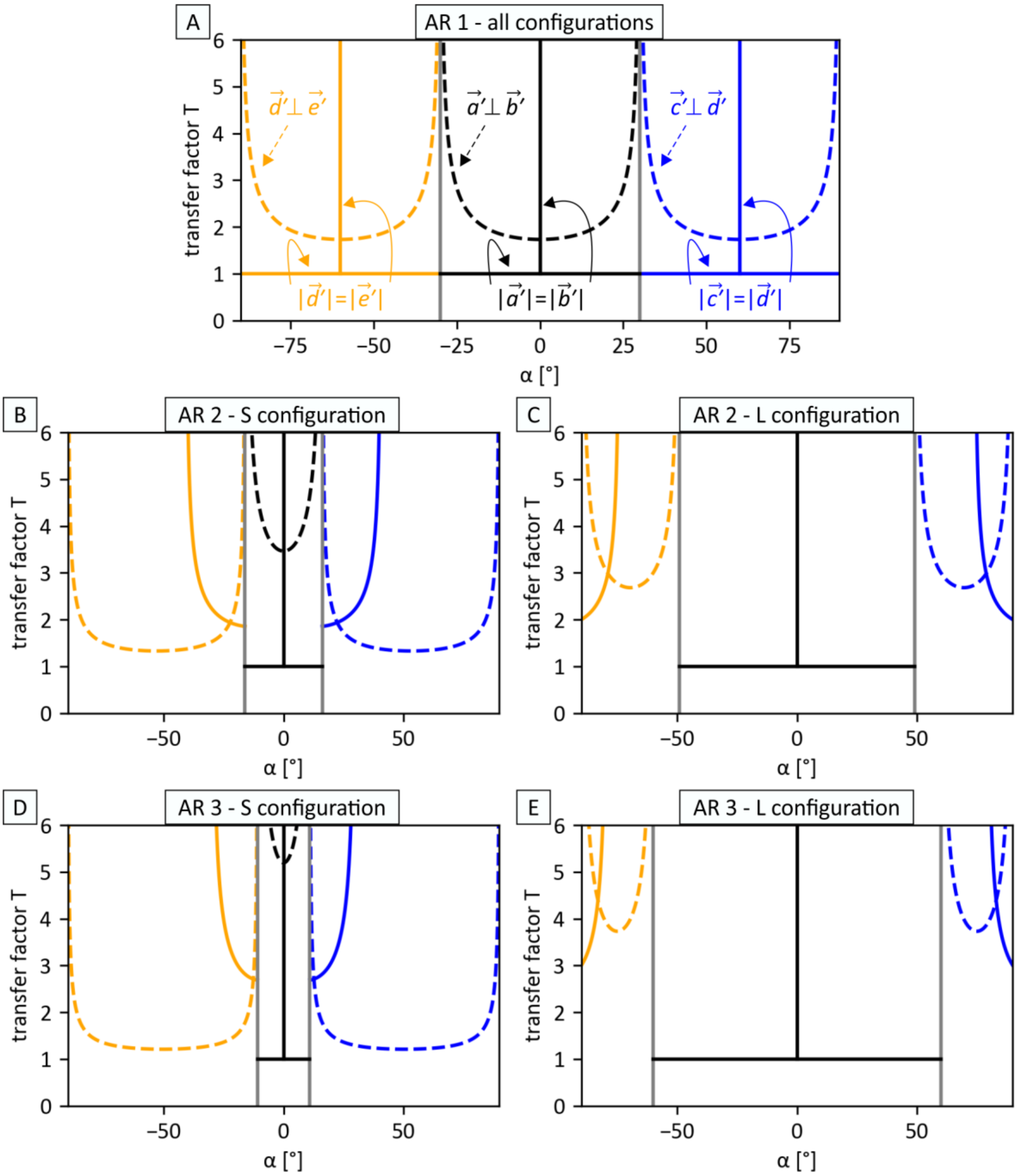


Figure SI V-1. Transfer factor phase diagrams of the two discussed starting lattices (S and L configuration) with different ARs. A: Hcp transfer with D = x and β = 60°. As D = x, A is valid for both configurations as S = L. B: close-packed ellipses starting from configuration S with AR 2 and D: with AR 3. C, E: close-packed ellipses starting from L configuration with AR 2 (C) and 3 (E). Note that there is no possibility for AR > 1.73 to have a square lattice by altering the vectors $\vec{a}$ and $\vec{b}$. In all diagrams, the dashed lines depict the plots for orthogonal vectors while solid lines show the equal length of the respective vectors. See Figure SI V-2 for simulation of transfer to the square lattice points.

V.8.2 SI to Chapter V.1.5 – Simulation of CP Ellipses Transfer

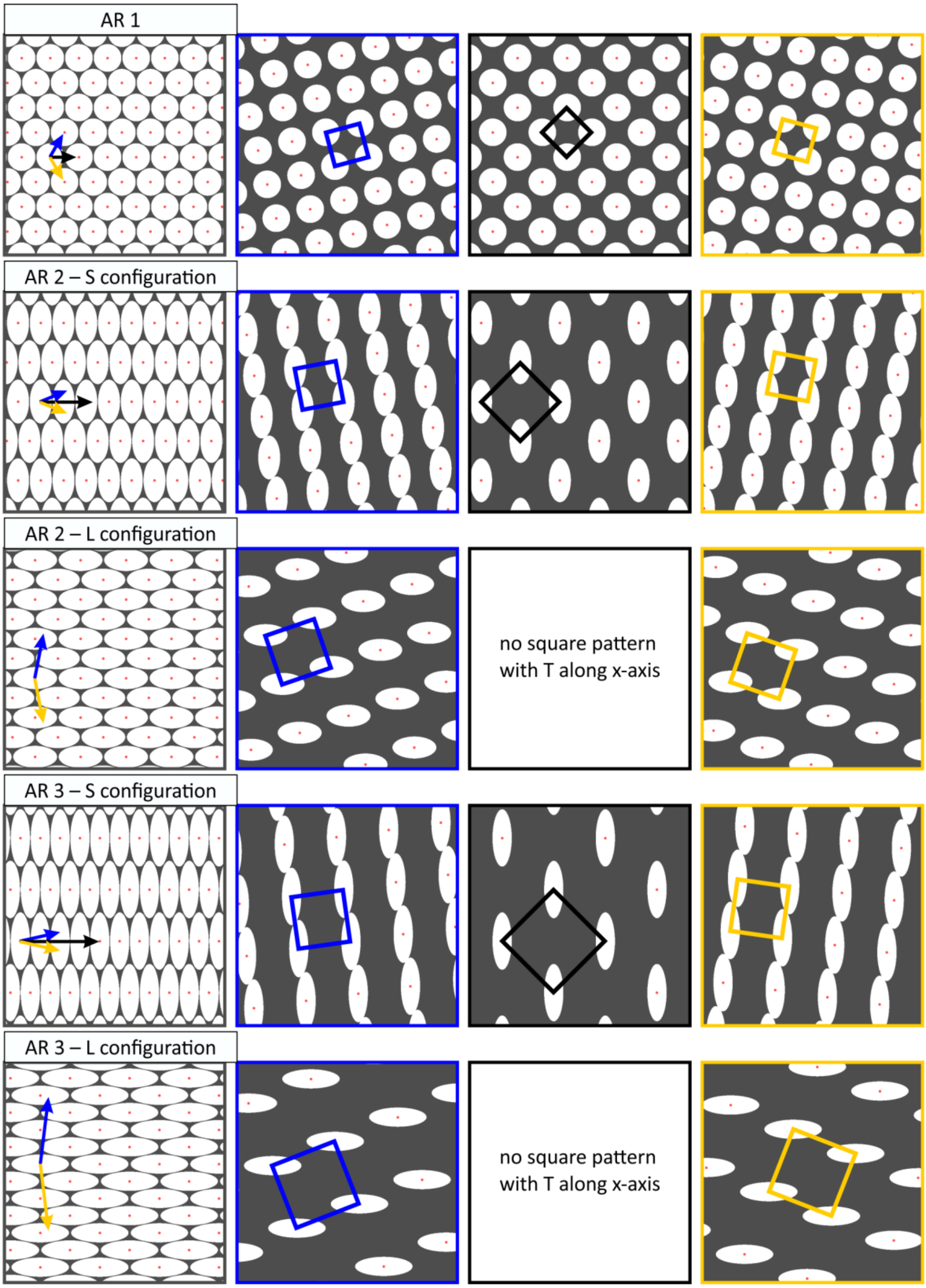


Figure SI V-2. Simulation of transferred ellipses with different AR (1, 2 and 3) and starting configuration. Squared lattices are found with the parameters of the intersection of equal-length and orthogonal spanning-vectors of the phase diagram from the previous SI Chapter V.8.1 (Figure SI V-1). Arrows display the directions of the respective transfer vectors. Hereby, the transfer vector length is shown by the ratio of the arrow length.

V.8.3 SI to Chapter V.2.2.2.1 – Devices for *VOGEL* Method

The quality of the scooped particle monolayers by the *VOGEL* method were enhanced using e.g. shielding against cross flow. Thus, a holder for glass slides (Figure SI V-3 A) was manufactured which could be partially immersed into the sub-phase (see Figure SI V-3 A particle covered area). The ramp for preparing the monolayer was then mounted between the two glass slides.

Additionally, diving particles were not beneficial for the quality of the scooped monolayer. Therefore, a small aquarium was manufactured which can be immersed into the sub-phase (Figure SI V-3 B) and had one side lower than the others. The sub-phase was adjusted so that there was just a connection over the lowered side (Figure SI V-3 B front), forming a U-shaped shield around the ramp. The monolayer could then float over this lowered side onto the surface of the surrounding basin. Excess particles that submerged into the sub-phase were mostly staying in the aquarium and did not influence the scooping of the monolayer.

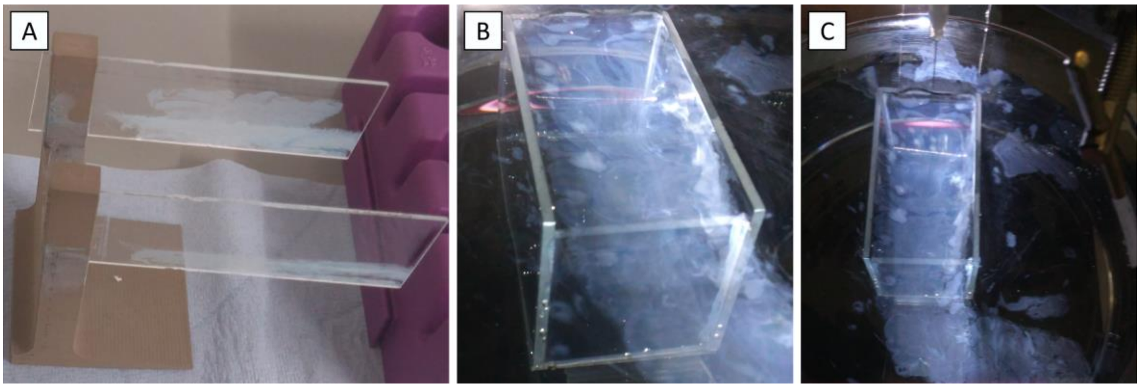


Figure SI V-3 Photographs of devices sometimes used during the particle monolayer formation by the VOGEL method. A: Shielding cross flow device and B, C: separation aquarium. C: The needle of the syringe and the glass slide ramp can be seen in the upper part of the photograph. The dispersion is dispensed on the ramp while the ramp is between A: two glass slides or B, C: inside the aquarium. The monolayer migrates out from the confined space (e.g. at image C, over the lowered side of the aquarium). A: Glass slide dimensions: 7.6 cm × 2.6 cm and B, C: aquarium 3 cm × 5 cm × 4 cm (width × length × height). No scale bars due to perspective views.

V.8.4 SI to Chapter V.2.3.1 – Monolayers by *RETSCH* Method

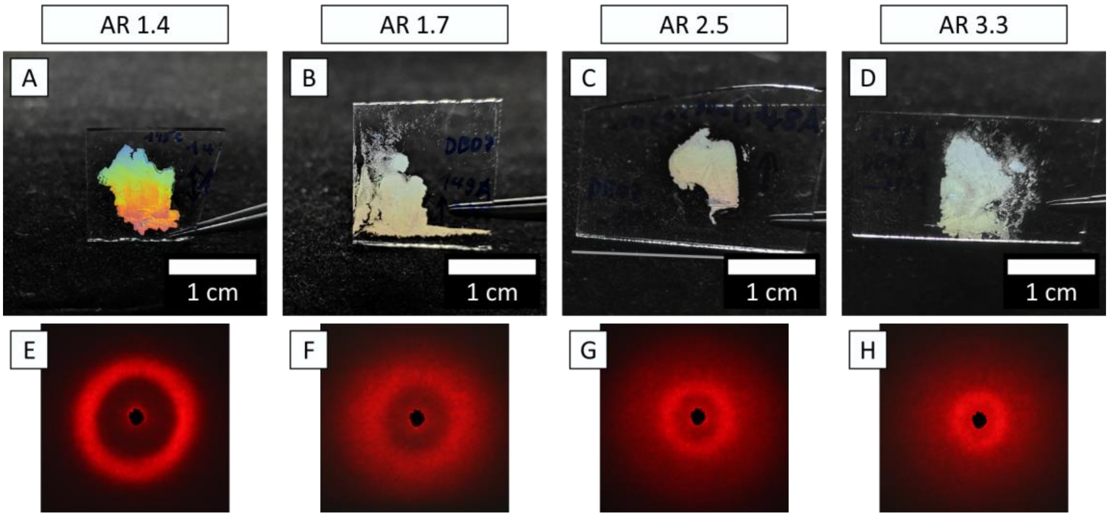


Figure SI V-4. Monolayers assembled by the RETSCH method from different stretched particles. A – D: mobile phone picture with illumination from the back by white light (scale bar estimated). E – H: The respective laser diffraction image of the monolayers. Samples from left to right: AR 1.39 (DB13 15 %_1), AR 1.74 (DB07 25 %), AR 2.45 (DB07 50 %), AR 3.26 (DB07 75 %).

V.8.5 SI to Chapter V.2.3.2 – Monolayers by *VOGEL* Method

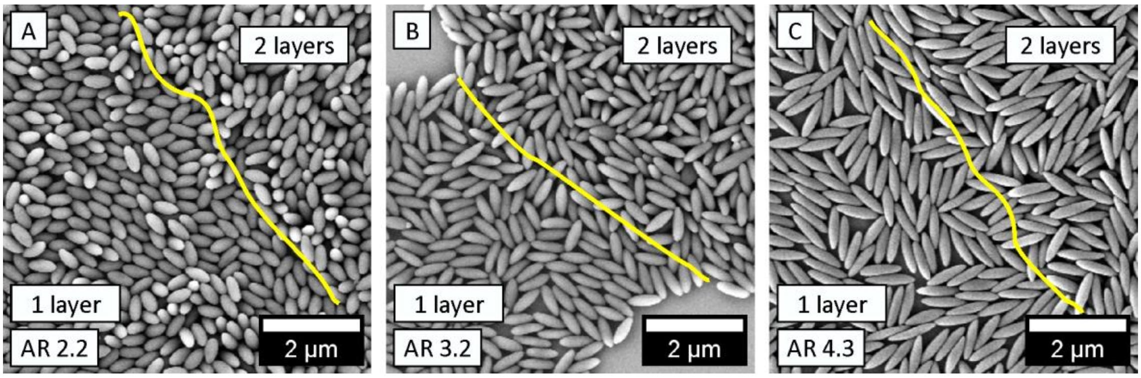


Figure SI V-5. Monolayers made from small particles by the VOGEL method often showed pushed on particles. Sometimes those pushed particles stand up or form multiple layers (yellow lines separate monolayers and bilayers). A: AR 2.19, B: AR 3.19, and C: AR 4.28.

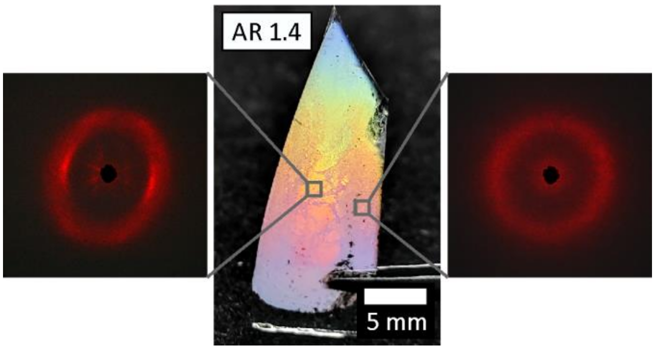


Figure SI V-6. Monolayer of big AR 1.42 (DB13 15 %) particles made by the VOGEL method. Different laser diffraction patterns were visible. Left: highly ordered, Right: ordered less. Scale bar estimated.

V.8.6 SI to Chapter V.3.1 – Rinsing and Transfer Factor

After retraction of the substrate, it is beneficial to rinse the layer with water to clean the sample from particles on top and from ingredients of the sub-phase (e.g. SDS or NaOH). A rinsing with ethanol is not recommended, as this led to the destruction of the transferred monolayer (Figure SI V-7).

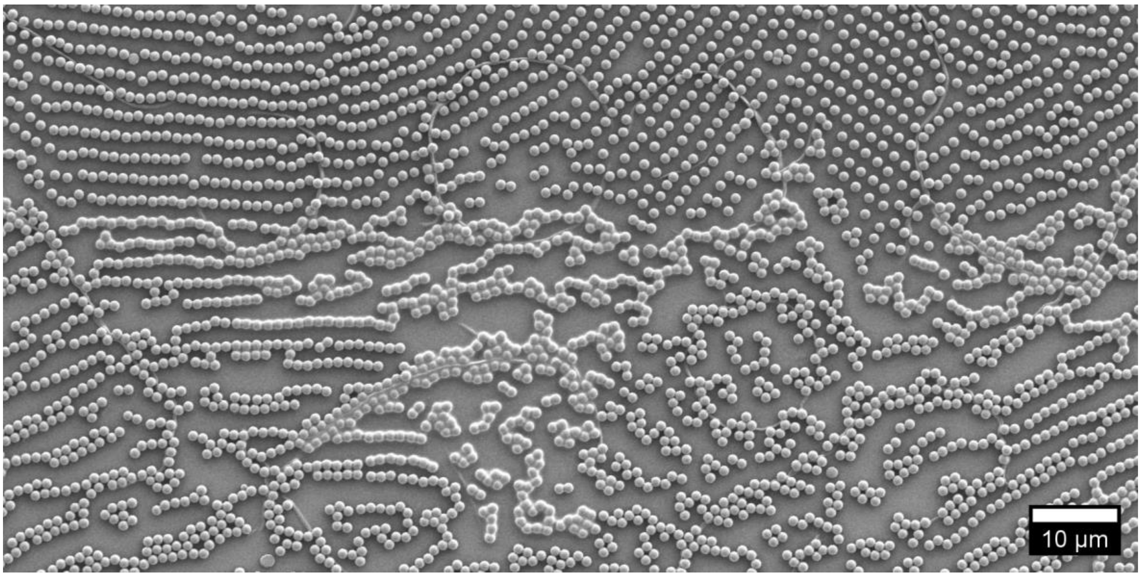


Figure SI V-7. Monolayer of spherical particles transferred. The upper part is just rinsed with water, while the lower part is rinsed with ethanol. Ethanol destroyed the lattice.

V.8.7 SI to Chapter V.3.1.1 – Estimated Transfer Factor

The transfer factor T was roughly estimated via Fiji/ImageJ measurements (Figure SI V-8) and Equation SI V-1.

$$x_{position\ on\ image} = T \cdot x_{previous\ position} \qquad \text{(Equation SI V-1)}$$

The distance of the particle center from the starting point is x. The starting point is therby set on an estimated not transferred point. The not transferred point was e.g. between two still touching particles. Additionally, as the SEM images were taken with transfer directions along y-axis, the not transferred point can be estimated to lay on a horizontal line between two neighbouring particles. To estimate the previous particle position, a particle representative is “moved” (Figure SI V-8 green circle) close to the starting point until it gets into contact with the not transferred particles forming an hcp lattice. The measurements were not exact as the previous particle positions were just estimated.

The resulting transfer factor lay roughly between 2.3 (Figure SI V-8 A) and 3 (Figure SI V-8 B). However, the angle dependency in Figure SI V-8 B, where the particle contact was not directly in transfer direction, was not taken into account for this rough estimation. Thus, the transfer factor should decrease from the measured 3 down to 2.x.

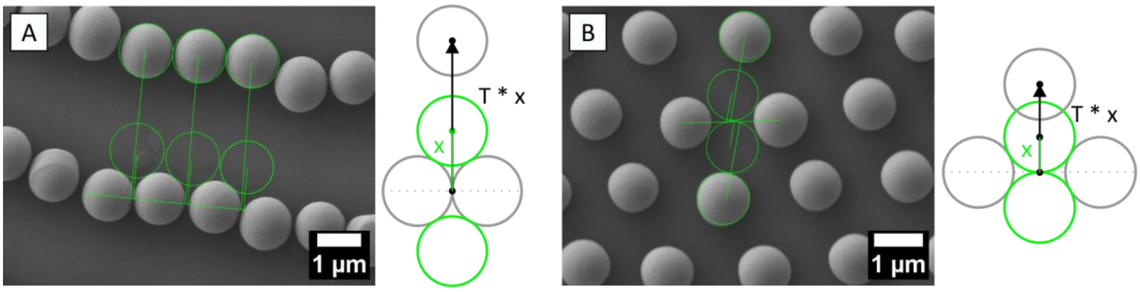


Figure SI V-8. Estimation of the transfer factor using Fiji/ImageJ measurements on the same sample but with different lattice orientations A and B. The drawings next to the SEM images (A and B) show the difference of transfer with the same factor (here 2.5) on the resulting position. The transfer factor measured without an angular correction lay in between 2.3 and 3.

V.8.8 SI to Chapter V.3.3 – Plots for Transferred Monolayers

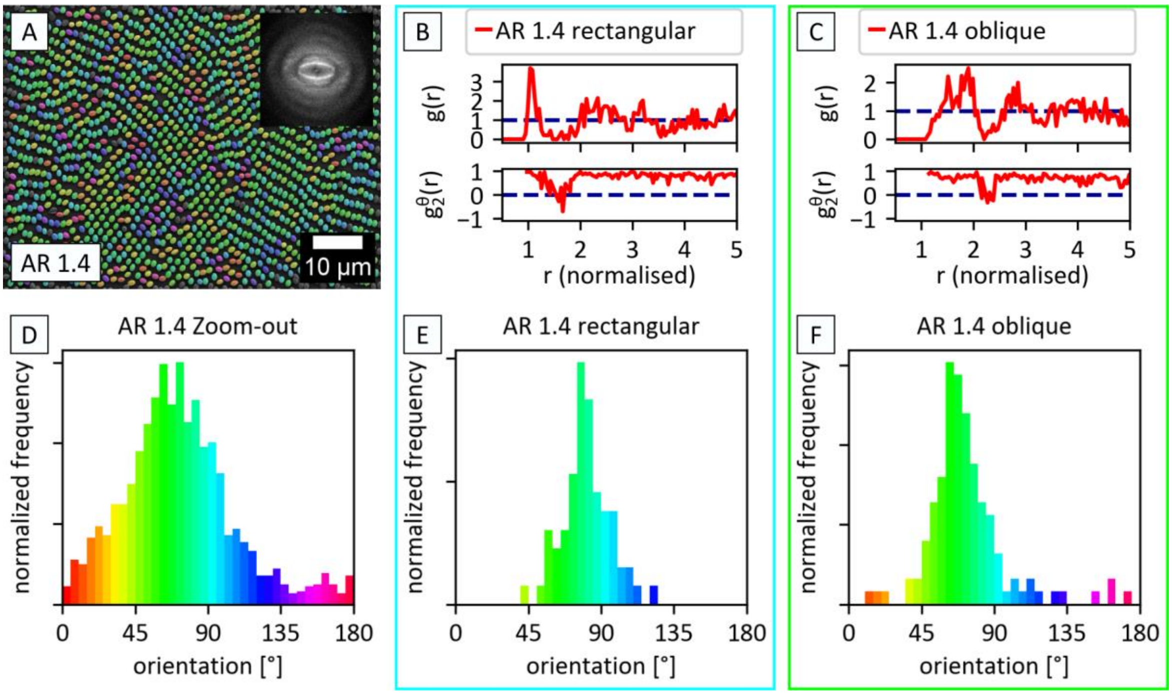


Figure SI V-9. Additional image and plots for transferred ellipsoidal particle monolayers of AR 1.42. A: Zoomed-out SEM image with inset FFT and angle dependent coloration (histogram at D). The FFT is very similar to the laser diffraction (compare to Figure V-28). B, C: Pair and angular correlation functions of the two grains (rectangular (cyan) and oblique (green)) depicted in Figure V-28 C – E. The angular distribution is close to 1 even at long distances. D – F: The angular distribution histograms show a preferred orientation of the transferred particles in the direction of transfer.

V.8.9 SI to Chapter V.4.4.3.1 – Pressure Dependent Rubbing

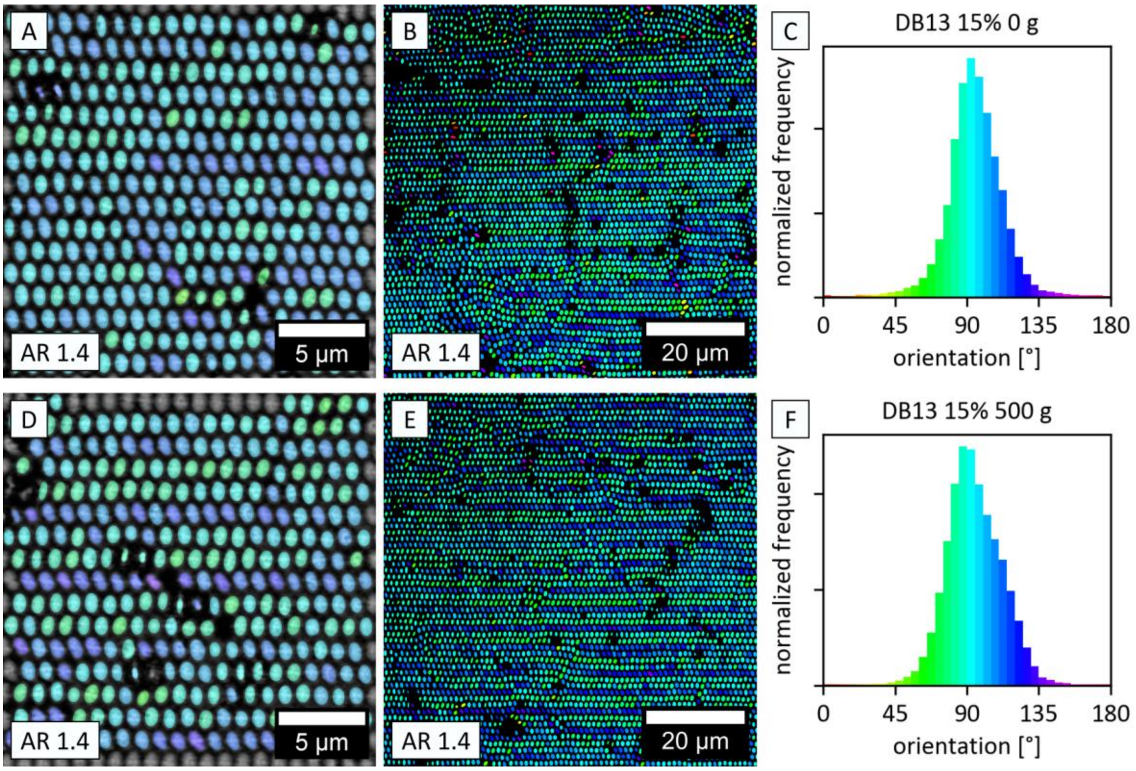


Figure SI V-10. Rubbed particle monolayers of AR 1.4 (DB13 15 %_2) at different rubbing forces. Samples were made on the semi-automated rubbing device 3 with 3 cm/s and additional weight of A – C: 0 g and D – F: 500 g. Rubbing direction left-to-right and vice versa. Higher force did not lead to a better alignment. A, D: Zoom-in of large area (B, F). The histograms (C, F) include data of more than 40 000 particles from multiple areas of the sample. Histogram areas are normalized to 1.

V.8.10 SI to Chapter V.5.2 – Square Lattice of Spherical Particles

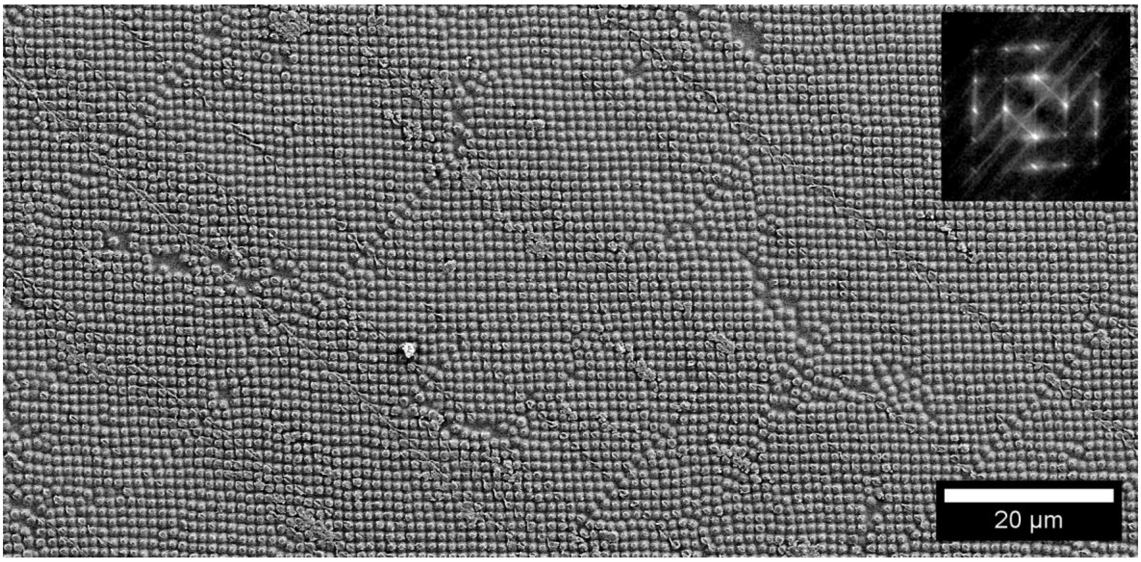


Figure SI V-11. Zoomed out SEM image of square lattice particle monolayer. Lattice spacing of around 1.2 µm (by FFT inset). Particles: DB13 0% (1.15 µm)

V.8.11 SI to Chapter V.5.3 – Calculated Values of Different AR

The values for the transfer factor T displayed in Table SI V-1 were calculated using the equations of Chapters V.1.3 and V.1.4.

Table SI V-1. Calculated values for transfer factor T of special points in the phase diagram for the PDMS stretching experiment in direction 1 of particle contact (with $\alpha = 0°$).

Resulting lattice	AR	T at start from S	T at start from L
Square	1	1.73	1.73
	1.42	2.46	1.22
	2	3.46	-
	3	5.20	-
Hexagonal	1	1 and 3	1 and 3
	1.42	1.42 and 4.26	2.11
	2	2 and 6	1.5
	3	3 and 9	1.0

Formulas of PDMS elongation in x and compression in y with position transfer factor (T). Formulas derived by linear fit at data from Figure V-48:

$$elongation\ in\ x\ [\%] = \big((0.5546 \cdot T +\ 0.4656) - 1\big) \cdot 100\ \% \qquad \text{(Equation SI V-2)}$$

$$compression\ in\ y\ [\%] = \big((-0.1872 \cdot T\ +\ 1.1669) - 1\big) \cdot 100\ \% \qquad \text{(Equation SI V-3)}$$

V.8.12 SI to Chapter V.6.2 – Images of Recovered Monolayers

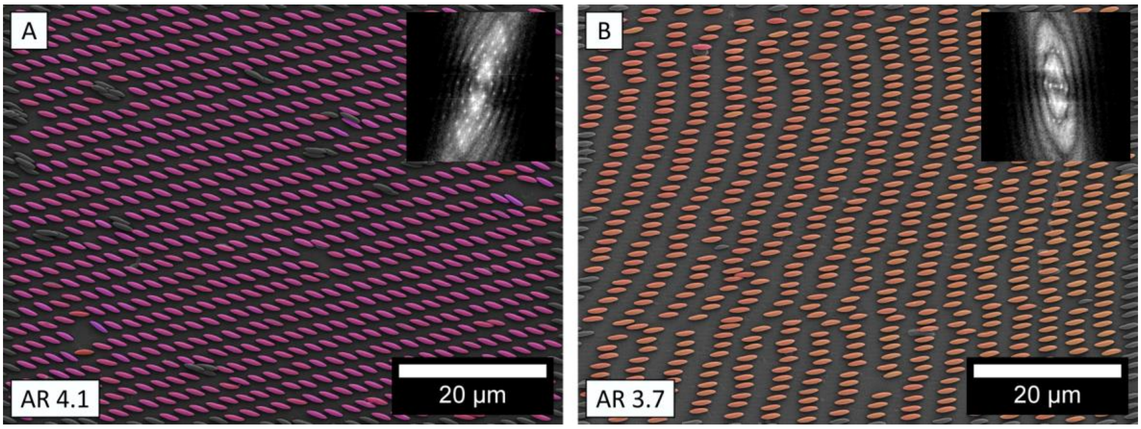


Figure SI V-12. Further SEM images (and FFT insets) with color overlay of large domains from the same recovered particle monolayer. Although same monolayer stretching (100 %), ARs of particles in images differ with the lattices. Lattice in A is similar to Figure V-54 B and in B similar to Figure V-54 C. Slight AR differences to Figure V-54 because of enlarged datasets.

Chapter VI. Application Microsieves

The following chapter shows findings of a research project that derived from a cooperation with the chair of Physical Chemistry of Chemnitz University of Technology (TU Chemnitz).

The cooperation was initiated by the PhD student Nadine Schwaar, M.Sc. and myself, after meeting at the conference Zsigmondy 2019, and got financed for 9 months by the Deutsche Forschungsgesellschaft (German research foundation, DFG).

The aim of the cooperation was to use my ellipsoidal particles within a process developed in the group of Professor Goedel, TU Chemnitz, in 2002 [167]. There, Xu and Goedel made microsieves (thickness down to 40 nm) for water filtration by using particle monolayers as a mask (float-cast microsieves).

In the standard process, the dispersion, which is spread on a water surface for assembly, contains hydrophobic monodisperse silica particles. The particle monolayers are prepared on a *LANGMUIR* trough, controlling the assembly by movable barriers [167-169], or by self-assembly, when covering the complete water surface of a petri dish [170-172]. The spreading dispersion contains a UV curable monomer. After the monolayer formation and solvent evaporation, the UV monomer is located in between the particles. Ideally, the monomer does not wet the particles, so it will form a continuous film between them. The monomer is then UV cured. The resulting membrane can be transferred from the water surface e.g. after printing a reinforcement structure on it [169] or by making a hierarchical structure of two differently sized microsieves [171, 172]. Then the silica particles inside the membrane are dissolved (by hydrofluoric acid). The resulting freestanding membrane, with holes instead of particles, is called a microsieve.

Theoretic calculations by Professor Goedel showed a possible advantage in using ellipsoidal particles as a mask. The resulting elliptical pores should show a higher filtration capability (higher permeance at lower size-cut-off) compared to circular pores [2].

Structure of the chapter

During the cooperation, I provided several batches of PS particles and ellipsoids by synthesis and stretching (see Chapter VI.1). The particles were then used by Nadine Schwaar at TU Chemnitz to prepare the membranes and microsieves (Chapter VI.2 and Chapter VI.3) and to test them on their permeance for water filtration [2].

Because of some difficulties in applying the established membrane preparation process to the ellipsoidal particles, I tested an alternative embedding system to form particle-loaded membranes (Chapter VI.4) that can be converted to microsieves (Chapter VI.5).

The chapter closes with a summary and a future perspective (Chapter VI.6).

VI.1. Particles for Microsieves

All particles used for the cooperation were stretched in the oven by the hand-stretching device. The foils were first prepared on glass plates (1.4 µm test particles with positive charge PS+, synthesized by Dr. A. Neuhöfer according to Chapter II.1.3.2, batch 0, images and data can be found in the supporting information VI.7.1) and later with the automated foil doctor blade setup.

A batch of monodisperse negatively charged PS particles (batch 1) in the size of 1.42 ± 0.02 µm (SEM evaluation) was bought at micro particles GmbH (Lot: PS-F-KM492-1). The other original spherical particle batches (batch 3, DB12, 0.95 ± 0.01 µm and batch 4, DB07, 1.06 ± 0.03 µm) were synthesized by myself in dispersion polymerizations with negative surface charge (PSS) according to Chapter II.1.3.3.

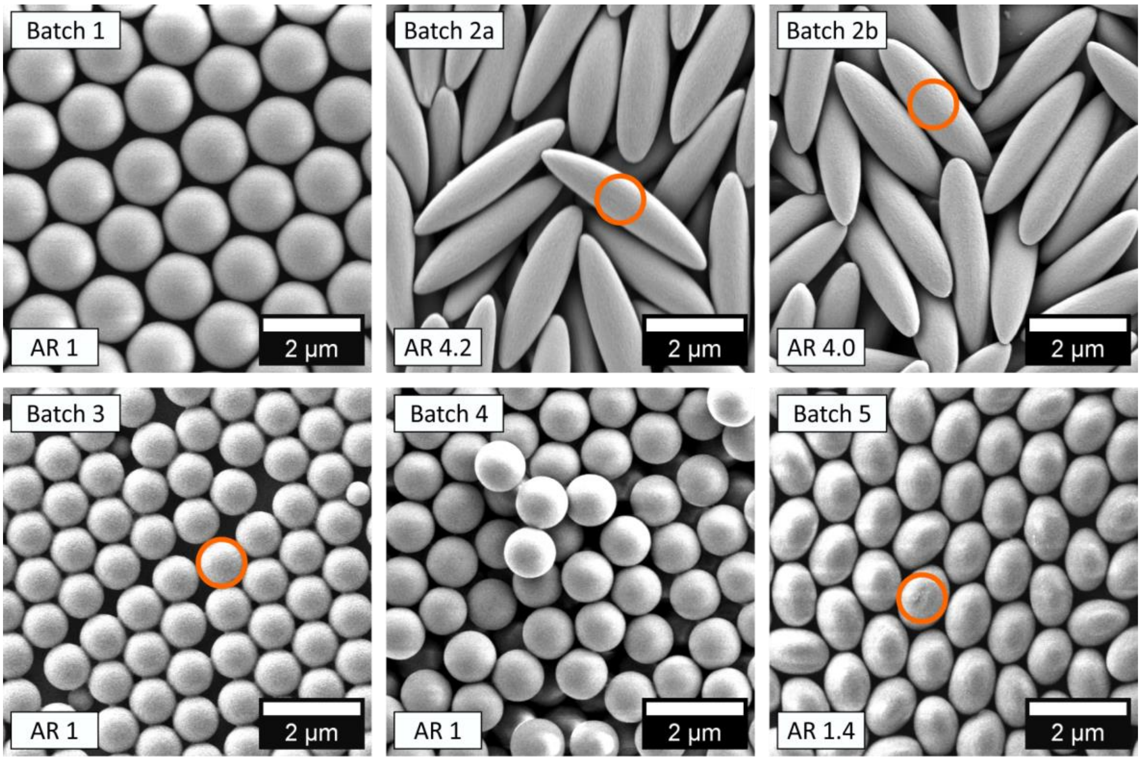


Figure VI-1 SEM images of the different particles used. Average particle diameter of batch 3 (0.94 ± 0.03 µm) is equal to minor diameters of batch 2 and 5 (orange circles). Test batch 0 can be found in the supporting information VI.7.1.

To compare the filtration properties of elliptical pore microsieves with standard circular pore microsieves, ellipsoids with short axes of the same size as spherical particles needed to be prepared. Due to the adjustment possibility of the AR and, therefore, the short axis, the resulting batch 2 (a: AR 4.2, and b: 4.0) and batch 5 (AR 1.4) had a short axis of around 0.95 µm, which was the average diameter of the spherical particles at batch 3 (Table VI-1). Batches 1 and 4 were the source of batch 2 and 5. The stretched particles showed restored zeta potentials, thus, were regarded to have no residual PVA on their surfaces. However, to render the spherical particle surface for embedding tests similar to the stretched particles, batches 1, 3, and 4 were embedded in PVA and heated in the oven as well (batches 1_h, 3_h and 4_h).

Table VI-1. Particle properties of the particles used. The short axis of the stretched particles batch 2 and batch 5 were the same size as batch 3 (grey shaded). Batches with "_h" were heated unstretched particles (0 %). Batch 1 (MP) consisted of bought particles from micro particles GmbH and stretched to batch 2a/b (100 %). Particle batch 4 (DB07) was stretched to batch 5 (15 %). ARs of spherical particles were 1.0 ± 0.0 for all batches, thus noted here as 1 without deviation.

Particles	**batch**	**stretching**	**synthesis**	**AR**	**long axis [µm]**	**short axis [µm]**	**zeta potential**
MP	1	-	bought	1	1.42 ± 0.02		-82 mV
	1_h	0 %	oven	1	1.42 ± 0.05		-77 mV
	2a	100 %	oven	4.2 ± 0.6	3.96 ± 0.34	0.95 ± 0.06	-61 mV
	2b	100 %	oven	4.0 ± 0.5	3.77 ± 0.26	0.96 ± 0.06	-63 mV
DB12	3	-	disp. pol.	1	0.95 ± 0.01		-63 mV
	3_h	0 %	oven	1	0.94 ± 0.03		-57 mV
DB07	4	-	disp. pol.	1	1.06 ± 0.03		-64 mV
	4_h	0 %	oven	1	1.05 ± 0.03		-63 mV
	5	15 %	oven	1.4 ± 0.1	1.33 ± 0.06	0.94 ± 0.03	-55 mV

VI.2. Microsieve Preparation Process

At TU Chemnitz, the following steps were conducted by Nadine Schwaar [2].

The watery particle dispersions were transferred to ethanol. For the assembly on water a spreading dispersion was prepared by mixing iPrOH and an UV curable monomer/initiator system in a predefined ratio to the particles in ethanol. The main UV monomer/initiator system used at Chemnitz was trimethylolpropane trimethacrylate (TMPTMA) with photo initiator Omnirad TPO-L.

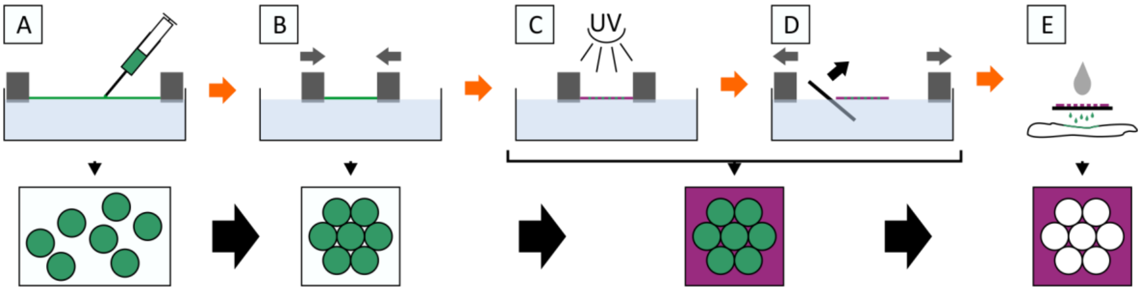


Figure VI-2. Scheme of the microsieve preparation process. A: The dispersion of particles, spreading agent and UV monomer is dispersed on a LANGMUIR trough. B: After compression a close-packed layer forms. C: The UV monomer is polymerized to form a membrane with particles inside the polymer layer. D: The membrane is scooped from the water surface by a TEM grid. E: A microsieve is prepared by dissolving the particles with toluene and flushing the residuals into a cloth or filter paper.

The spreading dispersion mixture was applied on the water surface of a *LANGMUIR* trough by direct deposition from a cannula of a syringe by hand (Figure VI-2 A). The floating particles and UV monomer/initiator system were pushed together by two barriers to form a dense layer (Figure VI-2 B). After complete evaporation of the solvents (iPrOH and ethanol), the particle layer was UV cured (Figure VI-2 C). The resulting composite membrane was transferred to TEM grids (Figure VI-2 D). The particles were dissolved by toluene and washed into a lint-free cloth or filter paper, leaving microsieves sitting on the TEM grid (Figure VI-2 E).

VI.3. Resulting Microsieves

The established methods [167, 168, 170] needed some adaption to the polymeric particles. The PS particles were not hydrophobic and, therefore, immersed in the sub-phase, leaving the waterside of the resulting hole larger compared to the airside (Figure VI-3 A and B). Additionally, sometimes the UV polymer covered the particles completely so the membrane was not opened after dissolving the PS particles template. This was especially the case with ellipsoidal particle monolayers (Figure VI-3 C and E). The voids between the particles were covered with polymer (orange arrows), but the resulting microsieves were nevertheless fragile and ripped easily. Although the particles were monodisperse, the resulting microsieves, especially the one with ellipsoids, did not have monodisperse holes.

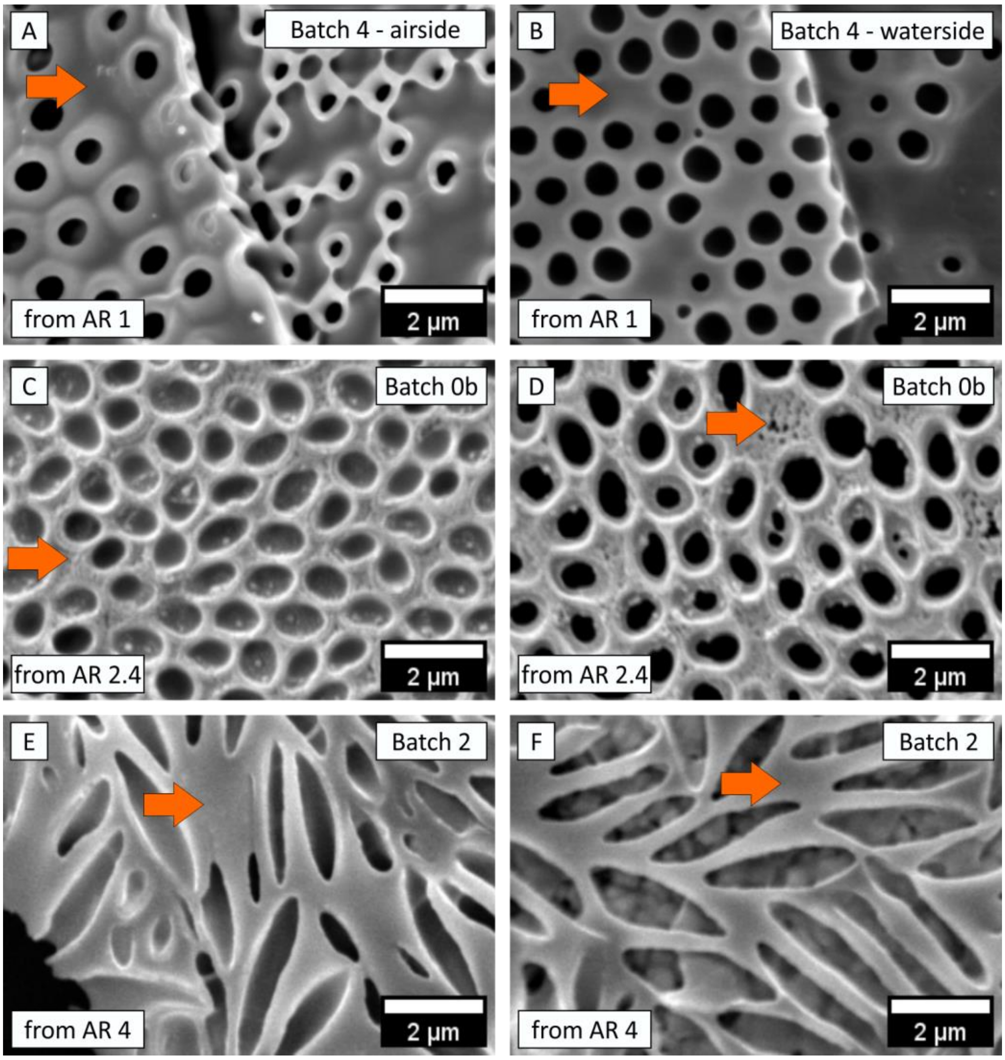


Figure VI-3. SEM images of membranes produced by N. Schwaar with my particles as a mask. Spherical particle microsieve (batch 4, DB07) imaged from A: airside and B: waterside. The holes were wider at the waterside due to immersion depth of the particles. C – F: Microsieves from ellipsoidal particles displayed elliptical pores (C, D: Batch 0b, see supporting information VI.7.1, AR 2.4, E, F: batch 2, AR 4). C, E: Some pores were closed. Holes between the particles were covered by the membrane polymer (orange arrows). Images by N. Schwaar, printed with permission of N. Schwaar, © TU Chemnitz 2023.

VI.3.1 Dissolving Particles

The crucial step for generating the pores of the microsieves was the dissolving of the PS particles. The challenge was to dissolve the PS particles without rupturing the membrane. N. Schwaar tried several methods (plasma etching) and solvents (Toluene, THF, Acetone). She found an appropriate protocol in dripping toluene on the membrane, while the dry membrane on the TEM grid was lying on a lint-free cloth or filter paper. Thereby, the toluene washed the PS away into the cloth.

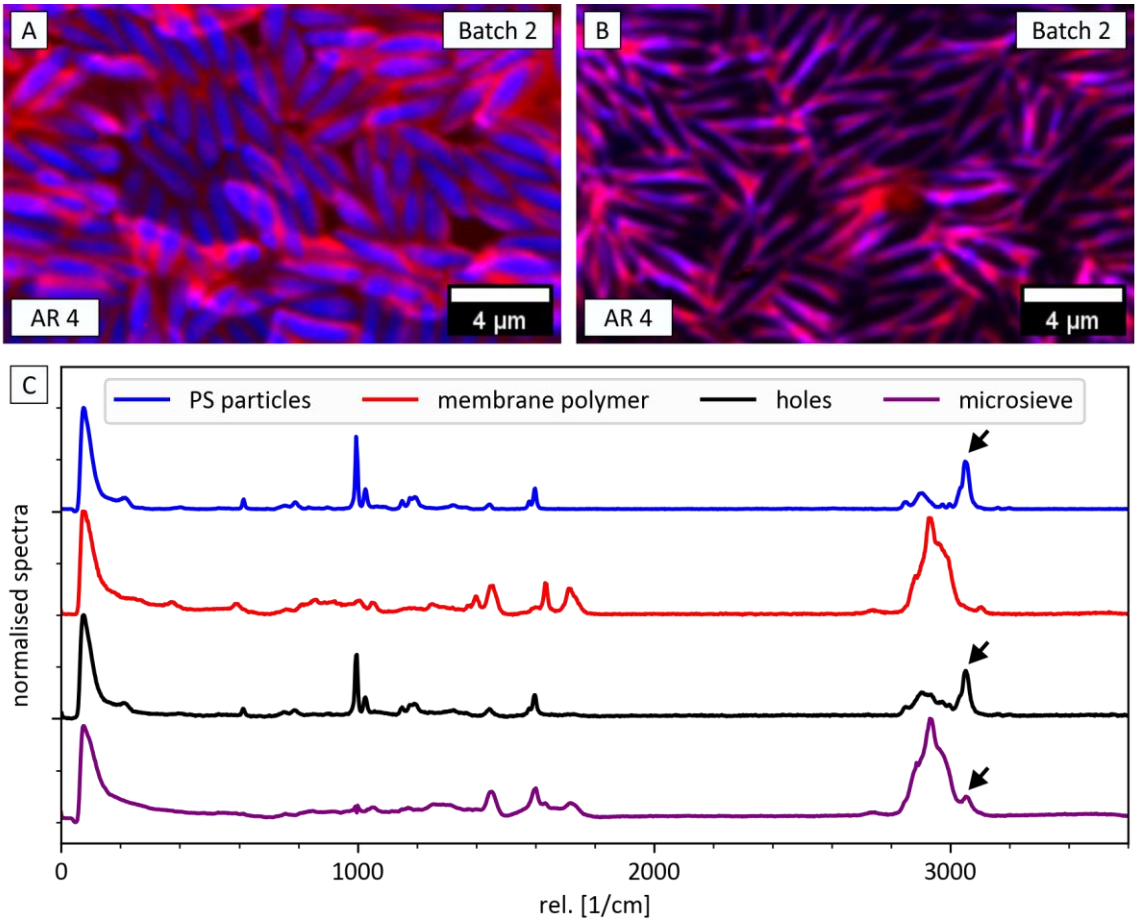


Figure VI-4. Raman spectroscopy on batch 2 membrane A: with PS particles and B: without particles. The signal for the UV cured membrane polymer was colored red, while the signal of polystyrene was colored blue. Residual PS was detected in holes and on the microsieve B, as shown in normalized spectra C (black arrows).

However, localized Raman spectroscopy at University Bayreuth (performed by myself with the help of Dr. Holger Schmalz, chair of Macromolecular Chemistry II, University Bayreuth) showed that the PS was not fully washed off from the particle membrane and that inside the pores some PS and membrane polymers remained (Figure VI-4).

After successfully opening the pores by an adjusted protocol, N. Schwaar measured the permeance of the microsieves. The new protocol and the results are summarized in a recent publication of N. Schwaar and myself [2] and will not be discussed here.

VI.4. Alternative Embedding System

When preparing the membranes, we noticed that the particles were highly immersed in the water phase. As known from literature [158], the pH value of the sub-phase can alter the immersion depth of particles (higher immersion with higher pH at PS particles with carboxylic acid groups, negatively charged). Therefore, a study to investigate the possibility of using different pH values for immersion depth regulation was done together with my internship student Pascal Reiß, B.Sc. at University Bayreuth.

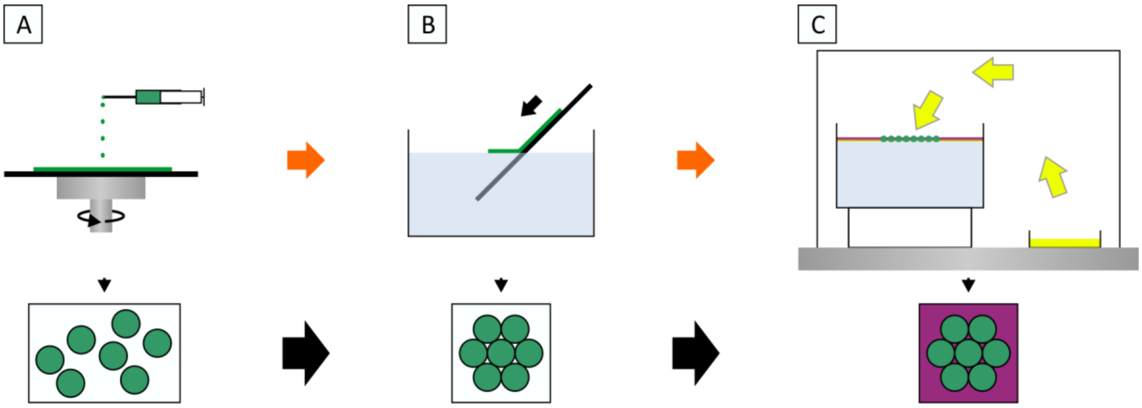


Figure VI-5. Scheme of the embedding process by n-butyl cyanoacrylate (BCA). A, B: After preparing the monolayer by the RETSCH method, the floating monolayer is brought into C: a closed container next to a petri dish filled with BCA (yellow). The BCA evaporates and reacts with the water forming a polymer layer (purple) on the water surface. The BCA migrates through the polymer layer and keeps polymerizing at the water/polymer interface. The particles are fixed in position by the polymer layer.

For investigating the immersion depth, the particles were fixed at their position after monolayer formation by an n-butyl cyanoacrylate (BCA) vapor method. The BCA vapor method was used by Vogel et al. in 2011 [158] to investigate the immersion depth at different pH with spherical particles (explicitly explained in 2014 [173]). The big advantages of the BCA method, in contrast to methods like gel trapping [144, 174, 175], is that the sub-phase stays unaltered (no gellant in the sub-phase) and it is easy to apply.

VI.4.1 Experimental

The particles used within this sub-chapter had ARs of 1.7, 2.5, and 3.3 (same particles as in Chapter V.2.3.1) and originated from batch 4 (DB07, see Table VI-1). For the investigation of spherical particles, batch 3 (PS-PSS particles, DB12) and PS-AA particles (synthesized with the same protocol as PS-PSS, but with acrylic acid instead of PSS, DB08, Chapter II.1.3.4) were used (Table VI-2).

Table VI-2. Particle properties of particles used in immersion depth evaluation. Batch number from Table VI-1 in brackets. ARs of spherical particles were 1.0 ± 0.0 for all batches, thus noted here as 1 without deviation.

Particles	batch	synthesis	AR	long axis [µm]	short axis [µm]	zeta potential
DB12	0 % (3)	disp. pol.	1	0.95 ± 0.01		-63 mV
DB07	25 %	oven	1.7 ± 0.1	1.53 ± 0.08	0.88 ± 0.04	-70 mV
	50 %	oven	2.5 ± 0.2	1.97 ± 0.12	0.80 ± 0.05	-67 mV
	75 %	oven	3.3 ± 0.4	2.40 ± 0.16	0.74 ± 0.05	-63 mV
DB08	PS-AA	disp. pol.	1	0.98 ± 0.03		-56 mV

The particle monolayers with particles of different ARs were prepared with the *RETSCH* method (see Chapter V.2.3.1) on water surfaces (Figure VI-5 A and B) with different pH values (pH 4, 6, and 11), and with and without 0.1 mM SDS as a surfactant.

The small glass containers with the monolayers were then put under a glass dome (part of a large glass beaker, sealed with a Teflon ribbon) on a heating plate next to a petri dish (Figure VI-5 C) containing the monomer n-butyl cyanoacrylate (BCA). The small glass container was spaced by a polymer petri dish from direct contact to the heating plate. The monomer was evaporated at 56 °C and polymerized in contact with water, freezing the particles in position. As the polymerization needs hydroxide ions to start (mechanism explained at Vogel et al., appendix of [158] or [173]), the reaction was slow at low pH values. Therefore, the reaction time needed to be varied at different pH levels (0.5 h for pH 11, 1.5 h for pH 6, and 17 h for pH 4). The PBCA was growing just inside the water phase leaving the airside of the particles uncovered. The monomer diffuses through the polymer to the water/polymer interface, resulting in thicker membranes with additional time.

The resulting membranes were scooped by Pascal Reiß and myself on plasma activated glass slides containing a scratch from a glasscutter. Care was taken that the monolayer was positioned directly over the scratch. For SEM investigations, the membranes on the glass slide were frozen with liquid nitrogen and broken on the scratch. The glass pieces were mounted vertically in SEM holders, fixed by screws and investigated by myself at the electron microscope at various angles. For evaluations, images perpendicular to the membranes were made.

The images were then evaluated, in terms of particle fractions standing out of the membranes, by Pascal Reiß using Fiji/ImageJ (version 1.53e).

VI.4.2 Results

The evaluation of the thin membranes was difficult, as the membranes tended to bend away from the electron beam during scanning (charging effect) or degenerated during scanning.

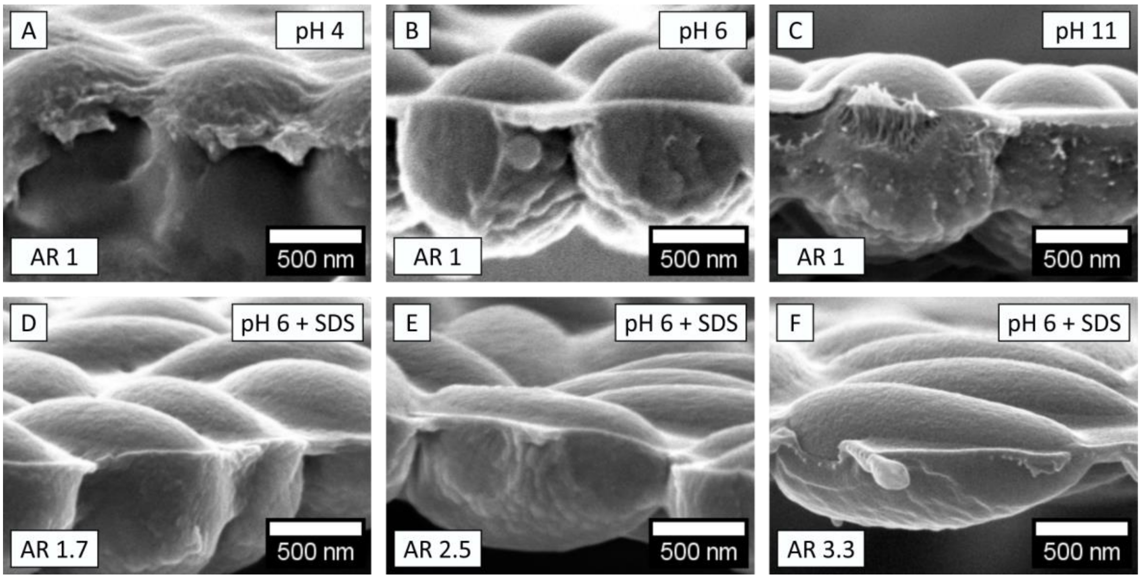


Figure VI-6. SEM images of different particles embedded in PBCA on different sub-phase: spherical particles (batch 3) on MilliQ-water with different pH (A: pH 4, B: pH 6, and C: pH 11). D – F: ellipsoidal particles of D: AR 1.7, E: 2.5, F: 3.3 from sub-phases with pH 6 and 0.1 mM SDS.

Investigations on spherical particles in batch 3 (Figure VI-6 A – C) showed that the BCA method did not work well with pH 4. The particles on those acidic sub-phases were mostly completely covered by the PBCA. Either this could be caused by particles immersed completely in the sub-phase, or by some water condensation on top of the particles due to the very long reaction time (reaction chamber was as well covered in PBCA). Remedy for condensation could be a redesign of the here used setup to separate the heating of the BCA from the monolayer container. Ellipsoidal particles were as well covered by PBCA at pH 4 (see supporting information VI.7.2). Completely covered particles were very hard to image due to charging effects.

While spherical monolayers were mostly ordered, ellipsoidal particle monolayers were unordered, leaving the embedded particles randomly oriented to the PBCA layer fracture edge (Figure VI-6 D – F). The ellipsoid’s immersion depth was calculated by measuring the height above the water/PBCA-line and by dividing this value by the dimensions of the particles from 2D SEM evaluation. To get the immersion depth, just particles on the fracture edge, where the membrane pointed towards the observer, could be used. Therefore, just a few particles per sample could be evaluated (Table VI-3).

Table VI-3. Immersion depth of the different particles according to the evaluation of Pascal Reiß. Values at pH 11 without SDS were hard to obtain due to bulging (see supporting information VI.7.3). Values in brackets were from one particle only. SEM images of the samples can be found in the supporting information VI.7.2.

Particles	batch	AR	immersion depth at pH 6	immersion depth at pH 11	immersion depth at pH 6 + SDS	immersion depth at pH 11 + SDS
DB12	0 % (3)	1	73 %	75 %	68 %	75 %
DB07	25 %	1.7	79 %	(84 %)	75 %	79 %
	50 %	2.5	68 %	(78 %)	71 %	63 %
	75 %	3.3	69 %	(70 %)	63 %	70 %
DB08	PS-AA	1	78 %	81 %	80 %	81 %

The particle-loaded membranes of pH 6 and 11 exhibited smooth shapes on the top (air-side). The bottom side (waterside) of the PBCA membranes was uneven (Figure VI-7), which was in contrast to the UV polymerization method in Chapter VI.3. I propose that the BCA monomer diffused along or through the PS particles, creating new polymerization sites below the particle (raspberry shape). This phenomenon was less visible when using SDS.

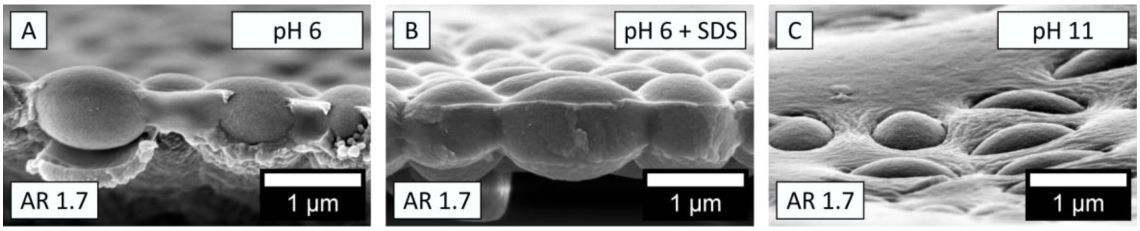


Figure VI-7. Different SEM images of ellipsoids with AR 1.7 (DB07_25 %) in PBCA layers. A: PBCA below the particles due to secondary nucleation (pH 6). B: Less secondary nucleation at pH 6 with surfactant SDS. C: Partially sunken particles at pH 11 without surfactant SDS.

Particles at pH 6 were immersed approximately 70 – 80 % in the sub-phase, leaving just 30 – 20 % of the particles exposed to air. When increasing the pH from 6 to 11, the immersion depth increased for the most of the investigated particles. This was in accordance to Vogel et al. [158]. Additionally, the contact angle seemed to increase for particles on pH 11, or the wetting of the BCA got worse. Either way, the particle's impression at pH 11 without surfactants was like partial-sunken almonds on a Christmas cookie (Figure VI-7 C). The PBCA layer between the ellipsoidal particles bulged; especially at DB07_50 % (see supporting information VI.7.3).

A slight decrease in immersion depths was visible by using SDS as a surface-active monomer. At pH 11, the wetting angle stayed low (no bulging), giving no visible difference to pH 6 samples.

VI.4.3 Conclusion of Immersion Depth Investigation

The idea for the immersion depth investigation by PBCA was to gain insight and give a suggestion for altering the particle immersion depth for a more centered embedding in the UV polymer membranes.

In general, the particles on non-basic sub-phases showed lower immersion depths than the particles on basic sub-phases. This hint at the idea that the more acidic the sub-phase gets, the more functional groups on the particles are protonated so the particles get more hydrophobic, thus lowering their immersion depth. Unfortunately, particles floating on pH 4 sub-phases could not be evaluated because of a continuous polymer layer on top of the particles that formed with the used setup. SDS could be beneficial as well for getting the particles slightly more out of the water phase. However, the results of immersion depths showed that the PS particles were in the water phase to more than 50 %. Therefore, it seems to be much more beneficial to functionalize them with hydrophobic groups, to get a lower immersion depth and center them in the UV polymer layer. However, this could not be done during my research and should be subject to further investigations.

VI.5. Proof-of-Concept: Microsieves of PBCA

A PBCA membrane of PS-AA (DB08, Table VI-2) with pH 11 on SDS was used in a proof-of-concept test to make microsieves from PBCA. As the thin PBCA membrane was not stiff enough by itself, it needed a support structure. Therefore, a successful test to dissolve the particles from the membrane with the previously described method of dripping toluene was done on nylon filter mesh (part of an E-D-Schnellsieb Nylon super-fein).

Although most of the membrane (Figure VI-8 A and B) was destroyed by the treatment with toluene (more support needed), few parts showed the expected microsieves with open pores (Figure VI-8 C).

Another test of a membrane with batch 4 particles (DB07, Table VI-1) on an alkaline sub-phase (pH 11) without SDS and therefore less wetting of the PS particles by BCA (compare to bulging of PBCA in the supporting information VI.7.3) was scooped by a TEM grid. The Membrane showed open pores just after drying on a paper towel (Figure VI-8 D). The open pores without toluene treatment were unsuspected but could be due to an arising hydrostatic pressure moving the particles out of the holes. Unfortunately, further investigations on this topic could not be conducted here.

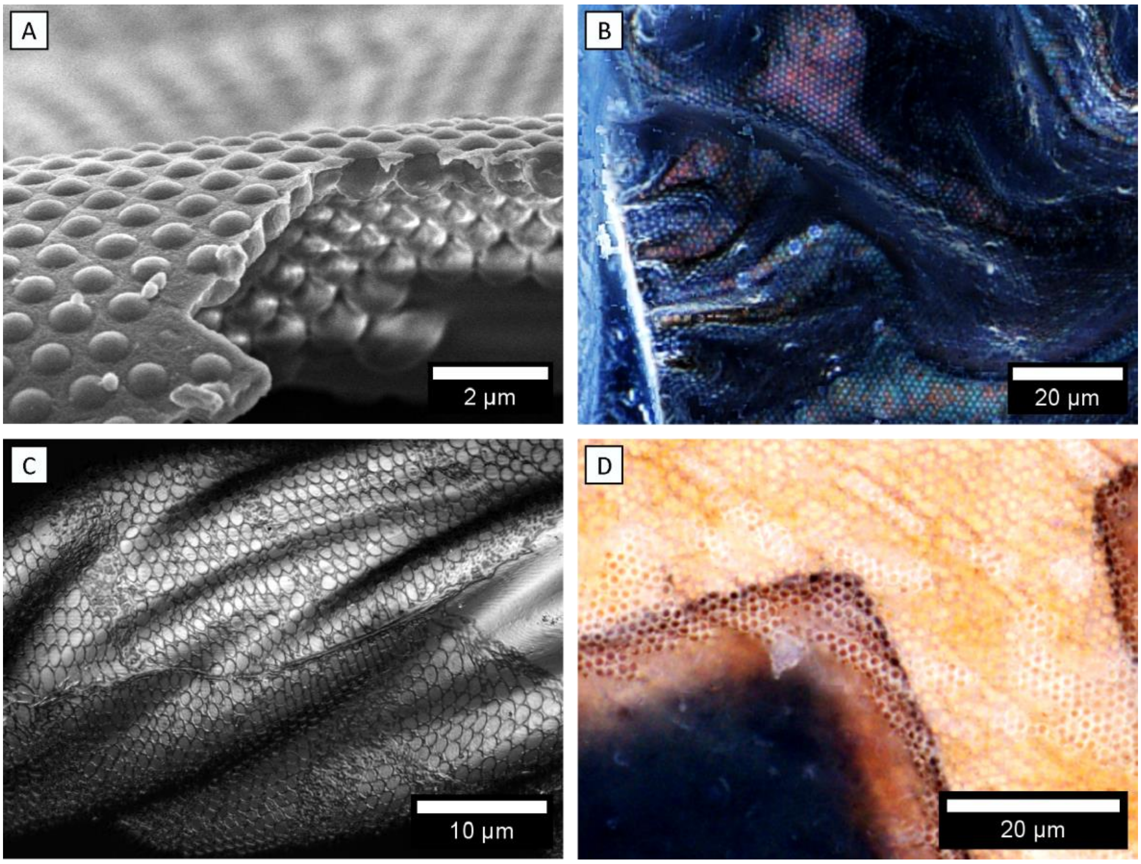


Figure VI-8. A: SEM image of a PBCA membrane (pH 11) with an hcp monolayer of spherical PS-AA particles prior dissolving, B: 3D optical image of PBCA composite membrane on nylon filter net, C: 3D laser scanning microscopy image (LSM) of a PBCA microsieve. The membrane was deposited on a nylon filter net and particles dissolved by toluene. Tension forces ripped the resulting microsieve and wrapped it around the nylon strings of the filter net. D: 3D optical image of another PBCA membrane with spherical particles on a TEM grid; A, B, C: particles PS-AA of 0.98 µm (DB08), D: particles PS-PSS of 1.06 µm (DB07, batch 4).

As seen in Figure VI-8 C, a PBCA-microsieve is very flexible as it could entwine around the nylon filter support. Therefore, one could possibly deform such a microsieve to form anisotropic evenly distributed elliptical pores that for example could be beneficial for lithography.

VI.6. Summary and Future Perspectives of Chapter VI

In this chapter, the studies done within a cooperation with TU Chemnitz are described.

The studies aimed to make microsieves with ellipsoidal particles by float-casting and to test such sieves for water filtration.

Several batches of stretched and unstretched particles were prepared with the previously described methods. The particles could then be successfully embedded at TU Chemnitz into polymer membranes by floating them on water and performing a UV polymerization of an added UV-curable monomer. The particles were immersed completely in the UV monomer so some of the holes that are gained after dissolving the particles were still covered by polymer. The opened holes show different diameters on the top and on the bottom of the microsieve due to the high immersion depth of the particles in the water phase.

At TU Chemnitz, the holes could be opened up completely by applying an adjusted protocol for removing the PS particles from the membranes. The permeance of the microsieves was then measured and the results are summarized in a recent publication [2].

Additionally, within this chapter, it could be shown by myself that a BCA vapor method is suitable to investigate the immersion depth of ellipsoidal particles in detail and even give access to microsieves. These microsieves were, however, not stable by themselves. They would need a support structure or hardening by further chemical treatment (e.g. crosslinking) which could be a task for future research.

Apart from the float-casting approach used here, it would be beneficial to use the already prepared monolayers by the rubbing method as a mask to create microsieves with elliptical pores. This was, however, not yet successful (see supporting information VI.7.4).

VI.7. Supporting Information of Chapter VI

VI.7.1 SI to Chapter VI.1 – Test Batch 0

Batch 0 was the first particle test batch of the cooperation. The ellipsoids were made from PS+ particles. Dr. Anna Neuhöfer synthesized the original spherical particles (AL166, Batch 0a). The ellipsoids were embedded in PVA on a glass plate and stretched in the oven by the hand-stretching device (see Chapter II.2.1).

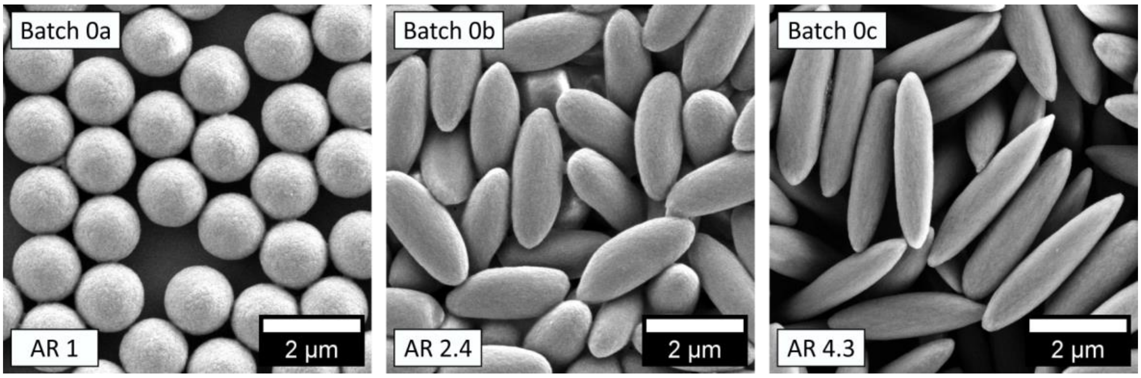


Figure SI VI-1. SEM images of test batch 0.

Table SI VI-1. Particle properties of test batch 0.

Particles	batch	synthesis	AR	long axis [μm]	short axis [μm]
AL166	0a	disp. pol.	1	1.41 ± 0.03	
	0b	oven	2.4 ± 0.3	2.52 ± 0.26	1.06 ± 0.08
	0c	oven	4.3 ± 0.4	3.66 ± 0.23	0.86 ± 0.06

VI.7.2 SI to Chapter VI.4.2 – PBCA Embedding at Different pH

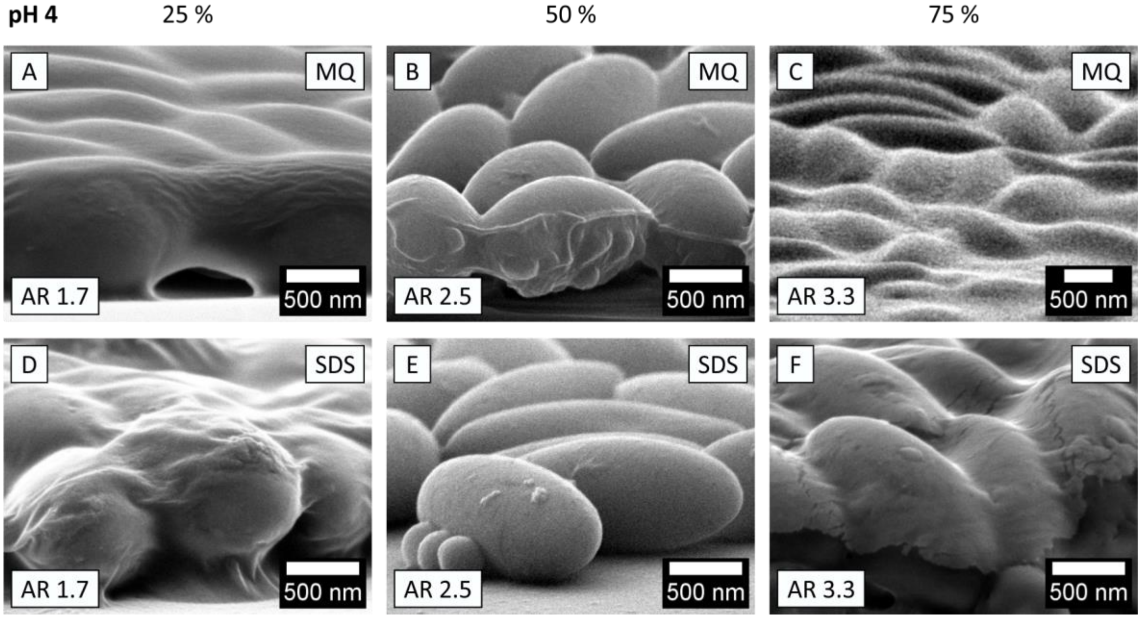


Figure SI VI-2. SEM images of membranes at pH 4. Prepared on A – C: MilliQ water and D – F: 0.1 mM SDS solution. A, D: 25 % (AR 1.7), B, E: 50 % (AR 2.5), and C, F: 75 % (AR 3.3) stretched particles. Unfortunately, the monomer at SDS 50 % (E) did not get enough reaction time to form a continuous membrane (no membrane visible).

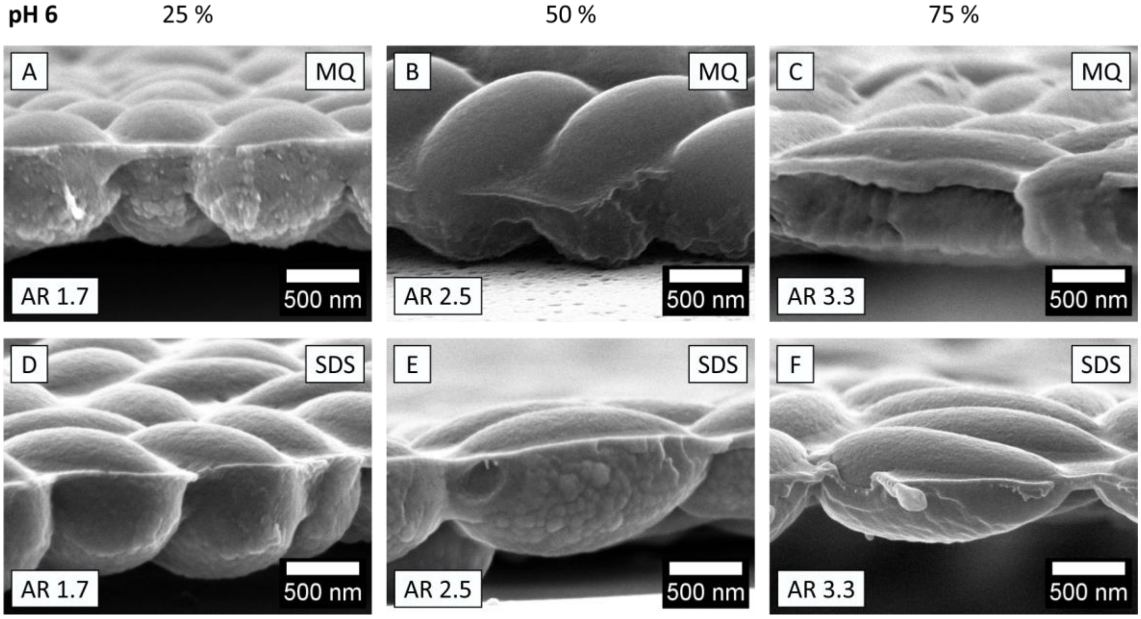


Figure SI VI-3. SEM images of membranes at pH 6. Prepared on A – C: MilliQ water and D – F: 0.1 mM SDS solution. A, D: 25 % (AR 1.7), B, E: 50 % (AR 2.5), and C, F: 75 % (AR 3.3) stretched particles.

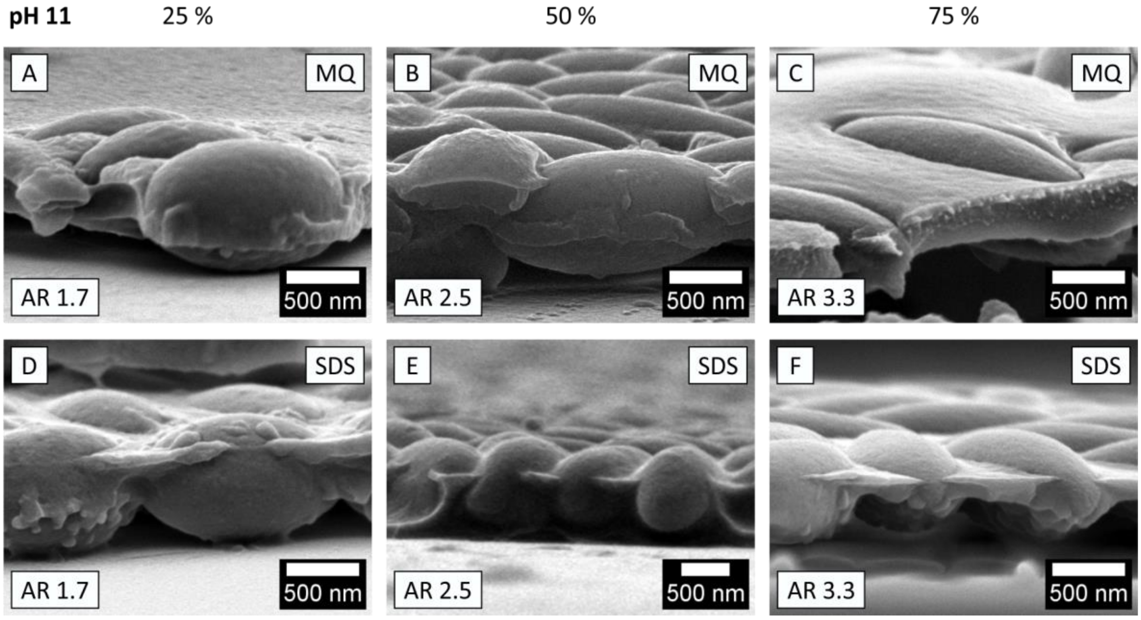


Figure SI VI-4. SEM images of particle pH 11. Prepared on A – C: MilliQ water and D – F: 0.1 mM SDS solution. A, D: 25 % (AR 1.7), B, E: 50 % (AR 2.5), and C, F: 75 % (AR 3.3) stretched particles.

VI.7.3 SI to Chapter VI.4.2 – Bulging of PBCA Layers on pH 11

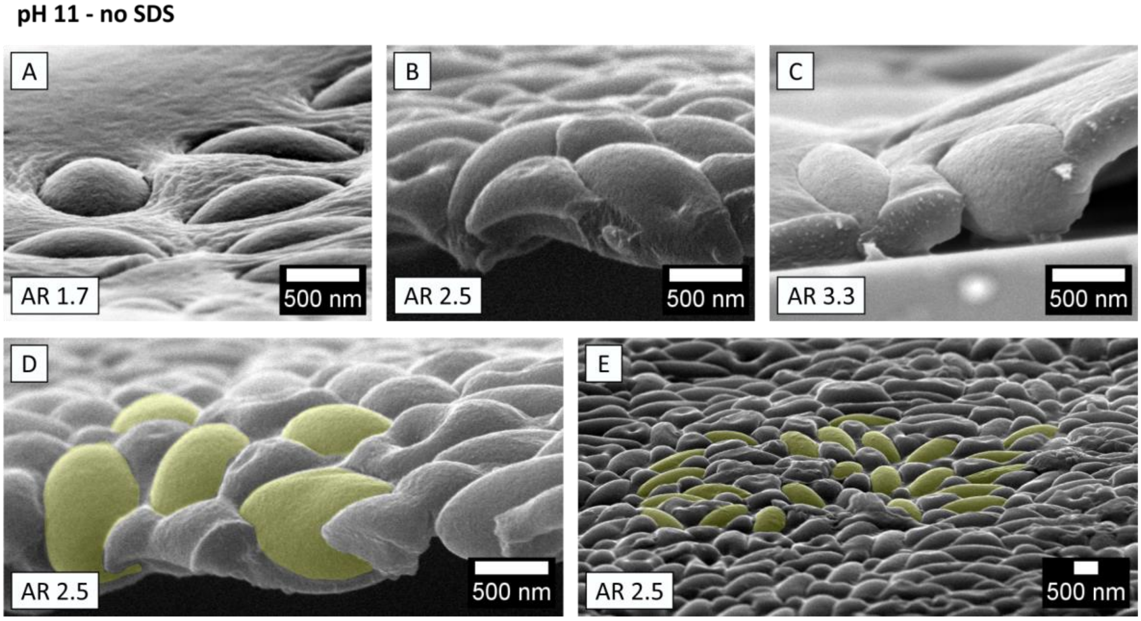


Figure SI VI-5. Membranes with different particles prepared on MilliQ without surfactant. Bulging between particles was visible. Particles: A: 25 %, B: 50 %, C: 75 %, D: 50 % with coloration of particles, E: zoom-out of membrane with 50 % particles and some particles colorized.

VI.7.4 Embedding Dry Monolayers for Microsieve Preparation

If it would be possible to apply the membrane-building polymer after monolayer formation, the rubbing method could be a versatile tool to create dense layers. A pre-test directly on PDMS rubbed particles was, however, unsuccessful as the particles were covered completely with the UV polymer (Figure SI VI-6 A).

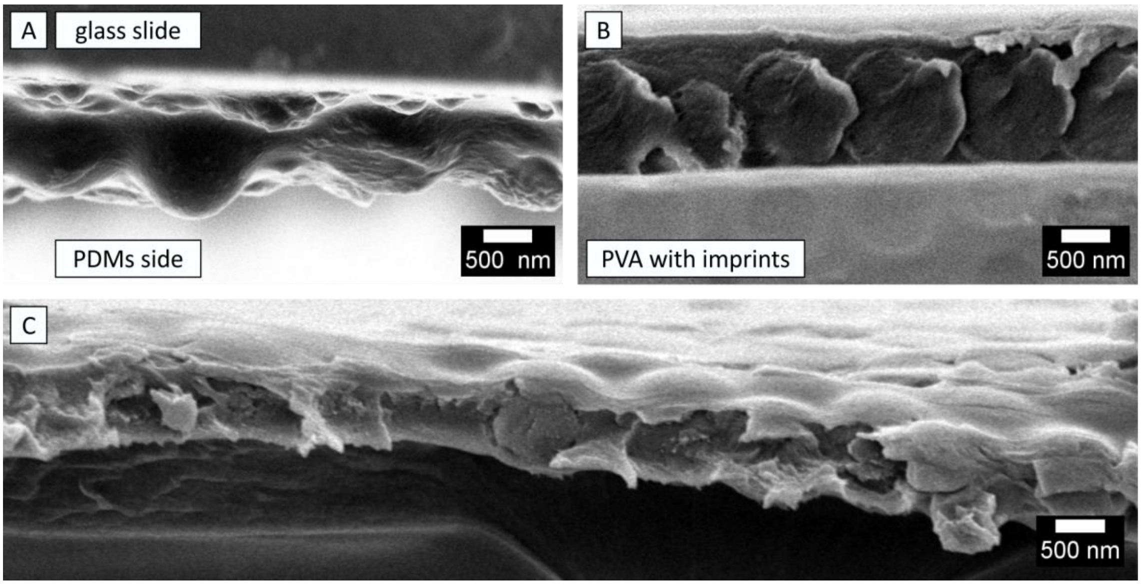


Figure SI VI-6. SEM images of A: rubbed particles were embedded in the UV polymer directly at PDMS and, taken up by a glass slide (poor image quality due to charging). The UV polymer covered the particles but left spaces in between. B, C: rubbed particles taken up from rubbing substrate with PVA and then embedded in the UV polymer. The UV polymer sat on top of the particle layer, while the particles were covered on the bottom by the PVA (B: imprints).

Another test after pick up of a monolayer with PVA and putting UV polymer on top, had areas with too much polymer on top (Figure SI VI-6 B) and areas with less polymer on top. However, the particles were still covered with polymer (Figure SI VI-6 C).

Hence, for using already prepared monolayers in the microsieves, the surface of the particles might be altered to give a low wettability of the UV polymer system. However, this experiment showed a classical infiltration problem.

Chapter VII. Summary

The research in this thesis was based on the preparation of ellipsoidal particles from spherical particles. The original monodisperse spherical particles of different size ranges could thereby be synthesized using dispersion or emulsifier-free emulsion polymerization (Chapter II.1). The particles were deformed to ellipsoids (Chapter II.2), using the uniaxial foil stretching method of Felder [51]. The basic parameters and results of this process were investigated (Chapter II.3) and were summarized in a publication [1]. It was found that the stretching process is well controllable, has a high inter-sample reproducibility and can be upscaled. Additionally, a particle size dependency of the strain on the resulting AR, internal energy storage and the recovery of surface charges of recovered particles from the foil were investigated.

The upscaling of the ellipsoidal particle manufacturing was successfully implemented by recommissioning and updating an automated foil doctor blade device (Chapter III.1) and a roll-stretching device (Chapter III.2). The automated foil doctor blade device and its updates are explained in detail. Due to issues with the foil doctor blading on the steel stripe used, the cleaning protocol for recovering the particles was updated and simplified. With the automated foil doctor blade device, particle-loaded foil ribbons of up to 8 m have been produced. These foil ribbons were then used in the roll-stretching device. For the roll-stretching device, the setup parameters and AR dependencies were investigated and explained. Some of the resulting particles exhibited different shapes (e.g. lemon or bullet) due to the short heating time in the device.

In addition to the upscaling of the process on the device parts used, the evaluation of particle images was "upscaled" as well. A semi-automated image evaluation via a python script was implemented (Chapter III.3). The resulting program in python was then used for particle sizing and positioning throughout this thesis.

Three-dimensional investigations of particles inside the foil by using the SBFI SEM method (Chapter IV) showed that the assumption of a one-dimensionally stretched particle was not true and the stretched particles exhibited three individual axes. To come to that conclusion, the evaluation of the SBFI image-stacks of polymer particles in a polymer matrix by the program AMIRA was implemented in detail within this thesis, including a necessary distortion correction of the images by a python script.

With horizontally embedded sample images from SBFI, it could additionally be shown that nearby particles hinder the shape evolution during particle stretching (Chapter IV.8). The frequently visible deformed ellipsoidal particles are, therefore, results from particles being too close during stretching.

However, with the prepared ellipsoidal particles, floating monolayers could be made (Chapter V.2) by two different self-assembly approaches on water surfaces (*RETSCH* [129] and *VOGEL* [130]). Small AR particles showed highly crystalline regions on their monolayers, while larger AR particle monolayers just depicted patches of oriented particles.

The floating monolayers could then be modified by a one-dimensional transfer (Chapter V.3) according to Hummel et al. [139]. It was shown that ellipsoidal particles could be transferred like spherical particles to differently open-packed monolayers. A present crystallinity can be retained in monolayers e.g. with small AR particles.

A versatile particle assembly approach was implemented with the rubbing method of Park et al. [151]. With this method, it was possible to access large areas of hcp mono-domain particle monolayers (spherical) and close-packed layers of ellipsoids with preferred orientations (Chapter V.4). Again, with a small AR, the ellipsoids could be assembled in crystalline lattices comparable to spherical ones. Interestingly, larger AR particles shifted their preferred orientation during rubbing from perpendicular (rolling behavior) to parallel (sliding behavior). The applied force during rubbing influenced this shift. Further, in Chapter V.4, a possible take-up procedure for rubbed particle monolayers was developed.

The rubbed crystalline monolayers of spherical and low-aspect particles were then modified by elastomer substrate stretching (Chapter V.5). Thus, the theoretic calculations of particle transfer (Chapter V.1) could be backed up with experimental data. The particle monolayers were successfully transferred to square and hexagonal open-packed lattices.

In Chapter V.6, the one-dimensional transfer monolayers of spherical particles (Chapter V.3) were used to stretch particles when they are already ordered in a crystalline lattice. The crystalline lattice positions could be successfully conserved during the recovering of the particles from the foil. The particles in the open-packed monolayers had different uniform shapes resulting from an uneven strain of the surrounding matrix due to nearby particles.

In the last chapter (Chapter VI), a possible application of the ellipsoidal particles in membranes and microsieves for water filtration was discussed. Additionally, a special embedding technique from Vogel et al. [158] was utilized to show the pH dependency of particles on their immersion depth and to create flexible microsieves.

Chapter VIII. Appendix

The appendix contains the analysis methods, the references, the tables of figure, tables, and equations, the plain python code of the ellipse detection program, and other python code snippets used for parts of the thesis.

VIII.1. Analysis Methods

In this chapter, the theory behind the used analysis methods will is explained. Every method paragraph contains a detailed description of the parameters used for measurements. These parameter paragraphs are sometimes taken from my paper [1] where indicated. Because of that, there are some few paragraphs with experimental parameters, which were not written by myself as indicated.

VIII.1.1 Scanning Electron Microscopy

Invented in 1937 by von Ardenne [176], the imaging technique of scanning electron microscopy is nowadays a basic method in scientific research. Detailed information can be found in standard books on analysis methods like *ROMEIS – Mikroskopische Technik, Chapter 1.2* [109] or *Springer Handbook of Materials Measurement Methods, Chapter B5.1.3* [177].

In this chapter, the basic principles necessary to understand the measurements in the thesis are shown.

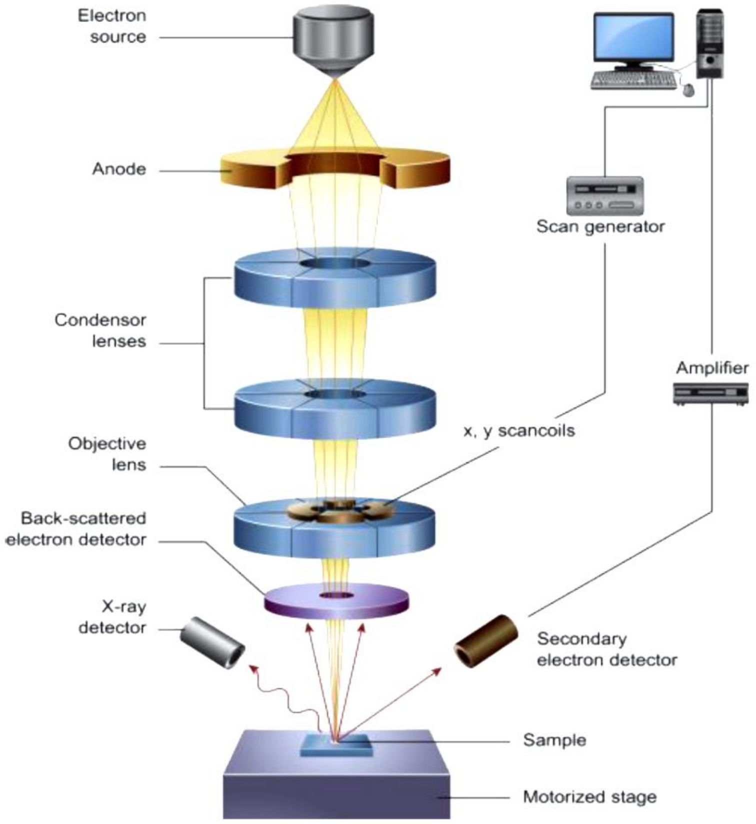


Figure Appendix VIII-1. Scheme of a scanning electron microscope. The electron beam is created (top), then focused (middle) and finally hitting the sample (bottom). Several detectors gather the scattered electrons from the sample and their signal is transferred to an image. Image from "Application of Nanomaterials in Environmental Improvement" by Ali Salman in the book "Nanotechnology and the Environment" [178] *with permission by the Creative Commons Attribution License (http://creativecommons.org/licenses/ by/3.0).*

The principle of generating a scanning electron microscopy image are depicted in Figure Appendix VIII-1. In short, a field-emitting cathode emits electrons (electron source), which are accelerated towards an anode with a pinhole in the center. The electrons propagate through the hole into the beam column. Inside the column, the electron beam is focused and deformed by electromagnetic lenses (stigmator and condenser lenses), and accelerated. The electron passes the last lens (objective lens), enters the sample chamber and hits the sample. On the sample, the electron beam interacts mainly with the surface of the sample, creating several kinds of electrons. Detectors at different positions inside the sample chamber or the column collect these electrons from the sample. The electron beam is moved over the sample area in a scanning motion, helping to counter chromatic abbreviations [176] and enlarging the field of view.

Samples have to be vacuum stable and kind of conductive to drain excess electrons. A conductive layer of gold or platinum is, therefore, often sputter coated on the sample prior investigation.

For imaging applications, the mainly used electrons created by the sample are back scattered (BSE) and secondary electrons (SE). Other rays like Auger electrons, photoluminescence or X-rays were not used in this thesis and will, therefore, not be explained here.

VIII.1.1.1 Back Scattered Electrons (BSE)

BSE are created by elastic scattering of the primary electron beam. They have, therefore, high energy levels of 200 – 50000 eV (see *Romeis – Mikroskopische Technik, Chapter 1.2.5.2.* [109]) and can be detected by backscattered electron detectors. The ZEISS electron microscopes used have three detectors for BSE. One is located directly above the sample inside the beam column above the objective lens. It is called Energy selective Backscattered detector (EsB) and gives access to material contrast. The EsB is working for low to medium beam currents. Another detector is directly below the objective lens and is called Angle selective Backscattered electron detector (AsB). Apart from angular resolution, the AsB can handle high beam current and complement the BSE detection. EsB and AsB electron detectors are very handy for imaging when the sample charges up. However, the event of backscattering is not so frequent and, therefore, the imaging resulted in poor quality or much longer scanning times (which are sometimes not possible due to charging and drifting).

The third detector for BSE, usually equipped in electron microscopes, is the *EVERHART-THORNLEY*-Detector (ETD), invented in 1960 by Everhart and Thornley [179]. The ETD sits at an angular position above the sample, has a charged cage around the detector and can collect BSE and secondary electrons. The collection of secondary electrons can be switched off by altering the collector cage voltage to negative, repelling all secondary electrons as they have a lower energy compared to BSE.

VIII.1.1.2 Secondary Electrons (SE)

SE are created by inelastic scattering of the primary electron beam. For secondary electron detection, there are the ETD, as mentioned above, and an In-Lens detector sitting right above the object lens inside the column. The In-Lens detector was used within this thesis especially with the small particle sizes for imaging. Beneficial are the location directly above the sample, giving access to the rims of the particles and less depth information. The energy of secondary electrons is about 3 – 5 eV (see *Romeis – Mikroskopische Technik, Chapter 1.2.5.2.* [109]).

VIII.1.1.3 Use of SEM in the Thesis

SEM images of sputter coated (1 – 2 nm platinum by Cressington 208HR) particles on glass slides, SI wafer pieces or PDMS (with a silver tape to ensure conductivity) were taken with two different electron microscopes from ZEISS.

In Chapter II a Leo 1530 or an Ultra Plus microscope at 3 kV acceleration voltage with an in-lens SE or ETD detector were used. The particle size for the evaluation of FN269 particles (Chapter II.3.1) was determined via image analysis by manual ellipse fitting on the depicted ellipsoids in the program Fiji/ImageJ (version 1.53c). Slightly bent particles were fitted by ellipses as well.

The normal images of further chapters were made by the Ultra Plus microscope with acceleration voltages of 2 – 3 kV and different detectors (AsB, EsB, ETD and InLense). Different detectors were chosen for different samples to minimize charging or scanning effects. However, small particles were mostly imaged with the InLense detector while big particles were mostly imaged by ETD. For few occasions, AsB and EsB were used. For the plain imaging without measurement, the ETD images were taken.

For evaluating the SEM images, in rare occasions, like e.g. measuring the immersion depth on individual particles (Chapter VI.4), the program Fiji/ImageJ (version 1.53c/e) was used. All other particle size measurements were performed with my semi-automated python program for particle analysis as described in Chapter III.3.

In Chapter IV, an Apreo volumescope SEM from FEI (Thermo Fisher Scientific Inc.) with an in-situ microtome was used for the serial block face imaging experiments (operated by Dr. Hendrik Bargel, chair of Biomaterials, University Bayreuth). The microscope had a special lens-mounted BSE detector for the low vacuum mode used (VS-DBS).

VIII.1.2 Light Scattering on Dispersion

Light scattering on particle dispersion is often used for the evaluation of the dispersed particle size distribution. The basic theories of light scattering (*FRAUNHOFER, RAYLEIGH, MIE, DEBYE*) will not be displayed in this thesis but can be read in textbooks like *Instrumentelle Analytik* [180]. However, one has to know that particles with a much smaller diameter (d) than the wavelength (λ) of incident light ($d < 0.05 \cdot \lambda$) are scattered as *RAYLEIGH*, in the size of the wavelength ($0.05 \cdot \lambda < d < \lambda$) as *DEBYE* and larger as the wavelength ($d > \lambda$) as *MIE* scattering. The particles used in this thesis were in the *DEBYE* and *MIE* scattering regime. The backwards scattering of the *MIE* scattering is smaller than with *DEBYE* while the forward scattering is similar [180].

In general, two different light scattering approaches are used for particle sizing.

VIII.1.2.1 Static Light Scattering (SLS)

The first method is the static light scattering (SLS). Here, the light shines on the sample from one side and the intensity at different angles respective to the incident light is recorded. The SLS measurement is nowadays often made with a multi angle (laser) light scattering (MALS) device and can be used in a continuous way, e.g. after FFF (VIII.1.3).

VIII.1.2.2 Dynamic Light Scattering (DLS)

The method mostly used nowadays is the dynamic light scattering (DLS). At DLS, the beam path is fixed with a given angle. In the device Nanosizer ZS (Malvern), the angles used for investigations are 13° and 173°. The size measurement is derived from an evaluation of the intensity fluctuation due to particle diffusion in and out of the scattering volume.

VIII.1.2.3 Use of DLS in the Thesis

The MALS and in-line-DLS measurements for the results in Chapter II.3.2 and in my paper of Benke et al. [1] were performed in combination with the FFF measurement by my former colleague Dr. Tanja Feller. The parameters will be explained within the chapter about the field-flow fractionation method (Chapter VIII.1.3, see next page).

VIII.1.3 Field-Flow Fractionation (FFF)

First mentioned by Giddings in 1966 [181], the field-flow fractionation (FFF) is now the umbrella term for several separation methods for dissolved or dispersed polymers, proteins or colloids using the principle of separation by flow in a channel through a special field. Each type of FFF uses a field perpendicular to the flow direction interacting with the sample to separate the sample in fractions of similar behavior. Fields, which can be applied, are e.g. magnetic, electric, temperature, gravimetric or cross-flow.

Inside the FFF, a parabolic flow profile is constructed by the carrier media. The to-be separated particles interact with the field while being pushed through the channel and accumulate on the wall of the channel. Counteracting this accumulation force is the diffusion back into the channel. As the interaction with the field is different for the several species, the stronger interacting ones are more close to the accumulation wall. With the parabolic flow profile, the strong interacting species will experience a low elution force and will just slowly move along in the channel. The weak interacting species, however, will be more distant from the channel wall, experiencing a higher elution force and will be eluted faster than the others (see *Instrumentelle Analytik: Grundlagen – Geräte – Anwendungen*, *Chapter 30.5* [180]).

VIII.1.3.1 Asymmetric Flow Field-Flow Fractionation (AF4)

The separation method for particles used here is the asymmetric flow FFF first described by Wahlund and Giddings in 1987 [182]. The field applied is a controllable cross-flow perpendicular to the channel direction. The cross-flow can be applied by having the accumulation wall exchanged to a porous frit/membrane bilayer where just the cross-flow/carrier medium can propagate through (Figure Appendix VIII-2). The device exhibits an asymmetric design as the other wall is still non-permeable. A trapezoidal channel design helps to focus the sample and to maintain a constant flow velocity along the channel [183].

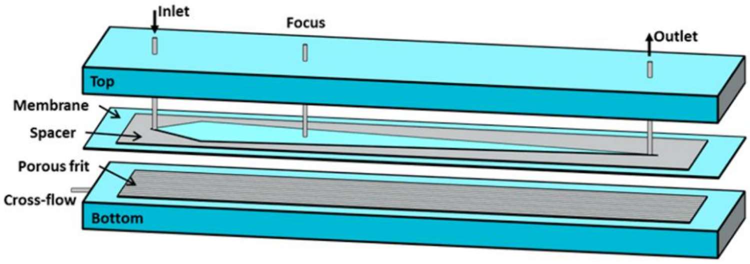


Figure Appendix VIII-2. An asymmetric channel device for AF4 separation. Figure reprinted with permission from Wagner et al. [184]. Copyright © 2014 American Chemical Society.

By tuning the cross-flow, the field strength can be adjusted easily and cross-flow force profiles can be run, making this FFF method very versatile for separating different kinds of particles. The normal countering force for the cross-flow is the diffusion of the particles back into the channel center. Small particles experience higher diffusion forces than medium-sized particles and will be eluted first (Figure Appendix VIII-3 B). However, this standard aka *BROWNIAN* mode is not true for large particles. Large particles will reach wide into the channel, although, they are accumulated to the wall. They will then experience higher carrier flow, resulting in a lifting force away from the membrane and, therefore, elute before smaller particles. This mode is called Steric mode (Figure Appendix VIII-3 C) and normally occurs for particle sizes > 1 µm [184].

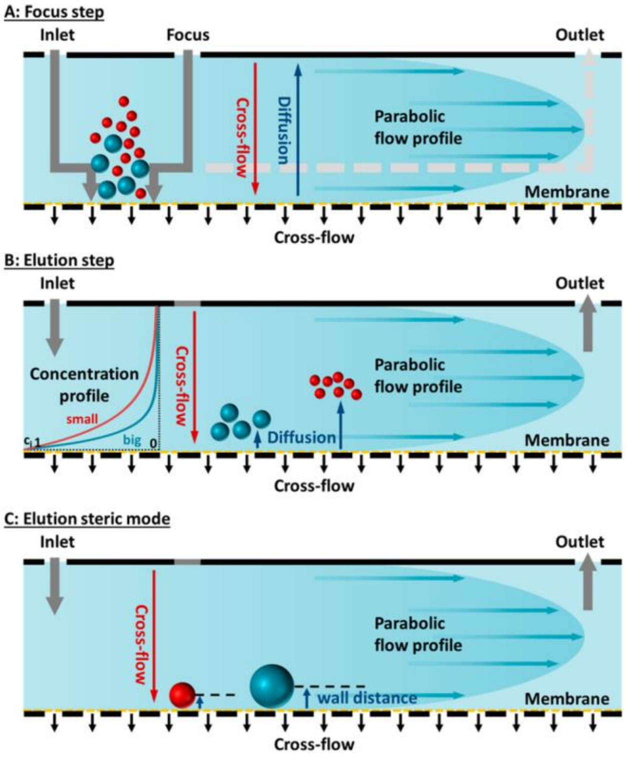


Figure Appendix VIII-3. Principle of cross flow FFF. Figure reprinted with permission from Wagner et al. [184]. Copyright © 2014 American Chemical Society.

Steps in an AF4 run are inserting and focusing (Figure Appendix VIII-3 A), elution, and evaluation by different in-line detectors. Each fraction of the separated sample can then be investigated with in-line measurement devices like UV-Vis spectrometry, MALLS, or DLS.

VIII.1.3.2 Use of FFF in the Thesis

For the results in paragraph II.3.2, the AF4 experiments were carried out by Dr. T. Feller (group of Professor M. Retsch at University Bayreuth). Therefore, the following description paragraph was written by her for the paper [1]:

"The AF4 experiments were carried out with the EAF2000 electrical flow FFF (Postnova). The channel length was 28 cm, and the spacer was 350 µm. A regenerated cellulose membrane with a 10 kDa molecular weight cutoff was used. The samples were eluted in 0.2 % v/v Novachem solution in ultrapure water. Flow rates were regulated with PN1130 isocratic pumps equipped with a vacuum degasser. All injections were performed with an autosampler. The elution was monitored at 254 nm by a PN3211 UV detector, a multi-angle light scattering (MALS) detector (PN3621) and a Malvern Zetasizer Nano Series® for continuous DLS measurements.

Generally, 20 µl of the samples with a concentration of 0.1 % w/w or 0.05 % w/w in water were injected to the channel. The detector flow rate was set to 0.5 ml/min for all measurements. The focus step lasted for 8 min (delay time and injection time) with a 1.1 mL/min flow rate. The cross-flow was set to 0.7 ml/min and decreased with an exponential decay during the elution within 45 minutes down to 0.1 ml/min. A rinsing step was included between the measurements to wash off any residue on the membrane.

For size determination, the light scattering detector was calibrated with a latex particle standard mixture (60 nm, 125 nm and 350 nm geometric diameter) and the particle scattering function was calculated by the software. The elugrams were shifted to the start of the void peak to highlight differences in elution time." [1]

VIII.1.4 Zeta Potential

A method to stabilize particles in dispersion is to have charges on their surfaces. The zeta potential gives a hint on the charge on the particle. Although, the zeta potential is sometimes treated as the charge of the particle surface, this is not the case. The zeta potential is the potential of the slipping plane next to the tightly absorbed layer (stern layer) (Figure Appendix VIII-4). The measurement incorporates the assumption of a spherical particle. By applying an electric field of different strength (voltage), the electrophoretic mobility of charged particles change. The movement increase/decrease is measured and the point of equal charge (highest mobility) is then recorded as the zeta potential.

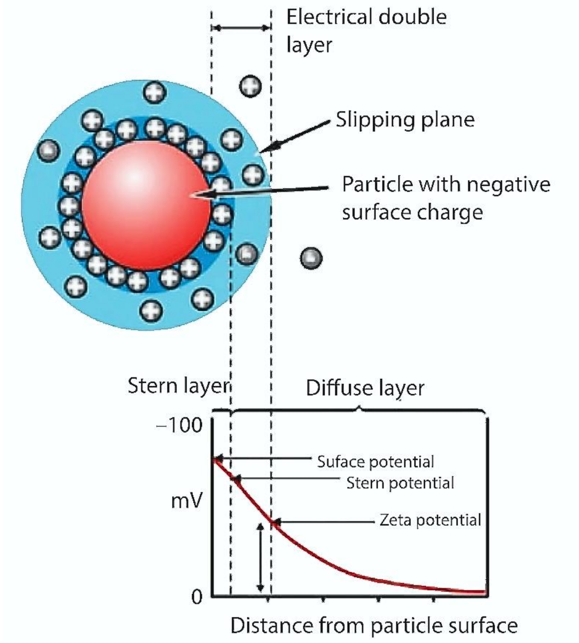


Figure Appendix VIII-4. Scheme of charges around a particle and the definition of zeta potential. Figure used with permission of Scrivener Publishing LLC, from Chapter 1.2 of the textbook Electrokinetics for Petroleum and Environmental Engineers*, Chilingar, George V., Haroun, Mohammed. © 2014; permission conveyed through Copyright Clearance Center, Inc.* [185]*.*

VIII.1.4.1 Zeta Potential Measurements in the Thesis

“Zeta potential measurements were determined on a zeta sizer Nano ZS (Malvern) within a DTS1070 folded capillary zeta cell. The particle samples were first [sonicated, then] diluted to 0.1 wt% and then 100 µl of this dispersion was mixed with 810 µl pure water and 90 µl 10 mM NaCl stock solution (same for all measurements). The final dispersion to measure contained ~0.01 wt% particles in 0.9 mM NaCl [and was sonicated again prior measurement]. The measurement temperature was set to 25 °C and 3[– 6] runs with up to 100 scans per sample were performed.” [1]

The average of the runs was rounded to integer and compared.

Although, zeta potential measurements of ellipsoidal particles were as well performed, the meaning is not the same as the equation for calculating the zeta potential is for spherical particles. However, trends should be visible and comparable.

VIII.1.5 X-ray Photoelectron Spectroscopy (XPS)

The x-ray photoelectron spectroscopy (XPS) was developed in the 1960s by Siegbahn in Sweden. Detailed explanation and spectra can be found in textbooks like e.g. in the *Handbook of X-ray Photoelectron Spectroscopy* [186]. In short, the XPS method can be used to identify the structure of molecules by the oxidation level of its atoms. Every element emits core electrons with specific energy when hit by monochromatic x-ray photons. The energy of the emitted electron depends on the shell where the electron comes from. Every element has a fingerprint as the binding energy of core electrons differs for several elements. The individual main peaks from different shells e.g. C1s (C-atom 1st s-shell) or O1s are scanned with high resolution to see fine structures. The exact structure and position of the maxima of the different element peaks depends on the chemical environment. Higher oxidation leads to higher electron binding energies and the peak shows a chemical shift. An example to show the shifting behavior of the carbon atom (C) signal is shown with ethyl trifluoroacetate in Figure Appendix VIII-5. The chemical shift for the C atom is larger for electron demanding neighbors (fluorine, peak on left), but is also very pronounced with oxygen neighbors (peak in middle). For hydrogen neighbors (2 peaks on the right), the C atom peak is shifted just slightly.

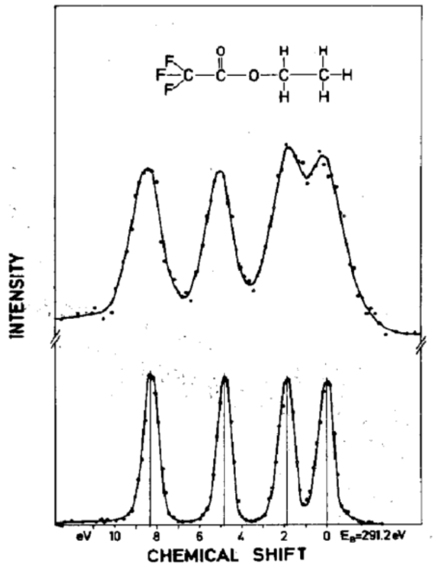


Figure Appendix VIII-5. XPS spectra of ethyl trifluoroacetate exhibiting the shifting behavior of C1s peak position with different chemical surroundings. Reprinted from Siegbahn [187], with permission from Elsevier. Copyright © 1974 Published by Elsevier B.V.

One should notice that XPS can just penetrate 1 – 5 nm into the sample, hence, giving surface properties of a sample (see *Instrumentelle Analytik: Grundlagen – Geräte – Anwendungen*, *Chapter 12* [180]). Thus, sample surfaces are normally sputter-cleaned prior measurement to have clean surfaces.

VIII.1.5.1 Use of XPS in the Thesis

The XPS spectra of PVA and PS particles differ as shown in Chapter II.3.4.1. Therefore, XPS could be used to identify residual PVA on the particles surfaces.

Measurements were performed by my colleague Dr. Sabine Rosenfeldt (chair of Physical Chemistry I, group of Professor M. Retsch, University Bayreuth) “on dry particle samples, mounted on a microscopy slide in a PHI 5000 VersaProbe III XPS system without sputtering prior measurement.” [1] Sputtering was not done on purpose, as I was interested in the molecules on the surface of the particles.

VIII.1.6 Simultaneous Thermal Analysis (STA)

A so called simultaneous thermal analysis is a direct combination of a thermal gravimetric analysis (for detailed information see *Springer Handbook of Materials Measurement Methods, Chapter C8.4* [177]) coupled with an infrared spectrometer (see *Springer Handbook of Materials Measurement Methods, Chapter B4.1* [177]). The advantage of the combination lies in identifying the species of molecule by the IR spectra at the temperature when a weight loss of the sample occurs (desorbing molecules).

VIII.1.6.1 Thermogravimetric Analysis (TGA)

A sample is heated in a controlled atmosphere while standing on a sensitive balance. The weight change of the sample is recorded as a function of the temperature. With the TGA measurement temperature ranges for physical changes like drying, sublimation, or evaporation, but also for chemical changes like degradation or oxidation of a sample can be identified by mass change [180]. The heating range of a TGA is normally from room temperature up to 1000 – 1600 °C [180]. For polymer samples, which degrade normally at around 200 °C, heating above 500 °C is normally not done. The heating speed can be adjusted and temperature programs including different chamber gases are used, e.g. first inert gases like argon or nitrogen and later oxygen to differ between degradation and oxidation processes.

VIII.1.6.2 Infrared Spectroscopy (IR)

Light in the infrared wavelength region (750 nm – 300 µm) shines on a sample and gets partially absorbed by the molecules when the light frequency fits their vibrational modes. The transmitted/reflected light is recorded by wavelength (or wavenumber, the reciprocal wavelength) with a spectrometer. Each substance gives an individual IR spectrum dependent on the possible vibration of its molecular structure, thus can be identified (e.g., by using spectra libraries). Substances from single atoms (e.g. gases like helium He) or linear molecules of two same atoms (like nitrogen N_2) do not show IR absorbance. The "infrared spectroscopy is applicable to gases, liquids and solid samples" (see *Springer Handbook of Materials Measurement Methods, Chapter B4.1* [177]).

VIII.1.6.3 Use of STA in the Thesis

The STA analysis was carried out by my colleague Dr. Tanja Feller (group of Professor M. Retsch at University Bayreuth) and the following paragraph was written by her:

"An STA 449 F3 (Netzsch) coupled with an Alpha II FT-IR spectrometer (Bruker) was used for thermal characterization. The measurements were performed in Pt/Rh crucibles from 30 – 800 °C with a heating rate of 10 K/min in nitrogen atmosphere."

VIII.1.7 Differential Scanning Calorimetry (DSC)

In the standard differential scanning calorimetry method (DSC), a sample and a reference are heated inside an oven with a given heating speed. The heat flux, difference from sample to reference, is recorded. Positive heat flux stems from exothermal processes like crystallization or chain relaxation, while a negative heat flux is from endothermal processes like melting. Within the DSC thermogram of polymers, another endothermal step is mostly visible, resembling the glass transition temperature (T_g, see Chapter VIII.1.7.2).

For the thermal evaluation, the peaks in the heat flow curves are extrapolated to get the onset temperature for the transition (e.g. T_g by onset, according to the industry norm ASTM 3418). The transition enthalpy can be found as the value of the area below the peak. Further details on the method and its evaluation can be found in textbooks like *Springer Handbook of Materials Measurement Methods (Chapter C8.2.2* [177]).

VIII.1.7.1 Modulated DSC (mDSC)

The differential scanning calorimetry measurement was enhanced by using a temperature modulation in 1993 [188]. The constant heating curve of a normal DSC measurement is altered with a sinusoidal heat modulation. This resulted in micro heating- and cooling-cycles during a constant heating or cooling process (see *Instrumentelle Analytik: Grundlagen – Geräte –*

Anwendungen, *Chapter 31.3* [180]). With a *FOURIER* transformation of the signal, it is then possible to split the total heat flux into two contributing parts: the non-reversible heat flux and the reversible heat-flux. The splitting in the different heat fluxes can help to identify different endothermal and exothermal processes especially in sample mixtures like polymer blends.

VIII.1.7.2 Glass Transition Temperature

The glass transition temperature (T_g) is the temperature where polymer chains start to move while staying in the same place (bending, twisting, rotating, etc.). Apart from the onset temperature (T_{g_onset}), the T_g is defined differently. For example in a paper of Rieger and, according to him, in the industry norm DIN 53765, T_g was defined by the temperature of half step-height (midpoint) of the glass transition, thus shifting the temperature to higher values (e.g. PS: T_{g_onset} = 98 ± 1 °C and $T_{g\_midpoint}$ = 107 ± 2 °C) [86]. The T_g for polystyrene is also subject of several other publications and is not strictly to be defined as it depends on the sample (molecular weight, mass, thermal history). Literature values range from 94 – 114 °C (overview see Marti et al. [189]). For a better discussion and as it fits the data of Chapter II.3.3, the $T_{g\_midpoint}$ value from Rieger (107 ± 2 °C) is used as the glass transition temperature for polystyrene within this thesis.

VIII.1.7.3 Use of mDSC in the Thesis

Measurements of spherical and ellipsoidal particles with the modulated DSC (discussed in Chapter II.3.3) were conducted by Dr. Anna Neuhöfer (group of professor M. Retsch at University Bayreuth) in aluminium "pans on a Discovery DSC 2500 (TA Instruments) to determine the reversing and non-reversing thermal properties of the samples. First, the sample was equilibrated at 50 °C. The temperature modulation to perform the mDSC run was set to 0.2 K for 60 s. The sample is then heated with 1 K/min to 150 °C (1st heating). After cooling down at 20 K/min to 50 °C and stabilizing the temperature, the 2nd heating with 1 K/min to 150 °C was performed without moving the sample." [1]

The comparison of the 1st and the 2nd heating cycle, especially in the non-reverse thermogram, gave the possibility to see differences resulting from the structure of the sample/polymer particles.

VIII.1.8 Pictures of Deposited Monolayers and Particles

The images of deposited particles were made not just with SEM but also with several other devices. Especially handy was the confocal laser-scanning microscope (LSM), which was used not just with the 3D imaging possibility but as well for 2D images. Further, photographs of the samples were taken with several devices and with special illumination.

VIII.1.8.1 LS Microscope

A confocal optical laser-scanning microscope (LEXT from Olympus) was used for imaging particles on glass slides or PDMS rubbing substrates (unstretched or stretched). The LSM microscope was set up inside a black box to shield it from light. The whole box was additionally damped to counter vibrations. The objective mostly used was 100x with an additional digital zoom of 1 – 6x. 2D-images in laser (LSM2D) or white-light modus were taken. The 3D modus gave height profiles and was used to check e.g. the thickness of spin coated PVA on glass slides.

VIII.1.8.2 Normal Photographs

Photographs of monolayers were performed with a camera (Canon EOS) or a mobile phone (Redmi Note 9 Pro). The colorful diffractive images of monolayers were made with daylight lamp TL40 (Breuer GmbH) from behind with various angles.

VIII.1.9 Laser Diffraction – Setup and Equations

The laser diffraction setup was home build. A laser pointer (Thorlabs, PL202, λ = 635 nm), a sample holder, a screen (white printing paper 80 g/qm) and a camera (Canon EOS) were fixed on an optical plate.

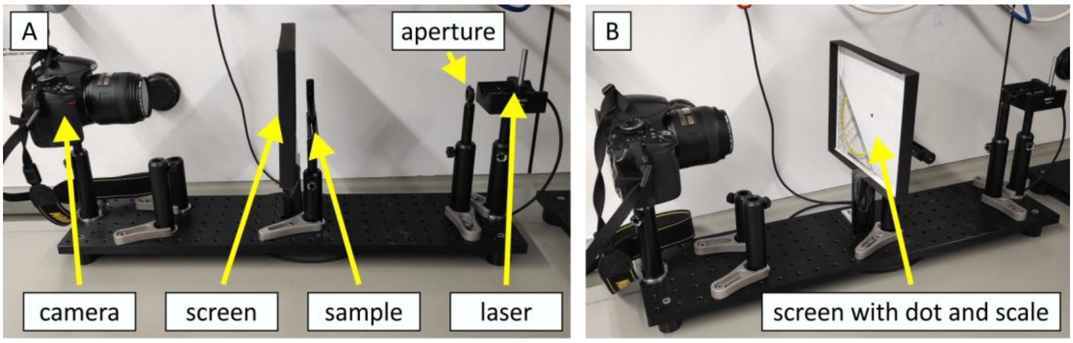


Figure Appendix VIII-6. Home-built setup for laser diffraction. The screen was a white printing paper with a black dot (blocking primary beam). A triangular ruler was used as a scale on the screen.

The samples were mounted on a movable sample holder in between the laser pointer and the screen. The distance of the samples to the screen was 25 ± 1 mm. The camera position was 300 mm behind the screen. A triangular ruler was positioned on one edge of the screen to have a scale bar for calculating lattice distances.

The equations to calculate lattice distances (d) were derived from *BRAGG*s law (see text books like *Introduction to solid state physics* [112] or *Springer Handbook of Materials Measurement Methods* [177]).

$$n \cdot \lambda = 2d \cdot \sin\theta \qquad \text{(Equation VIII.1.9-1)}$$

The laser diffraction maxima visible did normally not exceed the first maximum (n = 1). Further, at diffraction, the path difference of the wavelength for constructive interference needs to be counted just once, not twice as in *BRAGG* (90° for entrance and θ for exit), so the equation can be adapted to

$$\lambda = d \cdot \sin\theta_{dif} \qquad \text{(Equation VIII.1.9-2)}$$

The angle of the diffraction can then be calculated using basic math on the dimensions of sample distance to the screen (b) and distance of diffraction maxima to center (a).

$$\sin\theta_{dif} = \frac{a}{\sqrt{b^2+a^2}} \qquad \text{(Equation VIII.1.9-3)}$$

Inserting Equation VIII.1.9-3 in Equation VIII.1.9-1 results in:

$$d = \frac{\lambda \cdot \sqrt{b^2+a^2}}{a} \qquad \text{(Equation VIII.1.9-4)}$$

VIII.2. References of the Thesis

1. Benke, D.; Feller, T.; Krüsmann, M.; Neuhöfer, A. M.; Ganster, F.; Karg, M.; Retsch, M., Prolate Spheroidal Polystyrene Nanoparticles: Matrix-Assisted Synthesis, Interface Properties, and Scattering Analysis. *Soft Matter* **2023,** *19*, 9006-9016.
2. Schwaar, N.; Benke, D.; Retsch, M.; Goedel, W. A., Float-Cast Microsieves with Elliptical Pores. *Langmuir* **2024**, 22516-22525.
3. Feynman, R. P., There's Plenty of Room at the Bottom. Caltech Engineering and Science: **1960**; pp 22-36.
4. Ziegler, C.; Wolf, A.; Liu, W.; Herrmann, A.-K.; Gaponik, N.; Eychmuller, A., Modern Inorganic Aerogels. *Angew. Chem. Int. Ed.* **2017,** *56*, 13200-13221.
5. Sanders, J. V., Close-packed structures of spheres of two different sizes I. Observations on natural opal. *Philos. Mag. A* **2006,** *42*, 705-720.
6. Yin, Z.; Hannard, F.; Barthelat, F., Impact-resistant nacre-like transparent materials. *Science* **2019,** *364*, 1260-1263.
7. Wang, D.; Han, S.; Yang, M., Tooth Diversity Underpins Future Biomimetic Replications. *Biomimetics* **2023,** *8*, 42.
8. Boden, S. A.; Bagnall, D. M., Moth-Eye Antireflective Structures. In *Encyclopedia of Nanotechnology*, Bhushan, B., Ed. Springer Netherlands: Dordrecht, **2012**; pp 1467-1477.
9. Colvin, V. L., From Opals to Optics: Colloidal Photonic Crystals. *MRS Bulletin* **2011,** *26*, 637-641.
10. Takei, H.; Shimizu, N., Gradient Sensitive Microscopic Probes Prepared by Gold Evaporation and Chemisorption on Latex Spheres. *Langmuir* **1997,** *13*, 1865-1868.
11. Bishop, K. J. M.; Biswal, S. L.; Bharti, B., Active Colloids as Models, Materials, and Machines. *Annual Review of Chemical and Biomolecular Engineering* **2023,** *14*, 1-30.
12. Bharti, B.; Velev, O. D., Assembly of Reconfigurable Colloidal Structures by Multidirectional Field-Induced Interactions. *Langmuir* **2015,** *31*, 7897-7908.
13. Steinhaus, A.; Srivastva, D.; Nikoubashman, A.; Groschel, A. H., Janus Nanostructures from ABC/B Triblock Terpolymer Blends. *Polymers* **2019,** *11*, 1107.
14. Lee, J. G.; Al Harraq, A.; Bishop, K. J. M.; Bharti, B., Fabrication and Electric Field-Driven Active Propulsion of Patchy Microellipsoids. *J. Phys. Chem. B* **2021,** *125*, 4232-4240.
15. Ozaki, M.; Kratohvil, S.; Matijevic, E., Formation of Monodispersed Spindle-Type Hematite Particles. *Journal of Colloid and Interface Science* **1984,** *102*, 146-151.
16. Hu, X.; Yu, J. C., Continuous Aspect-Ratio Tuning and Fine Shape Control of Monodisperseα-Fe2O3 Nanocrystals by a Programmed Microwave–Hydrothermal Method. *Advanced Functional Materials* **2008,** *18*, 880-887.
17. Nasilowski, M.; Mahler, B.; Lhuillier, E.; Ithurria, S.; Dubertret, B., Two-Dimensional Colloidal Nanocrystals. *Chem. Rev.* **2016,** *116*, 10934-10982.
18. Erdem, O.; Gungor, K.; Guzelturk, B.; Tanriover, I.; Sak, M.; Olutas, M.; Dede, D.; Kelestemur, Y.; Demir, H. V., Orientation-Controlled Nonradiative Energy Transfer to Colloidal Nanoplatelets: Engineering Dipole Orientation Factor. *Nano Lett.* **2019,** *19*, 4297-4305.
19. Lu, Y.; Yin, Y.; Xia, Y., Three-Dimensional Photonic Crystals with Non-spherical Colloids as Building Blocks. *Advanced Materials* **2001,** *13*, 415-420.
20. Kuijk, A.; van Blaaderen, A.; Imhof, A., Synthesis of monodisperse, rodlike silica colloids with tunable aspect ratio. *Journal of the American Chemical Society* **2011,** *133*, 2346-2349.
21. M, M. R.; Pujala, R. K.; Dhara, S., Colloidal analogues of polymer chains, ribbons and 2D crystals employing orientations and interactions of nano-rods dispersed in a nematic liquid crystal. *Scientific Reports* **2019,** *9*, 4652.
22. Jacucci, G.; Longbottom, B. W.; Parkins, C. C.; Bon, S. A. F.; Vignolini, S., Anisotropic silica colloids for light scattering. *Journal of Materials Chemistry C* **2021,** *9*, 2695-2700.

23. Nikoobakht, B.; El-Sayed, M. A., Preparation and Growth Mechanism of Gold Nanorods (NRs) Using Seed-Mediated Growth Method. *Chemistry of Materials* **2003,** *15*, 1957-1962.

24. Chhatre, A.; Thaokar, R.; Mehra, A., Formation of Gold Nanorods by Seeded Growth: Mechanisms and Modeling. *Crystal Growth & Design* **2018,** *18*, 3269-3282.

25. Feller, D.; Otten, M.; Hildebrandt, M.; Krusmann, M.; Bryant, G.; Karg, M., Translational and rotational diffusion coefficients of gold nanorods functionalized with a high molecular weight, thermoresponsive ligand: a depolarized dynamic light scattering study. *Soft Matter* **2021,** *17*, 4019-4026.

26. Boles, M. A.; Engel, M.; Talapin, D. V., Self-Assembly of Colloidal Nanocrystals: From Intricate Structures to Functional Materials. *Chem. Rev.* **2016,** *116*, 11220-11289.

27. Su, H.; Hurd Price, C.-A.; Jing, L.; Tian, Q.; Liu, J.; Qian, K., Janus particles: design, preparation, and biomedical applications. *Materials Today Bio* **2019,** *4*, 100033.

28. Zhao, B.; Li, D.; Long, Y.; Song, K., Precisely Endowing Colloidal Particles with Silica Branches. *Scientific Reports* **2019,** *9*, 8591.

29. Johnson, P. M.; van Kats, C. M.; van Blaaderen, A., Synthesis of colloidal silica dumbbells. *Langmuir* **2005,** *21*, 11510-11517.

30. Xu, S.; Nie, Z.; Seo, M.; Lewis, P.; Kumacheva, E.; Stone, H. A.; Garstecki, P.; Weibel, D. B.; Gitlin, I.; Whitesides, G. M., Generation of Monodisperse Particles by Using Microfluidics: Control over Size, Shape, and Composition. *Angewandte Chemie* **2005,** *117*, 734-738.

31. Ohm, C.; Kapernaum, N.; Nonnenmacher, D.; Giesselmann, F.; Serra, C.; Zentel, R., Microfluidic synthesis of highly shape-anisotropic particles from liquid crystalline elastomers with defined director field configurations. *Journal of the American Chemical Society* **2011,** *133*, 5305-5311.

32. Alargova, R. G.; Paunov, V. N.; Velev, O. D., Formation of polymer microrods in shear flow by emulsification - solvent attrition mechanism. *Langmuir* **2006,** *22*, 765-774.

33. Song, S.-H.; Kim, K.; Choi, S.-E.; Han, S.; Lee, H.-S.; Kwon, S.; Park, W., Fine-tuned grayscale optofluidic maskless lithography for three-dimensional freeform shape microstructure fabrication. *Opt. Lett.* **2014,** *39*, 5162-5165.

34. Li, J.; Chen, L.; Xu, J.; Wang, K.; Wang, X.; He, X.; Dong, H.; Lin, S.; Zhu, J., Photoguided Shape Deformation of Azobenzene-Containing Polymer Microparticles. *Langmuir* **2015,** *31*, 13094-13100.

35. Nam, S. M.; Kim, K.; Kang, I.-S.; Park, W.; Lee, W., Generation of 3D Microparticles in Microchannels with Non-rectangular Cross Sections. *BioChip Journal* **2019,** *13*, 226-235.

36. Zhang, P.; Xia, J.; Luo, S., Generation of Well-Defined Micro/Nanoparticles via Advanced Manufacturing Techniques for Therapeutic Delivery. *Materials* **2018,** *11*, 623.

37. Sacanna, S.; Rossi, L.; Wouterse, A.; Philipse, A. P., Observation of a shape-dependent density maximum in random packings and glasses of colloidal silica ellipsoids. *Journal of Physics: Condensed Matter* **2007,** *19*, 376108.

38. Liu, G.; Yang, X.; Wang, Y., Synthesis of ellipsoidal hematite/silica/polymer hybrid materials and the corresponding hollow polymer ellipsoids. *Langmuir* **2008,** *24*, 5485-5491.

39. Martchenko, I.; Dietsch, H.; Moitzi, C.; Schurtenberger, P., Hydrodynamic properties of magnetic nanoparticles with tunable shape anisotropy: prediction and experimental verification. *J. Phys. Chem. B* **2011,** *115*, 14838-14845.

40. Pal, A.; De Filippo, C. A.; Ito, T.; Kamal, M. A.; Petukhov, A. V.; De Michele, C.; Schurtenberger, P., Shape Matters in Magnetic-Field-Assisted Assembly of Prolate Colloids. *ACS Nano* **2022,** *16*, 2558-2568.

41. Lu, Y.; Yin, Y.; Xia, Y., Preparation and Characterization of Micrometer-Sized "Egg Shells". *Advanced Materials* **2001,** *13*, 271-274.

42. Peltonen, L.; Hirvonen, J., Pharmaceutical nanocrystals by nanomilling: critical process parameters, particle fracturing and stabilization methods. *J. Pharm. Pharmacol.* **2010,** *62*, 1569-1579.

43. van Kats, C. M.; Johnson, P. M.; van den Meerakker, J. E. A. M.; van Blaaderen, A., Synthesis of Monodisperse High-Aspect-Ratio Colloidal Silicon and Silica Rods. *Langmuir* **2004,** *20*, 11201-11207.
44. Champion, J. A.; Katare, Y. K.; Mitragotri, S., Making polymeric micro- and nanoparticles of complex shapes. *PNAS* **2007,** *104*, 11901-11904.
45. Hu, Y.; Ge, J.; Zhang, T.; Yin, Y., A Blown Film Process to Disk-Shaped Polymer Ellipsoids. *Advanced Materials* **2008,** *20*, 4599-4602.
46. Liu, B.; Wang, D., High-throughput transformation of colloidal polymer spheres to discs simply via magnetic stirring of their dispersions. *Langmuir* **2012,** *28*, 6436-6440.
47. Shin, H.; Kim, C., Preparation of spheroidal and ellipsoidal particles from spherical polymer particles by extension of polymer film. *Colloid and Polymer Science* **2012,** *290*, 1309-1315.
48. Ahn, S. J.; Ahn, K. H.; Lee, S. J., Film squeezing process for generating oblate spheroidal particles with high yield and uniform sizes. *Colloid and Polymer Science* **2016,** *294*, 859-867.
49. Voggenreiter, M.; Roller, J.; Geiger, J.; Ebner, L.; Zumbusch, A.; Meijer, J.-M., Preparation and Tracking of Oblate Core-Shell Polymethyl-Methacrylate Ellipsoids. *Langmuir* **2020,** *36*, 13087-13095.
50. Lo, Y.-C.; Tseng, H.-F.; Chiu, Y.-J.; Wu, B.-H.; Li, J.-W.; Chen, J.-T., Solvent-Induced Shape Recovery of Anisotropic Polymer Particles Prepared by a Modified Thermal Stretching Method. *Langmuir* **2018,** *34*, 8326-8332.
51. Felder, B., 49. Über die Teilchengrössenabhängigkeit der Lichtabsorption in heterogenen Systemen. II. Experimentelle Untersuchungen an Modell-Teilchen. *Helvetica Chimica Acta* **1966,** *49*, 440-453.
52. Keville, K. M.; Franses, E. I.; Caruthers, J. M., Preparation and Characterization of Monodisperse Polymer Microspheroids. *Journal of Colloid and Interface Science* **1991,** *144*, 103-126.
53. Ho, C. C.; Keller, A.; Odell, J. A.; Ottewill, R. H., Monodisperse Ellipsoidal Polystyrene Latex Particles: Preparation and Characterisation. *Polymer International* **1993,** *30*, 207-211.
54. Felder, B., 137. Einfluss der Partikelgeometrie auf die optischen Eigenschaften stark absorbierender Farbstoffteilchen. *Helvetica Chimica Acta* **1968,** *51*, 1224-1234.
55. Madivala, B.; Vandebril, S.; Fransaer, J.; Vermant, J., Exploiting particle shape in solid stabilized emulsions. *Soft Matter* **2009,** *5*, 1717-1727.
56. Mohraz, A.; Solomon, M. J., Direct visualization of colloidal rod assembly by confocal microscopy. *Langmuir* **2005,** *21*, 5298-5306.
57. Roller, J.; Geiger, J. D.; Voggenreiter, M.; Meijer, J.-M.; Zumbusch, A., Formation of nematic order in 3D systems of hard colloidal ellipsoids. *Soft Matter* **2020,** *16*, 1021-1028.
58. Zhang, Z.; Pfleiderer, P.; Schofield, A. B.; Clasen, C.; Vermant, J., Synthesis and directed self-assembly of patterned anisometric polymeric particles. *Journal of the American Chemical Society* **2011,** *133*, 392-395.
59. Roller, J.; Pfleiderer, P.; Meijer, J.-M.; Zumbusch, A., Detection and tracking of anisotropic core-shell colloids. *Journal of Physics: Condensed Matter* **2018,** *30*, 395903.
60. Florez, L.; Herrmann, C.; Cramer, J. M.; Hauser, C. P.; Koynov, K.; Landfester, K.; Crespy, D.; Mailander, V., How Shape Influences Uptake: Interactions of Anisotropic Polymer Nanoparticles and Human Mesenchymal Stem Cells. *Small* **2012,** *8*, 2222-2230.
61. Brown, T. D.; Habibi, N.; Wu, D.; Lahann, J.; Mitragotri, S., Effect of Nanoparticle Composition, Size, Shape, and Stiffness on Penetration Across the Blood–Brain Barrier. *ACS Biomaterials Science & Engineering* **2020,** *6*, 4916-4928.
62. Fan, J.-B.; Song, Y.; Li, H.; Jia, J.-P.; Guo, X.; Jiang, L., Controllable drug release and effective intracellular accumulation highlighted by anisotropic biodegradable PLGE nanoparticles. *Journal of Materials Chemistry B* **2014,** *2*, 3911-3914.

63. Haryadi, B. M.; Hafner, D.; Amin, I.; Schubel, R.; Jordan, R.; Winter, G.; Engert, J., Nonspherical Nanoparticle Shape Stability Is Affected by Complex Manufacturing Aspects: Its Implications for Drug Delivery and Targeting. *Advanced Healthcare Materials* **2019,** *8*, 1900352.
64. Doshi, N.; Mitragotri, S., Needle-shaped polymeric particles induce transient disruption of cell membranes. *J. R. Soc. Interface* **2010,** *7*, S403-S410.
65. Zheng, Z.; Han, Y., Self-diffusion in two-dimensional hard ellipsoid suspensions. *J. Chem. Phys.* **2010,** *133*, 124509.
66. Kirvin, A.; Gregory, D.; Parnell, A.; Campbell, A. I.; Ebbens, S., Rotating ellipsoidal catalytic micro-swimmers via glancing angle evaporation. *Materials Advances* **2021,** *2*, 7045-7053.
67. Mathaes, R.; Manning, M. C.; Winter, G.; Engert, J.; Wilson, G. A., Shape Characterization of Subvisible Particles Using Dynamic Imaging Analysis. *J. Pharm. Sci.* **2020,** *109*, 375-379.
68. Rashidi, A.; Dominguez-Medina, S.; Yan, J.; Efremenko, D. S.; Vasilyeva, A. A.; Doicu, A.; Wriedt, T.; Wirth, C. L., Developing Scattering Morphology Resolved Total Internal Reflection Microscopy (SMR-TIRM) for Orientation Detection of Colloidal Ellipsoids. *Langmuir* **2020,** *36*, 13041-13050.
69. Song, G.; Steelman, Z. A.; Kendall, W.; Park, H. S.; Wax, A., Spatial scanning of a sample with two-dimensional angle-resolved low-coherence interferometry for analysis of anisotropic scatterers. *Biomedical Optics Express* **2020,** *11*, 4419-4430.
70. Loudet, J. C.; Yodh, A. G.; Pouligny, B., Wetting and Contact Lines of Micrometer-Sized Ellipsoids. *Phys. Rev. Lett.* **2006,** *97*, 018304.
71. Honda, K.; Sazuka, Y.; Iizuka, K.; Matsui, S.; Uchihashi, T.; Kureha, T.; Shibayama, M.; Watanabe, T.; Suzuki, D., Hydrogel Microellipsoids that Form Robust String-Like Assemblies at the Air/Water Interface. *Angew. Chem. Int. Ed.* **2019,** *58*, 7294-7298.
72. van Herk, A. M., Emulsion (Homo)polymerization. In *Encyclopedia of Polymeric Nanomaterials*, Kobayashi, S.; Müllen, K., Eds. Springer-Verlag: **2015**; pp 688-695.
73. Goodwin, J. W.; Hearn, J.; Ho, C. C.; Ottewill, R. H., Studies on the preparation and characterisation of monodisperse polystyrene laticee. *Colloid and Polymer Science* **1974,** *252*, 464-471.
74. Hansen, F. K.; Ugelstad, J., Particle Nucleation in Emulsion Polymerization. I. A Theory for Homogeneous Nucleation. *Journal of Polymer Science: Polymer Chemistry Edition* **1978,** *16*, 1953-1979.
75. Nutz, F. A., Thermal Transport in Polymer Colloidal Assemblies, Dissertation, Universität Bayreuth, Fakultät für Biologie, Chemie und Geowissenschaften, **2018**.
76. Juang, M. S.-D.; Krieger, I. M., Emulsifier-free emulsion polymerization with Ionic Comonomer. *Journal of Polymer Science: Polymer Chemistry Edition* **1976,** *14*, 2089-2107.
77. Chern, C. S., Emulsion polymerization mechanisms and kinetics. *Prog. Polym. Sci.* **2006,** *31*, 443-486.
78. Arshady, R., Suspension, emulsion, and dispersion polymerization: A methodological survey. *Colloid & Polymer Science* **1992,** *270*, 717-732.
79. Canelas, D., Dispersion Polymerization. In *Encyclopedia of Polymeric Nanomaterials*, Kobayashi, S.; Müllen, K., Eds. Springer Berlin Heidelberg: Berlin, Heidelberg, **2015**; pp 576-582.
80. Barrett, K. E. J., Dispersion Polymerisation in Organic Media. *British Polymer Journal* **1973,** *5*, 259-271.
81. Almog, Y.; Reich, S.; Levy, M., Monodisperse Polymeric Spheres in the Micron Size Range by a Single Step Process. *British Polymer Journal* **1982,** *14*, 131-136.
82. Nutz, F. A.; Philipp, A.; Kopera, B. A. F.; Dulle, M.; Retsch, M., Low Thermal Conductivity through Dense Particle Packings with Optimum Disorder. *Advanced Materials* **2018,** *30*, 1704910.

83. Lechner, A. M.; Feller, T.; Song, Q.; Kopera, B. A. F.; Heindl, L.; Drechsler, M.; Rosenfeldt, S.; Retsch, M., Scalable synthesis of smooth PS@TiO2 core-shell and TiO2 hollow spheres in the (sub) micron size range: understanding synthesis and calcination parameters. *Colloid and Polymer Science* **2020,** *298*, 867-878.

84. Horak, D.; Svec, F.; Fréchet, J. M. J., Preparation and Control of Surface Properties of Monodisperse Micrometer Size Beads by Dispersion Copolymerization of Styrene and Butyl Methacrylate in Polar Media. *Journal of Polymer Science: Polymer Chemistry Edition* **1995,** *33*, 2329-2338.

85. Ho, C. C.; Keller, A.; Odell, J. A.; Ottewill, R. H., Preparation of monodisperse ellipsoidal polystyrene particles. *Colloid and Polymer Science* **1993,** *271*, 469-479.

86. Rieger, J., The Glass Transition Temperature of Polystyrene. *J. Therm. Anal.* **1996,** *46*, 965-972.

87. Patil, A.; Dyawanapelly, S.; Dandekar, P.; Jain, R., Fabrication and Characterization of Non-spherical Polymeric Particles. *Journal of Pharmaceutical Innovation* **2021,** *16*, 747–758.

88. Crassous, J. J.; Dietsch, H.; Pfleiderer, P.; Malik, V.; Diaz, A.; Hirshi, L. A.; Drechsler, M.; Schurtenberger, P., Preparation and characterization of ellipsoidal-shaped thermosensitive microgel colloids with tailored aspect ratios. *Soft Matter* **2012,** *8*, 3538-3548.

89. Meyer, R. A.; Meyer, R. S.; Green, J. J., An automated multidimensional thin film stretching device for the generation of anisotropic polymeric micro- and nanoparticles. *Journal Biomedical Materials Research A* **2015,** *103*, 2747-2757.

90. Mickiewicz, D.; Basinska, T.; Gosecka, M.; Gadzinowski, M.; Slomkowski, S., Colloidal liquid crystal type assemblies of spheroidal polystyrene core/polyglycidol-rich shell particles (P[S/PGL]) formed at the liquid-silicon-air interface by a directed dewetting process. *Polymers for Advanced Technologies* **2019,** *30*, 1724-1731.

91. Ho, C. C.; Hill, M. J.; Odell, J. A., Morphology of ellipsoidal latex particles. *Polymer* **1993,** *34*, 2019-2023.

92. Rey, M.; Yu, T.; Guenther, R.; Bley, K.; Vogel, N., A Dirty Story: Improving Colloidal Monolayer Formation by Understanding the Effect of Impurities at the Air/Water Interface. *Langmuir* **2019,** *35*, 95-103.

93. Louette, P.; Bodino, F.; Pireaux, J.-J., Poly(vinyl alcohol) (PVA) XPS Reference Core Level and Energy Loss Spectra. *Surface Science Spectra* **2005,** *12*, 106-110.

94. Girardeaux, C.; Pireaux, J.-J., Analysis of Polystyrene (PS) by XPS. *Surface Science Spectra* **1996,** *4*, 130-133.

95. Ferrar, J. A.; Pavlovsky, L.; Viges, E.; Liu, Y.; Solomon, M. J., Two-step continuous production of monodisperse colloidal ellipsoids at rates of one gram per day. *AIChE Journal* **2018,** *64*, 697-707.

96. Clariant, *Mowiol - Polyvinyl Alcohol - brochure of the manufacturer*. Clariant GmbH Division CP BU Polyvinyl Alcohol / Polyvinyl Butyral Marketing: Am Unisys-Park 1, D - 65843 Sulzbach, **1999**.

97. Siglreitmeier, M. K., Multifunctional Layered Magnetic Composites, Universität Konstanz, **2015**.

98. Suppiah, D. D.; Johan, M. R., Influence of solution pH on the formation of iron oxide nanoparticles. *Materials Research Express* **2019,** *6*, 015008.

99. Bloo, J.; Romauer, E.; Hollerwöger, A. United States Patent: Foil Stretching Plant. 5,184,379, 09.02.1993, **1993**.

100. van der Walt, S.; Schonberger, J. L.; Nunez-Iglesias, J.; Boulogne, F.; Warner, J. D.; Yager, N.; Gouillart, E.; Yu, T.; contributors, s.-i., scikit-image: image processing in Python. *PeerJ* **2014,** *2*, 453.

101. Umesh, P., Image Processing in Python. *CSI Communications* **2012,** *23*.

102. Oliphant, T. E., *A guide to NumPy*. Trelgol Publishing USA: **2006**; Vol. 1.

103. Hunter, J. D., Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering* **2007,** *9*, 90-95.
104. Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; van der Walt, S. J.; Brett, M.; Wilson, J.; Millman, K. J.; Mayorov, N.; Nelson, A. R. J.; Jones, E.; Kern, R.; Larson, E.; Carey, C. J.; Polat, I.; Feng, Y.; Moore, E. W.; VanderPlas, J.; Laxalde, D.; Perktold, J.; Cimrman, R.; Henriksen, I.; Quintero, E. A.; Harris, C. R.; Archibald, A. M.; Ribeiro, A. H.; Pedregosa, F.; van Mulbregt, P., SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **2020,** *17*, 261-272.
105. Zheng, Z.; Han, Y., Glass transitions in monolayers of colloidal ellipsoids. *AIP Conference Proceedings* **2013,** *1518*, 153-161.
106. ThermoScientific, VolumeScope System Datasheet. **2017,** *DS0222-EN-10-2017*.
107. ThermoScientific, VolumeScope 2 System Datasheet. **2019,** *DS0304-EN-10-2019*.
108. de Goede, M.; Johlin, E.; Sciacca, B.; Boughorbel, F.; Garnett, E. C., 3D multi-energy deconvolution electron microscopy. *Nanoscale* **2017,** *9*, 684-689.
109. Mulisch, M.; Welsch, U., *Romeis - Mikroskopische Technik*. 19 ed.; Springer Spektrum: **2015**; p 611.
110. ThermoFisherScientific, *User's Guide Amira Software 2019*. FEI SAS, a part of Thermo Fisher Scientific: **2019**.
111. ThermoFisherScientific *Internal Help of Amira*, **2019**.
112. Kittel, C., *Introduction to solid state physics*. 8 ed.; Wiley: **2005**; p 700.
113. Pisco, M.; Galeotti, F., Nano- and Micropatterning on Optical Fibers by Bottom-Up Approach: The Importance of Being Ordered. *Applied Sciences* **2021,** *11*, 3254.
114. Lai, X.; Peng, J.; Cheng, Q.; Tomsia, A. P.; Zhao, G.; Liu, L.; Zou, G.; Song, Y.; Jiang, L.; Li, M., Bioinspired Color Switchable Photonic Crystal Silicone Elastomer Kirigami. *Angew. Chem. Int. Ed.* **2021,** *60*, 14307-14312.
115. Weekes, S. M.; Ogrin, F. Y.; Murray, W. A.; Keatley, P. S., Macroscopic arrays of magnetic nanostructures from self-assembled nanosphere templates. *Langmuir* **2007,** *23*, 1057-1060.
116. Fischer, U. C.; Zingsheim, H. P., Submicroscopic pattern replication with visible light. *Journal of Vacuum Science and Technology* **1981,** *19*, 881-885.
117. Boneberg, J.; Burmeister, F.; Schäfle, C.; Leiderer, P.; Reim, D.; Fery, A.; Herminghaus, S., The Formation of Nano-Dot and Nano-Ring Structures in Colloidal Monolayer Lithography. *Langmuir* **1997,** *13*, 7080-7084.
118. Burmeister, F.; Badowsky, W.; Braun, T.; Wieprich, S.; Boneberg, J.; Leiderer, P., Colloid monolayer lithography-A flexible approach for nanostructuring of surfaces. *Appl. Surf. Sci.* **1999**, 461-466.
119. Stelling, C.; Fossati, S.; Dostalek, J.; Retsch, M., Surface plasmon modes of nanomesh-on-mirror nanocavities prepared by nanosphere lithography. *Nanoscale* **2018,** *10*, 17983-17989.
120. Xie, F.; Centeno, A.; Ryan, M. R.; Riley, D. J.; Alford, N. M., Au nanostructures by colloidal lithography: from quenching to extensive fluorescence enhancement. *Journal of Materials Chemistry B* **2013,** *1*, 536-543.
121. Stelling, C.; Retsch, M., Nanomeshes at Liquid Interfaces: From Free-Standing Hole Arrays toward Metal-Insulator-Metal Architectures. *Advanced Materials Interfaces* **2018,** *5*, 1800154.
122. Stelling, C.; Mark, A.; Papastavrou, G.; Retsch, M., Showing particles their place: deterministic colloid immobilization by gold nanomeshes. *Nanoscale* **2016,** *8*, 14556-14564.
123. Mayarani, M.; Basavaraj, M. G.; Satapathy, D. K., Loosely packed monolayer coffee stains in dried drops of soft colloids. *Nanoscale* **2017,** *9*, 18798-18803.
124. Micheletto, R.; Fukuda, H.; Ohtsu, M., A Simple Method for the Production of a Two-Dimensional, Ordered Array of Small Latex Particles. *Langmuir* **1995,** *11*, 3333-3336.

125. Noppakuadrittidej, P.; Tonsomboon, K.; Ummartyotin, S., Importance of solvent singularity on the formation of highly uniform hexagonal close packed (HCP) colloidal monolayers during spin coating. *Colloid and Interface Science Communications* **2019,** *30*, 100177.

126. Dimitrov, A. S.; Nagayama, K., Continuous Convective Assembling of Fine Particles into Two-Dimensional Arrays on Solid Surfaces. *Langmuir* **1996,** *12*, 1303-1311.

127. Kim, M. H.; Im, S. H.; Park, O. O., Rapid Fabrication of Two- and Three-Dimensional Colloidal Crystal Films via Confined Convective Assembly. *Advanced Functional Materials* **2005,** *15*, 1329-1335.

128. Prevo, B. G.; Velev, O. D., Controlled, rapid deposition of structured coatings from micro- and nanoparticle suspensions. *Langmuir* **2004,** *20*, 2099-2107.

129. Retsch, M.; Zhou, Z.; Rivera, S.; Kappl, M.; Zhao, X. S.; Jonas, U.; Li, Q., Fabrication of Large-Area, Transferable Colloidal Monolayers Utilizing Self-Assembly at the Air/Water Interface. *Macromolecular Chemistry and Physics* **2009,** *210*, 230-241.

130. Vogel, N.; Goerres, S.; Landfester, K.; Weiss, C. K., A Convenient Method to Produce Close- and Non-close-Packed Monolayers using Direct Assembly at the Air-Water Interface and Subsequent Plasma-Induced Size Reduction. *Macromolecular Chemistry and Physics* **2011,** *212*, 1719-1734.

131. Parchine, M.; McGrath, J.; Bardosova, M.; Pemble, M. E., Large Area 2D and 3D Colloidal Photonic Crystals Fabricated by a Roll-to-Roll Langmuir-Blodgett Method. *Langmuir* **2016,** *32*, 5862-5869.

132. Delléa, O.; Lebaigue, O., Boostream: a dynamic fluid flow process to assemble nanoparticles at liquid interface. *Mechanics & Industry* **2017,** *18*, 602-608.

133. Goldenberg, L. M.; Wagner, J.; Stumpe, J.; Paulke, B.-R.; Görnitz, E., Simple Method for the Preparation of Colloidal Particle Monolayers at the Water/Alkane Interface. *Langmuir* **2002,** *18*, 5627-5629.

134. van Dommelen, R.; Fanzio, P.; Sasso, L., Surface self-assembly of colloidal crystals for micro- and nano-patterning. *Adv. Colloid Interface Sci.* **2018,** *251*, 97-114.

135. Isa, L.; Kumar, K.; Muller, M.; Grolig, J.; Textor, M.; Reimhult, E., Particle lithography from colloidal self-assembly at liquid-liquid interfaces. *ACS Nano* **2010,** *4*, 5665-5670.

136. Bähler, P. T.; Zanini, M.; Morgese, G.; Benetti, E. M.; Isa, L., Immobilization of Colloidal Monolayers at Fluid-Fluid Interfaces. *Gels* **2016,** *2*, 19.

137. Jiang, P.; Prasad, T.; McFarland, M. J.; Colvin, V. L., Two-dimensional nonclose-packed colloidal crystals formed by spincoating. *Applied Physics Letters* **2006,** *89*, 011908.

138. Cai, Z.; Liu, Y. J.; Leong, E. S. P.; Teng, J.; Lu, X., Highly ordered and gap controllable two-dimensional non-close-packed colloidal crystals and plasmonic–photonic crystals with enhanced optical transmission. *Journal of Materials Chemistry* **2012,** *22*, 24668-24675.

139. Hummel, M. E. J.; Stelling, C.; Kopera, B. A. F.; Nutz, F. A.; Karg, M.; Retsch, M.; Forster, S., Ordered Particle Arrays via a Langmuir Transfer Process: Access to Any Two-Dimensional Bravais Lattice. *Langmuir* **2019,** *35*, 973-979.

140. Koh, K.; Hwang, H.; Park, C.; Lee, J. Y.; Jeon, T. Y.; Kim, S.-H.; Kim, J. K.; Jeong, U., Large-Area Accurate Position Registry of Microparticles on Flexible, Stretchable Substrates Using Elastomer Templates. *ACS Applied Materials & Interfaces* **2016,** *8*, 28149-28158.

141. Yan, X.; Yao, J.; Lu, G.; Li, X.; Zhang, J.; Han, K.; Yang, B., Fabrication of non-close-packed arrays of colloidal spheres by soft lithography. *Journal of the American Chemical Society* **2005,** *127*, 7688-7689.

142. Li, X.; Wang, T.; Zhang, J.; Yan, X.; Zhang, X.; Zhu, D.; Li, W.; Zhang, X.; Yang, B., Modulating Two-Dimensional Non-Close-Packed Colloidal Crystal Arrays by Deformable Soft Lithography. *Langmuir* **2010,** *26*, 2930-2936.

143. Lehle, H.; Noruzifar, E.; Oettel, M., Ellipsoidal particles at fluid interfaces. *European Physical Journal E* **2008,** *26*, 151-160.

144. Madivala, B.; Fransaer, J.; Vermant, J., Self-assembly and rheology of ellipsoidal particles at interfaces. *Langmuir* **2009,** *25*, 2718-2728.

145. Luo, A. M.; Vermant, J.; Ilg, P.; Zhang, Z.; Sagis, L. M. C., Self-assembly of ellipsoidal particles at fluid-fluid interfaces with an empirical pair potential. *Journal of Colloid and Interface Science* **2019,** *534*, 205-214.

146. Zheng, Z.; Ni, R.; Wang, F.; Dijkstra, M.; Wang, Y.; Han, Y., Structural signatures of dynamic heterogeneities in monolayers of colloidal ellipsoids. *Nature Communications* **2014,** *5*, 3829.

147. Dugyala, V. R.; Basavaraj, M. G., Self-assembly of nano-ellipsoids into ordered structures via vertical deposition. *RSC Advances* **2015,** *5*, 60079-60084.

148. Li, T.; Lilja, K.; Morris, R. J.; Brandani, G. B., Langmuir-Blodgett technique for anisotropic colloids: Young investigator perspective. *Journal of Colloid and Interface Science* **2019,** *540*, 420-438.

149. Zhang, H.; Cadusch, J.; Kinnear, C.; James, T.; Roberts, A.; Mulvaney, P., Direct Assembly of Large Area Nanoparticle Arrays. *ACS Nano* **2018,** *12*, 7529-7537.

150. Lele, P. P.; Furst, E. M., Assemble-and-stretch method for creating two- and three-dimensional structures of anisotropic particles. *Langmuir* **2009,** *25*, 8875-8878.

151. Park, C.; Lee, T.; Xia, Y.; Shin, T. J.; Myoung, J.; Jeong, U., Quick, Large-Area Assembly of a Single-Crystal Monolayer of Spherical Particles by Unidirectional Rubbing. *Advanced Materials* **2014,** *26*, 4633-4638.

152. Younge, K.; Christenson, C.; Bohara, A.; Crnkovic, J.; Saulnier, P., A model system for examining the radial distribution function. *American Journal of Physics* **2004,** *72*, 1247-1250.

153. Kopera, B. A. F.; Retsch, M., Computing the 3D Radial Distribution Function from Particle Positions: An Advanced Analytic Approach. *Anal. Chem.* **2018,** *90*, 13909-13914.

154. Butt, H.-J. G., K.; Kappl, M., Surface forces. In *Physics and Chemistry of Interfaces*, WILEY-VCH GmbH & Co. KGaA: **2003**; pp 80-117.

155. Maestro, A.; Guzmán, E., Colloids at Fluid Interfaces. *Processes* **2019,** *7*, 942.

156. Szilagyi, I.; Sadeghpour, A.; Borkovec, M., Destabilization of Colloidal Suspensions by Multivalent Ions and Polyelectrolytes: From Screening to Overcharging. *Langmuir* **2012,** *28*, 6211-6215.

157. Roach, L.; Hereu, A.; Lalanne, P.; Duguet, E.; Treguer-Delapierre, M.; Vynck, K.; Drisko, G. L., Controlling disorder in self-assembled colloidal monolayers via evaporative processes. *Nanoscale* **2022,** *14*, 3324-3345.

158. Vogel, N.; de Viguerie, L.; Jonas, U.; Weiss, C. K.; Landfester, K., Wafer-Scale Fabrication of Ordered Binary Colloidal Monolayers with Adjustable Stoichiometries. *Advanced Functional Materials* **2011,** *21*, 3064-3073.

159. Tripathi, A. K.; Tsavalas, J. G., A surprisingly gentle approach to cavity containing spherocylindrical microparticles from ordinary polymer dispersions in flow. *Materials Horizons* **2021,** *8*, 2808-2815.

160. Basavaraj, M. G.; Fuller, G. G.; Fransaer, J.; Vermant, J., Packing, flipping, and buckling transitions in compressed monolayers of ellipsoidal latex particles. *Langmuir* **2006,** *22*, 6605-6612.

161. Meng, X.; Qiu, D., Gas-Flow-Induced Reorientation to Centimeter-Sized Two-Dimensional Colloidal Single Crystal of Polystyrene Particle. *Langmuir* **2014,** *30*, 3019-3023.

162. Menath, J.; Mohammadi, R.; Grauer, J. C.; Deters, C.; Bohm, M.; Liebchen, B.; Janssen, L. M. C.; Lowen, H.; Vogel, N., Acoustic Crystallization of 2D Colloidal Crystals. *Advanced Materials* **2023,** *35*, 2206593.

163. Shinotsuka, K.; Kajita, Y.; Hongo, K.; Hatta, Y., Crystal Perfection of Particle Monolayer at the Air-Water Interface. *Langmuir* **2015,** *31*, 11452-11457.

164. Nathan ten Napel, D., Active Anisotropic Colloids from Colloidal Monolayers, Universiteit Utrecht, **2021**.

165. Park, C.; Koh, K.; Jeong, U., Structural Color Painting by Rubbing Particle Powder. *Scientific Reports* **2015,** *5*, 8340.

166. Wang, T.; Li, X.; Zhang, J.; Ren, Z.; Zhang, X.; Zhang, X.; Zhu, D.; Wang, Z.; Han, F.; Wang, X.; Yang, B., Morphology-controlled two-dimensional elliptical hemisphere arrays fabricated by a colloidal crystal based micromolding method. *Journal of Materials Chemistry* **2010,** *20*, 152-158.

167. Xu, H.; Goedel, W. A., Polymer−Silica Hybrid Monolayers as Precursors for Ultrathin Free-Standing Porous Membranes. *Langmuir* **2002,** *18*, 2363-2367.

168. Xu, H.; Goedel, W. A., From particle-assisted wetting to thin free-standing porous membranes. *Angew. Chem. Int. Ed.* **2003,** *42*, 4694-4696.

169. Goedel, W. A.; Glaser, K.; Mitra, D.; Hammerschmidt, J.; Thalheim, R.; Ueberfuhr, P.; Baumann, R. R., Printing Reinforcing Structures onto Microsieves That Are Floating on a Water Surface. *Langmuir* **2019,** *35*, 2196-2208.

170. Yan, F.; Goedel, W. A., Polymer Membranes with Two-Dimensionally Arranged Pores Derived from Monolayers of Silica Particles. *Chemistry of Materials* **2004,** *16*, 1622-1626.

171. Yan, F.; Ding, A.; Girones, M.; Lammertink, R. G. H.; Wessling, M.; Borger, L.; Vilsmeier, K.; Goedel, W. A., Hierarchically Structured Assembly of Polymer Microsieves, made by a Combination of Phase Separation Micromolding and Float-Casting. *Advanced Materials* **2012,** *24*, 1551-1557.

172. Wachner, D.; Marczewski, D.; Goedel, W. A., Utilising Spontaneous Self-organization of Particles to Prepare Asymmetric, Hierarchical Membranes Comprising Microsieve-Like Parts. *Advanced Materials* **2013,** *25*, 278-283.

173. Vogel, N.; Ally, J.; Bley, K.; Kappl, M.; Landfester, K.; Weiss, C. K., Direct visualization of the interfacial position of colloidal particles and their assemblies. *Nanoscale* **2014,** *6*, 6879-6885.

174. Paunov, V. N., Novel Method for Determining the Three-Phase Contact Angle of Colloid Particles Adsorbed at Air−Water and Oil−Water Interfaces. *Langmuir* **2003,** *19*, 7970-7976.

175. Cayre, O. J.; Paunov, V. N., Fabrication of microlens arrays by gel trapping of self-assembled particle monolayers at the decane–water interface. *Journal of Materials Chemistry* **2004,** *14*, 3300-3302.

176. von Ardenne, M., Das Elektronen-Rastermikroskop. *Zeitschrift für Physik* **1938,** *109*, 553-572.

177. Czichos, H.; Saito, T.; Smith, L., *Springer Handbook of Materials Measurement Methods*. Springer: **2006**; p 1215.

178. Ali Salman, A., Application of Nanomaterials in Environmental Improvement. In *Nanotechnology and the Environment*, Mousumi, S., Ed. IntechOpen: Rijeka, **2020**; pp 1-20.

179. Everhart, T. E.; Thornley, R. F. M., Wide-band detector for micro-microampere low-energy electron current. *Journal of Scientific Instruments* **1960,** *37*, 246-248.

180. Skoog, D. A.; Holler, F. J.; Crouch, S. R., *Instrumentelle Analytik: Grundlagen - Geräte - Anwendungen*. 6 ed.; Springer Spektrum: **2013**; p 901.

181. Giddings, J. C., A New Separation Concept Based on a Coupling of Concentration and Flow Nonuniformities. *Separation Science* **1966,** *1*, 123-125.

182. Wahlund, K.-G.; Giddings, J. C., Properties of an Asymmetrical Flow Field-Flow Fractionation Channel Having One Permeable Wall. *Anal. Chem.* **1987,** *59*, 1332-1339.

183. Wahlund, K.-G., Asymmetrical Flow Field-Flow Fractionation. In *Field-Flow Fractionation Handbook*, Schimpf, M. E.; Caldwell, K.; Giddings, J. C., Eds. Wiley: **2000**.

184. Wagner, M.; Holzschuh, S.; Traeger, A.; Fahr, A.; Schubert, U. S., Asymmetric flow field-flow fractionation in the field of nanomedicine. *Anal. Chem.* **2014,** *86*, 5201-5210.

185. Chilingar, G. V.; Haroun, M.; Shojaei, H.; Shin, S., Chapter 1.2 Zeta Potential and the Electric Double Layer Interaction. In *Electrokinetics for Petroleum and Environmental Engineers*, Scrivener Publishing LLC: **2014**; pp 1-265.

186. Moulder, J. F.; Stickle, W. F.; Sobol, P. E.; Bomben, K. D., *Handbook of X-ray Photoelectron Spectroscopy*. Perkin-Elmer Corporation: **1992**; p 260.

187. Siegbahn, K., Electron Spectroscopy - An Outlook. *J. Electron. Spectrosc. Relat. Phenom.* **1974,** *5*, 3-97.

188. Gill, P. S.; Sauerbrunn, S. R.; Reading, M., Modulated Differential Scanning Calorimetry. *J. Therm. Anal.* **1993,** *40*, 931-939.

189. Marti, E.; Kaisersberger, E.; Moukhina, E., Heatcapacity functions of polystyrene in glassy and in liquid amorphous stateand glass transition. *J. Therm. Anal. Calorim.* **2006,** *85*, 505-525.

VIII.3. Table of Figures

Chapter IV

Chapter V

Chapter VI

Chapter VIII – Appendix

VIII.4. Table of Tables

VIII.5. Table of Equations

VIII.6. Python Code

VIII.6.1 Python Code of the Ellipse Detection Program

An executable version of my ellipse detection program (version 0.54) can be found on the enclosed CD-ROM, or on the download page for this thesis at University Bayreuth.

The python program is divided into 5 separate files that were accessed by the main file in a linear order. The individual files are named as tab 1 – 4.

VIII.6.1.1 Tab Main

```
#######################################################################
# #####    Dominik Benke, M.Sc. programed this app during his   ##### #
# #####      PhD thesis at University Bayreuth 2019-2024        ##### #
#######################################################################

# import own source code
import sourceCode_ellipseDetection.tab1_startPage_v6 as tab1_startPage
import sourceCode_ellipseDetection.tab2_Page_Area_detection_v8 as tab2_Page_Area_detection
import sourceCode_ellipseDetection.tab3_Page_Area_Selection_v10 as tab3_Page_Area_Selection
import sourceCode_ellipseDetection.tab4_Page_EllipsePicking_v10 as tab4_Page_EllipsePicking

# import packages
import matplotlib
from matplotlib.figure import Figure
from matplotlib.pyplot import rcParams
import matplotlib.pyplot as plt
import numpy as np
import tkinter as tk
from tkinter import ttk
from tkinter import filedialog
from skimage import feature
from skimage import color, img_as_ubyte, io
from skimage.filters import rank
from skimage.segmentation import watershed
from skimage.morphology import disk
import os
from scipy import ndimage as ndi
from skimage.measure import regionprops
import datetime

# global plot parameters
matplotlib.use('TkAgg')
LARGE_FONT = ('Verdana', 12)
GlobFontSize = 10
params = {
    'axes.labelsize': GlobFontSize,
    'font.size': GlobFontSize,
    'axes.titlesize': GlobFontSize,
    'font.family': 'Calibri',
    'legend.fontsize': GlobFontSize,
    'xtick.labelsize': GlobFontSize,
    'ytick.labelsize': GlobFontSize,
    'text.usetex': False,
    'figure.constrained_layout.use': True,
    'lines.markersize': 3.0,
    'boxplot.flierprops.markersize': 3
}
rcParams.update(params)
```

```
# app code
class EllipsoidDetectionApp(tk.Tk):
    version = 'Version 0.54'
    print('Py file location: ', os.getcwd())
    print('used py file path: ', __file__)
    # administration for variables
    progr_count = 1
    variable_1 = 1.0
    # For diagnosis it is possible to print more data on console
    console_print_on = False
    initialdirectory = 'C:/'

    # load dummypic file from folder Testdata
    dummydirectory = os.getcwd() + '\\testdata'
    print('The dummy images are located in: ' + str(dummydirectory))
    dummypic = dummydirectory + '\\190607_DBmL_41B_14.bmp'
    # other dummys in folder:
    # '\\220125DBmLR61_DB13_15%_quer_0%_pos1__014_LSM2D.jpg'
    # '\\190607_DBmL_41A_10.tif'

    ## dummys for variables interchange ##
    # create all varaibles and images that are necessary for interchange
    # in between the different tabs of the program
    image = color.rgb2gray(io.imread(dummypic)[:630])
    image_rgb = color.rgb2gray(io.imread(dummypic)[:630])
    original = color.rgb2gray(io.imread(dummypic)[:630])
    # imagesize and new_imagesize is array of (height, width)
    imagesize = image_rgb.shape
    new_imagesize = image_rgb.shape
    # show initial dummy pxl ratio
    pxl_ratio = 11.16 #21.1
    edges = feature.canny(image_rgb)
    gradient = rank.gradient(img_as_ubyte(edges), disk(3)) < 10
    markers = ndi.label(gradient)[0]
    labels = watershed(edges, markers)
    ParticleProp = regionprops(labels)
    particles_at_border = False
    # create an empty image with size of dummypic
    # np.zero needs dimension of height, width --> using new_imagesize in normal order
    image_ellipses = np.zeros([new_imagesize[0], new_imagesize[1], 3])
    image_numbers = image_ellipses
    ellipse_draw_list = []
    image_path = ''
    ParticleList = []
    final_ParticleList = []
    WorkingList = []
    type_threshold = ''
    filter_used_list = []
    final_polar_plot = Figure()
    analyse_orientations_img = Figure()
    # figure needs size in width, height -> turn array of img.shape
    # --> new_imagesize[1], new_imagesize[0]
    finalfig_orgdim = Figure(figsize=(new_imagesize[1] / 100, new_imagesize[0] / 100))

    # initial parameters for sliders
    sigma = 0.9
    low_threshold = 0.40
    high_threshold = 200
    labelLines_parameter = 1

    # parameters for in frame image size
    img_x = 6.0
    img_y = 3.7
    finalfigure = Figure(figsize=(img_x, img_y))

    # Initialization of the program
    # arguments and kwargs = keyword arguments (dictionaries)
    def __init__(self, *args, **kwargs):
        ## prepare GUI ##
        tk.Tk.__init__(self, *args, **kwargs)
        tk.Tk.wm_title(self, '2D Ellipse Particle Detector')
        # make Window fit full screen
        # but for images in thesis: 1000,700 instead screenwidth and pad
        pad = 2
        self.geometry('{0}x{1}+0+0'.format(self.winfo_screenwidth() - pad,
                self.winfo_screenheight() - pad))
        self.update()
        # print in console if parameter above is set True
        if self.console_print_on: print(self.winfo_width())
        if self.console_print_on: print(self.winfo_height())

        # define figure size within the full screen
        dpi = 100
        self.img_x = round(((self.winfo_width() - 220) / dpi) / 2, 2)   # for fig III-17 use 3
        self.img_y = round(((self.winfo_height() * 0.85) / dpi) / 2, 2)   # for fig III-17 use 0.6
        if self.console_print_on: print(self.img_x)
        if self.console_print_on: print(self.img_y)

        ## Implementation of menu bar ##
        menubar = tk.Menu(self)
        # 'File' dropdown menu
        filemenu = tk.Menu(menubar, tearoff=0)
        filemenu.add_command(label='New', command=lambda: self.open_new(self.tabControl))
        filemenu.add_command(label='New with Evaluation File',
                command=lambda: [self.load_evaluation_parameter(), self.open_new(self.tabControl)])
        self.load_menu_point = filemenu.add_command(label='Load Evaluation Parameter',
                command=self.load_evaluation_parameter, state='normal')
        filemenu.add_command(label='Close', command=self.close_current)
        menubar.add_cascade(label='File', menu=filemenu)
        # 'Edit' dropdown menu
        self.editmenu = tk.Menu(menubar, tearoff=0)
        self.automated_crop = tk.BooleanVar()
        self.editmenu.add_checkbutton(label='Automated Line finder', variable=self.automated_crop)
        self.no_crop = tk.BooleanVar()
        self.editmenu.add_checkbutton(label='No crop', variable=self.no_crop)
        self.image_w_axes = tk.BooleanVar()
        self.image_w_axes.set(True)
        self.editmenu.add_checkbutton(label='Images with axes', variable=self.image_w_axes, )
        self.thresholding_var = tk.BooleanVar()
        self.thresholding_var.set(False)
        self.editmenu.add_checkbutton(label='Thresholding Image', variable=self.thresholding_var)
        self.save_angle_dist = tk.BooleanVar()
        self.save_angle_dist.set(True)
        self.editmenu.add_checkbutton(label='Save angle distribution, too',
                variable=self.save_angle_dist)
        self.editmenu.add_separator()
        self.consol_print = tk.BooleanVar()
        self.editmenu.add_checkbutton(label='Console print out on',
                variable=self.consol_print, command=self.toogle_consoleprint)
        self.editmenu.add_command(label='Credits', command=self.credits_win)
        menubar.add_cascade(label='Edit', menu=self.editmenu)
        # mount menubar on window
        self.menubar = menubar
        self.config(menu=self.menubar)

        # create dummy tabs
        self.new_tabcontrol()

        # start program
        self.new_start()

    # switch console print outs on or off by menu button
    def toogle_consoleprint(self):
        if self.consol_print.get():
            self.console_print_on = True
            print('console on')
        else:
            self.console_print_on = False
            print('console off')

    # create new dummy tabs
    def new_tabcontrol(self):
        # tab control frames
        tabControl = ttk.Notebook(self)
        tabControl.pack(side='top', fill='both', expand=True)
        # #tabID 0
        self.tab1 = ttk.Frame(tabControl)
        tabControl.add(self.tab1, text='Image')
        # #tabID 1
        self.tab2 = ttk.Frame(tabControl)
        tabControl.add(self.tab2, text='Edge Detection')
        # #tabID 2
        self.tab3 = ttk.Frame(tabControl)
        tabControl.add(self.tab3, text='Area Selection')
        # #tabID 3
        self.tab4 = ttk.Frame(tabControl)
        tabControl.add(self.tab4, text='Ellipse Picking')  # Statistics & Evaluation
        # #hide tabs 1-3, show tab 0
        tabControl.hide(1)
        tabControl.hide(2)
        tabControl.hide(3)
        tabControl.pack(side='top', fill='both', expand=True)
        self.tabControl = tabControl

    # start the evaluation of an image by loading the image
    # with or without parameters in new popup frame
    def new_start(self):
        choose = tk.Tk()
        choose.geometry('250x110')
        choose.wm_title('Start up')
        choose_label1 = tk.Label(master=choose,
                text='Welcome to Dominik Benkes ellipse detection.', anchor='center')
        choose_label2 = tk.Label(master=choose, text=self.version)
        choose_label1.grid(row=0, sticky='EW')
        choose_label2.grid(row=1)
        choose_button1 = ttk.Button(master=choose, text='Load File',
                command=lambda: [self.next_step(1), choose.destroy()])
        choose_button1.grid(row=3)
        choose_button1 = ttk.Button(master=choose, text='Load with pre-set for evaluation',
                command=lambda: [self.load_evaluation_parameter(), self.next_step(1),
                choose.destroy()], state='enabled')
        choose_button1.grid(row=4)
        self.progr_count = 1

    # go to the next part/tab of the evaluation by clicking on the respective button in the tab
    # the program counter increases
    def next_step(self, progr_count):
        self.progr_count = progr_count
        # bind content of next steps to frames/tabs
        if self.progr_count == 1:
            if self.console_print_on: print('Program counter: ' +str(self.progr_count))
            self.load_image()
            # north south east west = nsew
            self.content_tab1 = tab1_startPage.StartPage(self.tab1, self)
            self.content_tab1.grid(row=0, column=0, sticky='nsew')
        elif self.progr_count == 2:
            if self.console_print_on: print('Program counter: ' +str(self.progr_count))
            self.content_tab2 = tab2_Page_Area_detection.Page_Area_detection(self.tab2, self)
            self.content_tab2.grid(row=0, column=0, sticky='nsew')
        elif self.progr_count == 3:
            if self.console_print_on: print('Program counter: ' +str(self.progr_count))
            self.content_tab3 = tab3_Page_Area_Selection.Page_Area_Selection(self.tab3, self)
            self.content_tab3.grid(row=0, column=0, sticky='nsew')
        elif self.progr_count == 4:
            if self.console_print_on: print('Program counter: ' +str(self.progr_count))
            self.content_tab4 = tab4_Page_EllipsePicking.Page_EllipsePicking(self.tab4, self)
            self.content_tab4.grid(row=0, column=0, sticky='nsew')
        else:
            None
        if self.console_print_on: print('Evaluations since program start: ' + str(self.variable_1))
        if self.console_print_on: print(self.tabControl.index('end'))


    # show the chosen tab on the window
    def show_frame(self, cont):
        self.tabControl.select(cont)
```

```
# close the current evaluation
def close_current(self):
    variable_2 = self.variable_1+1
    print(variable_2)
    self.variable_1 = variable_2
    self.tabControl.destroy()

# start a new evaluation
def open_new(self, container):
    self.new_dummys()
    # Refresh of tab control
    self.tabControl.destroy()
    self.new_tabcontrol()
    self.progr_count = 1
    self.next_step(1)
    if self.console_print_on: print('open new file')

# save the data from all tabs
def write_datafile(self):
    '''
    Variables to be saved:
    file A:     self.final_ParticleList  //List

    file B:     self.version
                self.progr_count
                self.initialdirectory
                self.pxl_ratio
                self.thresholding_var
                self.particles_at_border
                self.sigma
                self.low_threshold
                self.high_threshold
                self.labelLines_parameter
                self.ellipse_draw_list   //List

    Img colored ellipses:       self.finalfigure
    Plot of orientations:       self.final_polar_plot
    '''
    self.save_window()
    #progress bar for saving
    self.progressbar.step(10)
    # save parameters of tab 1
    self.content_tab1.make_save_update_from_tab1(self)
    self.progressbar.step(10)
    # save parameters of tab 2, if present
    if self.progr_count>1:
        self.content_tab2.make_save_update_from_tab2(self)
    self.progressbar.step(10)
    # save parameters of tab 3, if present
    if self.progr_count>2:
        self.content_tab3.make_save_update_from_tab3(self)
    self.progressbar.step(10)
    # save parameters of tab 4, if present
    if self.progr_count>3:
        self.content_tab4.make_save_update_from_tab4(self)
    self.progressbar.step(10)

    ## write file A - Datafile ##
    # #create filename with timestamp
    time =  datetime.datetime.now().strftime('%Y-%m-%d %H-%M-%S')
    split_name = os.path.splitext(str(self.image_path[1]))
    new_name = str(self.image_path[0])+'\\'+str(split_name[0])+'_evaluation_'+time+'.txt'
    if self.console_print_on: print(new_name)

    # #write ellipse details to file
    print_List = self.final_ParticleList
    if print_List!=[]:
        fileA =  open(str(new_name),'w')
        for i in print_List:
            # removes brakets on start and end
            # + Add filename in front, just not on first row (column names)
            if str(i)[0] =='P':
                j = str(self.file_name)+', '+str(i)
            else:
                j = str(self.file_name)+', '+str(i)[1:-1]
            fileA.write('%s\n' % j)
        fileA.close()
        if self.console_print_on: print('Particle List saved')
    else:
        if self.console_print_on: print('No particle data to save')
    self.progressbar.step(25)

    ## save figures ##
    # figure with axes but with low white space around (bbox and pad_inches)
    self.finalfigure.savefig((str(self.image_path[0]) + '\\' + str(split_name[0])
            + '_evaluation_' + time + '.png'), dpi=100, bbox_inches='tight', pad_inches=0)
    # figures polar plot and orientation analysis, if used at tab 4
    if self.save_angle_dist.get():
        self.final_polar_plot.savefig((str(self.image_path[0]) + '\\' + str(split_name[0])
            + '_orientation_' + time + '.png'), dpi=155, bbox_inches='tight', pad_inches=0)
    if len(self.analyse_orientations_img.get_children()) > 1:
        self.analyse_orientations_img.savefig((str(self.image_path[0]) + '\\'
            + str(split_name[0]) + '_analysed_orientations_' + time + '.png'), dpi=300,
            bbox_inches='tight', pad_inches=0)
        print('analysed orientations' + str(len(self.analyse_orientations_img.get_children())))
    # save the final figure with overlay and the overlay only; both with original dimensions
    # same scale (pxl/nm) as original image
    self.finalfig_orgdim.savefig((str(self.image_path[0]) + '\\' + str(split_name[0])
            + '_evaluation2_' + time + '.png'), dpi=1)
    plt.imsave(fname=(str(self.image_path[0]) + '\\' + str(split_name[0]) + '_evaluation3_'
            + time + '.png'), arr=self.image_ellipses, format='png')

    ## write file B - config-data for re-evaluation ##
    config_file_name = str(self.image_path[0]) + '\\' + str(split_name[0]) + '_progr_data_'
            + time + '.txt'
    if self.console_print_on: print(config_file_name)
    fileB = open(str(config_file_name), 'w')
    fileB.write('Last modified: ' + time + '\n')
    fileB.write('Program ' + self.version)
    fileB.write(' Program counter: ' + str(self.progr_count - 1) + '\n')
    fileB.write('Tab 1' + '\n')
    fileB.write(' Initial directory: ' + str(self.initialdirectory) + '\n')
    fileB.write(' Pxl ratio: ' + str(self.pxl_ratio) + '\n')
    fileB.write(' Used Threshold: ' + str(self.thresholding_var.get()) + '\n')
    fileB.write(' Which Threshold: ' + str(self.type_threshold) + '\n')
    fileB.write('Tab 2' + '\n')
    fileB.write(' Border? ' + str(self.particles_at_border) + '\n')
    fileB.write(' Sigma: ' + str(self.sigma) + '\n')
    fileB.write(' Low Threshold: ' + str(self.low_threshold) + '\n')
    fileB.write(' High Threshold: ' + str(self.high_threshold) + '\n')
    fileB.write(' LabelsLine Parameter: ' + str(self.labelLines_parameter) + '\n')
    fileB.write('Areas were detected and refined at ')
    fileB.write('Tab 3' + '\n')
    fileB.write(' After filtering with \n')
    fileB.write(str(self.filter_used_list))
    fileB.write(', those ellipsoids were drawn on the image \n')
    for i in self.ellipse_draw_list:
        fileB.write('%s\n' % i)
    fileB.write('maybe some were then kicked by hand to get the list in the file: evaluation')
    fileB.close()

    self.progressbar.step(24)

    labelo = tk.Label(master=self.save_sucess, text='Saving successful!')
    labelo.pack(pady=10, padx=10)
    butt_save_win = ttk.Button(master=self.save_sucess, text='OK',
            command=lambda: self.save_sucess.destroy())
    butt_save_win.pack(pady=10, padx=10)


# load the config-data of previous evaluated images
def load_evaluation_parameter(self):
    self.ev_filename = filedialog.askopenfilename(initialdir=self.initialdirectory,
            title='Select evaluation file', filetypes=(('Evaluation config', '*.txt'),
            ('All files', '*.*')))
    ev = str(self.ev_filename)
    # catch error if filedialog is closed without choosing a file
    if ev:
        file_ev = open(ev, 'r')
        ev_content = file_ev.readlines()
        file_ev.close()
        if self.console_print_on: print(ev_content)
        k = 0
        for i in ev_content:
            if 'Pxl ratio:' in i:
                if self.console_print_on: print(ev_content[4])
                self.pxl_ratio = float(i.split(': ')[1])
                if self.console_print_on: print(self.pxl_ratio)
                k = 1
            if ' Border? ' in i:
                particles_at_border = str(i.split('? ')[1])
                print(particles_at_border)
                k = 1
            if ' Sigma: ' in i:
                self.sigma = float(i.split(': ')[1])
                k = 1
            if ' Low Threshold: ' in i:
                self.low_threshold = float(i.split(': ')[1])
                k = 1
            if ' High Threshold: ' in i:
                self.high_threshold = float(i.split(': ')[1])
                k = 1
            if ' LabelsLine Parameter: ' in i:
                self.labelLines_parameter = float(i.split(': ')[1])
                k = 1
        if k == 1:
            if self.console_print_on: print('some parameters from file successfully loaded')
        else:
            if self.console_print_on: print('no parameter in file')
    if self.progr_count == 2:
        # update pxl ratio
        self.content_tab1.update_pxlratio_tab1(self.pxl_ratio)
    else:
        print(self.progr_count)

# save window that replies successful data saved with status bar
def save_window(self):
    self.save_sucess = tk.Tk()
    self.save_sucess.geometry('140x120')
    self.progressbar = ttk.Progressbar(master=self.save_sucess, length=200, mode='determinate',
            maximum=100)
    self.progressbar.pack()
    self.save_sucess.focus_force()
    self.save_sucess.lift()

# credits window
def credits_win(self):
    credits = tk.Tk()
    credits.geometry('140x70')
    label1 = tk.Label(master=credits, text=self.version)
    label1.pack(padx=10)
    label2 = tk.Label(master=credits, text='Copyright 2024')
    label2.pack(pady=5, padx=10)
    label3 = tk.Label(master=credits, text='Dominik Benke, M.Sc.')
    label3.pack(padx=10)

# load image to the program
def load_image(self):
    if self.console_print_on: print(self.automated_crop.get())
    # open image file and parse it to self.original and self.image_rgb
    self.filename = filedialog.askopenfilename(initialdir=self.initialdirectory,
            title='Select file', filetypes=(('Images', '*.tif *.bmp *.jpg'), ('All files',
            '*.*'), ('Tiffs', '*.tif'), ('Bitmaps', '*.bmp'), ('Jpegs', '*.jpg')))
    a = str(self.filename)
    # catch error if filedialog is closed without choosing a file
    # dummypic is used
    if a == '':
        a = self.dummypic
    # save image path at global variables
```

```
        self.image_path = os.path.split(os.path.abspath(a))
        self.initialdirectory = self.image_path[0]
        if self.console_print_on: print('The image directory: ' + str(self.image_path))
        self.file_name = self.image_path[1]
        if self.console_print_on: print('The image name: ' + str(self.image_path[1]))
        # add imagename to the menu bar
        self.menubar.add_command(label=self.file_name,
                command=lambda: self.open_new(self.tabControl))
        self.config(menu=self.menubar)

        ## image read and crop ##
        # #global original
        self.original = io.imread(a)
        # #global imagesize
        self.imagesize = self.original.shape
        if self.console_print_on: print(self.imagesize[0])
        if self.console_print_on: print(self.imagesize[1])
        # if automated_crop =1 (ON, toggle at menu): automated cutting of the annotation in the img
        # by finding the first horizontal pixel row with highest color black
        # (value 0, but not all pixel have this) and cutting picture there
        self.cutoff_pic = 0
        if self.automated_crop.get():
            self.grey_complete = img_as_ubyte(io.imread(a, as_gray=True))
            if self.console_print_on: print(self.original[689:693])
            if self.console_print_on: print(self.grey_complete[689:693])
            blackline_y_list = []
            for y in self.grey_complete:
                line_add = 0
                for x in y:
                    if x < 10:
                        line_add = line_add + 1
                blackline_y_list.append(line_add)
            if self.console_print_on: print(blackline_y_list)
            if self.console_print_on: print('The maximum of black pxl in a line is at position: '
                        + str(np.argmax(blackline_y_list)))
            self.cutoff_pic = int(np.argmax(blackline_y_list)-1)
        # if no_crop is ON (toggle at menu) use full size of image
        # if no_crop is OFF the automated crop or a standard crop at 90% will be done.
        # 90% crop = crop standard space of annotations
        # at University Bayreuths SEM images from ZEISS microscopes
        if self.no_crop.get():
            y = self.imagesize[0]
        else:
            if self.cutoff_pic<10:
                y = int(round(self.imagesize[0]*0.9, 0))
            else:
                y = self.cutoff_pic
        self.image_rgb = io.imread(a, as_gray = True)[:y]
        if self.console_print_on: print(self.image_rgb.shape)
        if self.console_print_on: print(self.image_rgb)
        # transform rgb image to appropriate color/grey value format
        self.image_rgb = img_as_ubyte(self.image_rgb)
        self.image_loaded = self.image_rgb.copy()
        if self.console_print_on: print(self.image_rgb.shape)
        if self.console_print_on: print(self.image_rgb)
        # global new_imagesize array of (height, width)
        self.new_imagesize = self.image_rgb.shape
        if self.console_print_on: print(self.image_rgb.shape)

    # new dummys for (re-)start evaluation
    def new_dummys(self):
        if self.console_print_on: print('new dummys')
        ## dummys for global variables ##
        self.image_rgb = color.rgb2gray(io.imread(self.dummypic)[:630])
        self.original = []
        # imagesize and new_imagesize are arrays of (height, width)
        self.imagesize = self.image_rgb.shape
        self.new_imagesize = self.image_rgb.shape
        self.edges = feature.canny(self.image_rgb)
        self.gradient = rank.gradient(img_as_ubyte(self.edges), disk(3)) < 10
        self.markers = ndi.label(self.gradient)[0]
        self.labels = watershed(self.edges, self.markers)
        self.ParticleProp = regionprops(self.labels)
        self.particles_at_border = False
        self.image_path = []
        # np.zeros need (height, width), so using new_imagesize in normal order is OK
        self.image_ellipses = np.zeros([self.new_imagesize[0], self.new_imagesize[1], 3])
        self.image_numbers = self.image_ellipses
        self.ellipse_draw_list = []
        self.ParticleList = []
        self.final_ParticleList = []
        self.WorkingList = []

# start main application
def main():
    app = EllipsoidDetectionApp()
    app.mainloop()

if __name__ == '__main__':
    main()
```

VIII.6.1.2 Tab 1

```
################################
# ############################ #
# ### Tab 1 - Start Page - ### #
# ############################ #
################################

# import packages
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.backends.backend_tkagg import NavigationToolbar2Tk
from matplotlib.figure import Figure
from matplotlib.pyplot import rcParams
import tkinter as tk
from tkinter import ttk
from skimage.filters import threshold_otsu, threshold_local

# global plot parameters
matplotlib.use("TkAgg")
LARGE_FONT = ("Verdana", 12)
GlobFontSize = 10
params = {
    'axes.labelsize': GlobFontSize,
    'font.size': GlobFontSize,
    'axes.titlesize': GlobFontSize,
    'font.family': "Calibri",
    'legend.fontsize': GlobFontSize,
    'xtick.labelsize': GlobFontSize,
    'ytick.labelsize': GlobFontSize,
    'text.usetex': False,
    'figure.constrained_layout.use': True,
    'lines.markersize': 3.0,
    'boxplot.flierprops.markersize': 3
}
rcParams.update(params)

# tab 1 code
class StartPage(tk.Frame):
    def __init__(self, parent, controller):
        # get variables from controller
        self.console_print_on = controller.console_print_on
        self.image_rgb = controller.image_rgb
        self.original = controller.original
        self.img_x = controller.img_x
        self.img_y = controller.img_y
        self.thresholding_var = controller.thresholding_var.get()
        self.image_w_axes = controller.image_w_axes.get()

        # make working copy of greyscale & resized image
        self.image = self.image_rgb
        ## create tab layout ##
        tk.Frame.__init__(self, parent)
        # # devide window into 2 frames
        tab1_left = ttk.Frame(master=self)
        tab1_left.grid(row=0, column=0, sticky="nw")
        tab1_left.columnconfigure(0, minsize=220)
        self.tab1_right = ttk.Frame(master=self, relief="groove")
        self.tab1_right.grid(row=0, column=1, sticky="nsew")

        ## make menu on the left of tab 1 with buttons and field for pxl ratio ##
        butt1_2 = ttk.Button(master=tab1_left, text="Use Threshold",
                command=lambda: self.thresholding_images(controller))
        butt1_2.grid(row=0, column=0, pady=10, sticky='ew')
        label = tk.Label(master=tab1_left, text="Pixel ratio in nm/pxl from SEM image
                \n (use dot, not comma):")
        label.grid(row=1, column=0, rowspan=1)
        self.entryfield_pxlratio = tk.Entry(master=tab1_left)
        self.entryfield_pxlratio.grid(row=2, column=0, rowspan=1)
        self.entryfield_pxlratio.insert(0, str(controller.pxl_ratio))
        ttk.Separator(master=tab1_left, orient='horizontal').grid(row=3, column=0, pady=10,
                sticky="ew")
        butt1 = ttk.Button(master=tab1_left, text=">> Go to Canny edge detection",
                command=lambda: self.update_controller_from_tab1(controller))
        butt1.grid(row=4, column=0, sticky='ew')

        # display images on right side of tab 1
        self.show_original_image()
        self.show_gray_image()
        # if wanted one can threshold the image and choose the appropriate image to go further
        if self.thresholding_var:
            self.thresholding_images(controller)

    # show original image on tab 1
    def show_original_image(self):
        fig1 = Figure(figsize=(self.img_x, self.img_y + 2))
        a = fig1.add_subplot(111)
        a.imshow(self.original)
        if self.image_w_axes:
            a.set_title("Original")
            a.set_xlabel("Horizontal Pixel ID")
            a.set_ylabel("Vertical Pixel ID")
        canvas1_1 = FigureCanvasTkAgg(fig1, master=self.tab1_right)
        canvas1_1.get_tk_widget().grid(row=0, column=0, rowspan=1)
        # toolbar
        toolbar = NavigationToolbar2Tk(canvas1_1, self.tab1_right, pack_toolbar=False)
        toolbar.update()
        toolbar.grid(row=1, column=0, rowspan=1)

    # show converted to grayscale and resized image
    def show_gray_image(self):
        fig2 = Figure(figsize=(self.img_x, self.img_y + 2))
        a2 = fig2.add_subplot(111)
        a2.imshow(self.image, cmap="gray")
        a2.set_title("Grayscale & Resized")
        a2.set_xlabel("Horizontal Pixel ID")
        a2.set_ylabel("Vertical Pixel ID")
        canvas1_2 = FigureCanvasTkAgg(fig2, master=self.tab1_right)
        canvas1_2.get_tk_widget().grid(row=0, column=1, rowspan=1, columnspan=2)
        # toolbar
        toolbar = NavigationToolbar2Tk(canvas1_2, self.tab1_right, pack_toolbar=False)
        toolbar.update()
        toolbar.grid(row=1, column=1, rowspan=1)

    # thresholding the image and show result in a new window
    def thresholding_images(self, controller):
        self.threshold_window = tk.Tk()

        # local thresholding
        block_size = 35
        local_thresh = threshold_local(self.image, block_size)
        self.binary_local = self.image < local_thresh
        fig = Figure(figsize=(self.img_x, self.img_y))
        thres_plot = fig.add_subplot(111)
        thres_plot.imshow(self.binary_local, cmap="gray")
        thres_plot.set_title("local thresholding")
        thres_plot.set_xlabel("Horizontal Pixel ID")
        thres_plot.set_ylabel("Vertical Pixel ID")
        canvas3 = FigureCanvasTkAgg(fig, self.threshold_window)
        canvas3.get_tk_widget().grid(row=3, column=0, rowspan=1)

        # global thresholding (otsu)
        global_thresh = threshold_otsu(self.image)
        self.binary_global = self.image < global_thresh
        figa = Figure(figsize=(self.img_x, self.img_y))
        thresplot2 = figa.add_subplot(111)
        thresplot2.imshow(self.binary_global, cmap="gray")
        thresplot2.set_title("global thresholding")
        thresplot2.set_xlabel("Horizontal Pixel ID")
        thresplot2.set_ylabel("Vertical Pixel ID")
        canvas4 = FigureCanvasTkAgg(figa, self.threshold_window)
        canvas4.get_tk_widget().grid(row=3, column=1, rowspan=1, columnspan=2)

        # chose which threshold to use by buttons
        ttk.Button(self.threshold_window, text="Use binary local",
                command=lambda: self.checkbx_local(controller)).grid(row=4, column=0)
        ttk.Button(self.threshold_window, text="Use binary global",
                command=lambda: self.checkbx_global(controller)).grid(row=4, column=1)

    # routine after click on button for local thresholding
    def checkbx_local(self, controller):
        self.image = self.binary_local
        # new initial parameters for sliders at tab 2
        controller.sigma = 2
        controller.low_threshold = 0
        controller.high_threshold = 1
        controller.labelLines_parameter = 2
        controller.thresholding_var.set(True)
        controller.type_threshold = 'binary_local'
        self.threshold_window.destroy()

    # routine after click on button for global thresholding
    def checkbx_global(self, controller):
        self.image = self.binary_global
        # new initial parameters for sliders at tab 2
        controller.sigma = 3.0
        controller.low_threshold = 0
        controller.high_threshold = 1
        controller.labelLines_parameter = 2
        controller.thresholding_var.set(True)
        controller.type_threshold = 'binary_global_otsu'
        self.threshold_window.destroy()

    # routine after click on button '>> Go to Canny edge detection'
    def update_controller_from_tab1(self, controller):
        # hadover of working copy of image to controller
        controller.image = self.image
        # read pxl_ratio
        f = self.entryfield_pxlratio.get()
        try:
            controller.pxl_ratio = float(f)
            controller.next_step(2)
            controller.show_frame(1)
        # if wrong/no number in entry field
        except ValueError:
            if self.console_print_on: print("oops, default: " + str(controller.pxl_ratio))
            self.error_catcher(controller)
        if self.console_print_on: print(f)

    # handle exception of wrong number in entry field
    def error_catcher(self, controller):
        self.error_frame = tk.Tk()
        self.error_frame.geometry("220x100")
        label_text = "wrong value, please write new"
        label = tk.Label(master=self.error_frame, text=label_text)
        label.pack(pady=10, padx=10)
        self.newentry = tk.Entry(master=self.error_frame)
        self.newentry.pack()
        butt1 = tk.Button(master=self.error_frame, text="read",
                command=lambda: self.try_read_again(controller))
        butt1.pack()

    # handle exception if still a wrong number in entry field
    def try_read_again(self, controller):
        try:
            controller.pxl_ratio = float(self.newentry.get())
            if self.console_print_on: print("New value for pxl is " + str(controller.pxl_ratio))
            self.error_frame.destroy()
            controller.next_step(2)
            controller.show_frame(1)
        except ValueError:
            tk.Label(master=self.error_frame, text="try again with dot, no comma").pack()

    # routine executed when saving
    def make_save_update_from_tab1(self, controller):
        controller.image = self.image
        # read pxl_ratio again
        f = self.entryfield_pxlratio.get()
        try:
            controller.pxl_ratio = float(f)
        except ValueError:
            if self.console_print_on: print("oops, default: " + str(controller.pxl_ratio))
        if self.console_print_on: print(f)

    # update pxl ratio when config-data is loaded for re-evaluation
    def update_pxlratio_tab1(self, pxl_new):
        self.entryfield_pxlratio.delete(0, 'end')
        self.entryfield_pxlratio.insert(0, str(pxl_new))
        if self.console_print_on: print('A new pxl ratio is loaded: ' + str(pxl_new))
```

VIII.6.1.3 Tab 2

```
##########################################
# ###################################### #
# ### Tab 2 - Canny Edge Detection - ### #
# ###################################### #
##########################################

# import packages
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.backends.backend_tkagg import NavigationToolbar2Tk
from matplotlib.figure import Figure
from matplotlib.pyplot import rcParams
import matplotlib.pyplot as plt
import numpy as np
import tkinter as tk
from tkinter import ttk
from skimage import feature, img_as_ubyte
from skimage.filters import rank
from skimage.segmentation import watershed
from skimage.morphology import disk
from skimage.measure import regionprops
from scipy import ndimage as ndi

# global plot parameters
matplotlib.use("TkAgg")
LARGE_FONT = ("Verdana", 12)
GlobFontSize = 10
params = {
    'axes.labelsize': GlobFontSize,
    'font.size': GlobFontSize,
    'axes.titlesize': GlobFontSize,
    'font.family': "Calibri",
    'legend.fontsize': GlobFontSize,
    'xtick.labelsize': GlobFontSize,
    'ytick.labelsize': GlobFontSize,
    'text.usetex': False,
    'figure.constrained_layout.use': True,
    'lines.markersize': 3.0,
    'boxplot.flierprops.markersize': 3
}
rcParams.update(params)


# tab 2 code
class Page_Area_detection(tk.Frame):
    def __init__(self, parent, controller):
        # get variables from controller
        self.console_print_on = controller.console_print_on
        self.particles_at_border = controller.particles_at_border
        self.image = controller.image
        self.image_loaded = controller.image_rgb
        self.img_x = controller.img_x
        self.img_y = controller.img_y
        self.sigm = controller.sigma
        self.high_threshold = controller.high_threshold
        self.low_threshold = controller.low_threshold
        self.labelLines_parameter = controller.labelLines_parameter
        self.image_w_axes = controller.image_w_axes.get()
        # imagesize is (height, width)
        self.new_imagesize = controller.new_imagesize

        ## create tab layout ##
        tk.Frame.__init__(self, parent)
        # devide window into 2 frames
        tab2_left = ttk.Frame(master=self)
        tab2_left.grid(row=0, column=0, sticky="nw")
        tab2_left.columnconfigure(0, minsize=220)
        if self.console_print_on: print("show particles at border: ")
        if self.console_print_on: print(controller.particles_at_border)
        self.tab2_right = ttk.Frame(master=self)
        self.tab2_right.grid(row=0, column=1, sticky="nsew")
        self.tab2_right.update()
        if self.console_print_on: print("size of tab2_right")
        if self.console_print_on: print(self.tab2_right.winfo_width())

        # make tab 2 menu on left container
        # # Different variables for tab menu
        sc_sigm = tk.DoubleVar()
        sc_b = tk.DoubleVar()
        sc_c = tk.DoubleVar()
        sc_param = tk.DoubleVar()
        self.checkbx1_var = tk.BooleanVar()
        self.checkbx2_var = tk.BooleanVar()
        self.checkbx2_var.set(False)

        # # create checkbuttons
        ttk.Checkbutton(tab2_left, text="Simultaneous Plotting", variable=self.checkbx1_var,
                command=self.checkbx1).grid(row=0, column=0, sticky="ew")
        ttk.Checkbutton(tab2_left, text="Add Particles at Borders", variable=self.checkbx2_var,
                command=self.checkbx2).grid(row=1, column=0, sticky="ew")
        if self.console_print_on: print("Particles at borders used? "
                + str(self.checkbx2_var.get()))
        # # variable sigma
        ttk.Label(tab2_left, text="Sigma").grid(row=2, column=0, sticky="ew")
        self.scale_sigma = tk.Scale(tab2_left, variable=sc_sigm, from_=0.00, to=10.00, length=200,
                digits=3, orient=tk.HORIZONTAL, resolution=0.025)
        if self.console_print_on: print("width scale")
        if self.console_print_on: print(self.scale_sigma.cget("width"))
        self.scale_sigma.set(self.sigm)
        self.scale_sigma.grid(row=3, column=0, sticky="ew")
        # # show the grey value histogram on left side
        self.fig_h = Figure(figsize=(2, 2))
        hist_plot = self.fig_h.add_subplot(111)
        hist_plot.clear()
        hist_plot.hist(self.image.ravel(), 256, [0, 256])
        canvas2_h1 = FigureCanvasTkAgg(self.fig_h, tab2_left)
        canvas2_h1.get_tk_widget().grid(row=4, column=0, rowspan=1)
        # # create sliders for threshold
        ttk.Label(tab2_left, text="low_threshold").grid(row=5, column=0, sticky="ew")
        self.scale_b = tk.Scale(tab2_left, variable=sc_b, from_=0.00, to=256.0, length=200,
                digits=2, orient=tk.HORIZONTAL, resolution=0.05)
        self.scale_b.set(self.low_threshold)
        self.scale_b.grid(row=6, column=0, sticky="ew")
        ttk.Label(tab2_left, text="high_threshold").grid(row=7, column=0, sticky="ew")
        self.scale_c = tk.Scale(tab2_left, variable=sc_c, from_=0.00, to=256.0, length=200,
                digits=2, orient=tk.HORIZONTAL, resolution=0.05)
        self.scale_c.set(self.high_threshold)
        self.scale_c.grid(row=8, column=0, sticky="ew")
        # # create vertical lines on the histogram to adjust for edge detection
        self.lower_limit_line = hist_plot.axvline(self.scale_b.get(), color='k')
        self.upper_limit_line = hist_plot.axvline(self.scale_c.get(), color='k')
        self.scale_b.config(command=self.update_histo)
        self.scale_c.config(command=self.update_histo)
        # # button for edge detection
        butt2_1 = ttk.Button(tab2_left, text="> Edge Detection",
                command=self.scroll_canny_detection)
        butt2_1.grid(row=9, column=0, sticky="ew")
        # # create slider for line parameter
        ttk.Label(tab2_left, text="Labels - Line Parameter").grid(row=11, column=0, sticky="ew",
                pady=(10, 0))
        self.scale_param = tk.Scale(tab2_left, variable=sc_param, from_=0.00, to=10.00, digits=1,
                orient=tk.HORIZONTAL, resolution=1)
        self.scale_param.set(self.labelLines_parameter)
        self.scale_param.config(state="disabled")
        self.scale_param.grid(row=12, column=0, sticky="ew")
        # # button for label image
        self.butt2_3 = ttk.Button(tab2_left, text="> Labels", command=self.plot_markers,
                state="disabled")
        self.butt2_3.grid(row=13, column=0, sticky="ew")
        # # button for watershed
        self.butt2_4 = ttk.Button(tab2_left, text="> Watershed Separation",
                command=self.watershedsep, state="disabled")
        self.butt2_4.grid(row=14, column=0, sticky="ew", pady=(10, 0))
        ttk.Separator(tab2_left, orient='horizontal').grid(row=15,column=0, pady=10, sticky="ew")
```

```
    # # button for perform all steps at once
    butt2_6 = ttk.Button(tab2_left, text="DO ALL", command=self.all_do)
    butt2_6.grid(row=16, column=0, sticky="ew")
    ttk.Separator(tab2_left, orient='horizontal').grid(row=17,column=0, pady=10, sticky="ew")
    # # button for next tab
    self.butt2_5 = ttk.Button(tab2_left, text=">> Go to Evaluation",
            command=lambda: self.update_controller_from_tab2(controller), state="disabled")
    self.butt2_5.grid(row=18, column=0, sticky="ew")

    tab2_left.update()
    # variables for updating tab during replotting images
    self.canny_drawn = 0
    self.marker_drawn = 0
    self.watershed_drawn = 0

# Update the position of the vertical lines in the histogram on left
def update_histo(self, value):
    self.lower_limit_line.set_xdata([self.scale_b.get(), self.scale_b.get()])
    self.upper_limit_line.set_xdata([self.scale_c.get(), self.scale_c.get()])
    if self.console_print_on: print("updated histo")
    self.fig_h.canvas.draw_idle()

# perform all steps at once
def all_do(self):
    self.scroll_canny_detection()
    self.plot_markers()
    self.watershedsep()

# in checkbox 1 you can say if you want interactive plotting
def checkbx1(self):
    if self.checkbx1_var.get() is True:
        if self.console_print_on: print("Interactive mode: ON")
        self.scale_sigma.config(command=self.simultane_canny)
        self.scale_b.config(command=self.simultane_canny)
        self.scale_c.config(command=self.simultane_canny)
        self.scale_param.config(command=self.simultane_plot_markers)
    else:
        if self.console_print_on: print("Interactive mode: OFF")
        self.scale_sigma.config(command=self.no_action)
        self.scale_b.config(command=self.update_histo)
        self.scale_c.config(command=self.update_histo)
        self.scale_param.config(command=self.no_action)

# if checkbox 1 is on, canny image is plotted when slider value is changed
def simultane_canny(self, a):
    self.update_histo(a)
    self.plot_Canny(a)
    if not self.marker_drawn == 0:
        self.plot_markers()
        if not self.watershed_drawn == 0:
            self.watershedsep()

# if checkbox 1 is on, marker image is plotted when slider value is changed
def simultane_plot_markers(self, value):
    self.plot_markers()
    if not self.watershed_drawn == 0:
        self.watershedsep()

# checkbox 2 let you choose if you want to take bordered particles into account
def checkbx2(self):
    if self.checkbx2_var.get() is True:
        self.particles_at_border = True
        if self.console_print_on: print("Particles at border: ON")
    else:
        self.particles_at_border = False
        if self.console_print_on: print("Particles at border: OFF")

# method needed as the sliders command cannot be turned off elseway
def no_action(self, value):
    unused_var = 1
    if self.console_print_on: print(value)
```

```
# method needed as the sliders will hand over the slider value; and canny does not need one
def plot_Canny(self, value):
    self.scroll_canny_detection()
    if self.console_print_on: print("Current slider value:")
    if self.console_print_on: print(value)

# detect borders of particles/areas by using canny of skimage package
def scroll_canny_detection(self):
    # get slider variables
    self.sigm = float(str(self.scale_sigma.get()))
    self.low_threshold = float(str(self.scale_b.get()))
    self.high_threshold = float(str(self.scale_c.get()))
    # make canny
    self.edges = feature.canny(self.image, sigma=self.sigm, low_threshold=self.low_threshold,
            high_threshold=self.high_threshold)
    # create images with the detected particle/area borders
    # check if images are already drawn, create if not, update if yes
    if self.canny_drawn == 0:
        # show the canny image
        self.fig_canny = Figure(figsize=(self.img_x, self.img_y))
        canny_plot = self.fig_canny.add_subplot(111)
        canny_plot.clear()
        self.ca_plot = canny_plot.imshow(self.edges, cmap="gray", interpolation="nearest")
        if self.image_w_axes:
            canny_plot.axis("on")
            canny_plot.set_title("Canny")
            canny_plot.set_xlabel("Horizontal Pixel ID")
            canny_plot.set_ylabel("Vertical Pixel ID")
        else:
            canny_plot.axis("off")
        canvas2_1 = FigureCanvasTkAgg(self.fig_canny, self.tab2_right)
        canvas2_1.get_tk_widget().grid(row=0, column=0, rowspan=1)
        # toolbar
        toolbar = NavigationToolbar2Tk(canvas2_1, self.tab2_right, pack_toolbar=False)
        toolbar.update()
        toolbar.grid(row=1, column=0, rowspan=1)

        # show canny on grey scale image
        self.fig2_2 = Figure(figsize=(self.img_x, self.img_y))
        d = self.fig2_2.add_subplot(111)
        self.ca_on_img = d.imshow(self.edges, cmap=plt.cm.hsv)
        d.imshow(self.image, cmap=plt.cm.gray, alpha=0.5)
        if self.image_w_axes:
            d.axis("on")
            d.set_title("Canny on Original")
            d.set_xlabel("Horizontal Pixel ID")
            d.set_ylabel("Vertical Pixel ID")
        else:
            d.axis("off")
        canvas2_2 = FigureCanvasTkAgg(self.fig2_2, self.tab2_right)
        canvas2_2.get_tk_widget().grid(row=0, column=1, rowspan=1)
        # toolbar
        toolbar2 = NavigationToolbar2Tk(canvas2_2, self.tab2_right, pack_toolbar=False)
        toolbar2.update()
        toolbar2.grid(row=1, column=1, rowspan=1)

        self.canny_drawn = 1
        self.fig_canny.canvas.draw_idle()
        self.fig2_2.canvas.draw_idle()

    else:
        # update canny and on grey scale image when self.edges altered
        self.ca_plot.set_data(self.edges)
        self.ca_on_img.set_data(self.edges)
        self.fig_canny.canvas.draw_idle()
        self.fig2_2.canvas.draw_idle()
        if self.console_print_on: print("canny images updated")

    self.tab2_right.update()
    if self.console_print_on: print("size of tab2_right")
    if self.console_print_on: print(self.tab2_right.winfo_width())
    if self.console_print_on: print("Canny Image Plotted")
```

```
        # enable next step
        self.butt2_3.config(state="normal")
        self.scale_param.config(state="normal")

    # create marker image from slider value for different areas
    def plot_markers(self):
        # global gradient via rank
        self.labelLines_parameter = float(str(self.scale_param.get()))
        self.gradient = rank.gradient(img_as_ubyte(self.edges),
                disk(self.labelLines_parameter)) < 10
        # global markers
        self.markers = ndi.label(self.gradient)[0]
        if self.console_print_on: print(ndi.label(self.gradient)[0])
        newcmp = 'YlOrRd'
        # check if image is already drawn, create if not, update if yes
        if self.marker_drawn == 0:
            # show the marker image 1st time
            self.fig2_3 = Figure(figsize=(self.img_x, self.img_y))
            marker_img = self.fig2_3.add_subplot(111)
            self.marker_img_update = marker_img.imshow(self.markers, cmap=newcmp)
            if self.image_w_axes:
                marker_img.set_title("Markers")
                marker_img.set_xlabel("Horizontal Pixel ID")
                marker_img.set_ylabel("Vertical Pixel ID")
            canvas2_3 = FigureCanvasTkAgg(self.fig2_3, self.tab2_right)
            canvas2_3.get_tk_widget().grid(row=2, column=0, rowspan=1)
            # toolbar
            toolbar2_3 = NavigationToolbar2Tk(canvas2_3, self.tab2_right, pack_toolbar=False)
            toolbar2_3.update()
            toolbar2_3.grid(row=3, column=0, rowspan=1)
            if self.console_print_on: print("Marker Image Plotted")
            self.marker_drawn = 1

        else:
            # update marker image and displayed label color when scale_param altered
            self.marker_img_update.set_data(self.markers)
            self.marker_img_update.set_clim(vmax=np.unique(self.markers).size)
            self.fig2_3.canvas.draw_idle()
            if self.console_print_on: print("marker image updated")
        # enable next step
        self.butt2_4.config(state="normal")

    # perform watershed separation
    def watershedsep(self):
        self.labels = watershed(self.edges, self.markers)
        # check if image is already drawn, create if not, update if yes
        if self.watershed_drawn == 0:
            # show the watershed image
            self.fig2_4 = Figure(figsize=(self.img_x, self.img_y))
            a = self.fig2_4.add_subplot(111)
            self.watershed_img_update = a.imshow(self.labels, cmap='prism')
            if self.image_w_axes:
                a.set_title("Watershed Labels")
                a.set_xlabel("Horizontal Pixel ID")
                a.set_ylabel("Vertical Pixel ID")
            canvas2_4 = FigureCanvasTkAgg(self.fig2_4, self.tab2_right)
            canvas2_4.get_tk_widget().grid(row=2, column=1, rowspan=1)
            # toolbar
            toolbar2_4 = NavigationToolbar2Tk(canvas2_4, self.tab2_right, pack_toolbar=False)
            toolbar2_4.update()
            toolbar2_4.grid(row=3, column=1, rowspan=1)
            if self.console_print_on: print("Watershed Separation Plotted")
            self.watershed_drawn = 1
        else:
            self.watershed_img_update.set_data(self.labels)
            self.watershed_img_update.set_clim(vmax=np.unique(self.labels).size)
            self.fig2_4.canvas.draw_idle()
            if self.console_print_on: print("watershed image updated")
        # enable next step
        self.butt2_5.config(state="normal")
```

```
# read out of properties of the detected areas
def create_ParticleProp_list(self):
    self.ParticleProp = regionprops(self.labels)
    # new ParticleProp_List if no border particles should be taken into account,
    # else unaltered ParticleProp_List is used
    if self.particles_at_border is False:
        if self.console_print_on: print("Do remove areas/particles at borders")
        new_ParticleProp = []
        for P in self.ParticleProp:
            # self.new_imagesize[0] == height ; self.new_imagesize[1] == width
            # Bounding box (min_row, min_col, max_row = height, max_col = width)
            if P.bbox[0] > 2 and P.bbox[1] > 2 and P.bbox[2] < (self.new_imagesize[0] - 2)
                    and P.bbox[3] < (self.new_imagesize[1] - 2):
                new_ParticleProp.append(P)
        self.ParticleProp = new_ParticleProp

    # clean particle prop from minor and major axis == 0
    new_list = []
    zero_list = []
    for P in self.ParticleProp:
        if not P.major_axis_length == 0 and not P.minor_axis_length == 0:
            new_list.append(P)
        else:
            zero_list.append(P)
    self.ParticleProp = new_list

    if self.console_print_on:
        for item in zero_list:
            print('Particle with zero dimension: No. {0} with major {1} and minor {2}'
                    .format(item.label, item.major_axis_length, item.minor_axis_length))

# update images and then handover the altered parameters to the controller,
# so they can be used with next tab
def update_controller_from_tab2(self, controller):
    self.all_do()
    self.create_ParticleProp_list()
    # variables/pictures/list which are made/altered by window
    controller.edges = self.edges        # edges of areas
    controller.gradient = self.gradient  # gradient used
    controller.markers = self.markers    # markers of areas
    controller.labels = self.labels      # image of separeted areas
    controller.ParticleProp = self.ParticleProp  # region properties of separated areas
    # checkbox variable
    controller.particles_at_border = self.particles_at_border
    # current values of sliders
    controller.sigma = self.sigm
    controller.high_threshold = self.high_threshold
    controller.low_threshold = self.low_threshold
    controller.labelLines_parameter = self.labelLines_parameter
    # go to next tab 3
    controller.next_step(3)
    controller.show_frame(2)

# handover when saving the parameters
def make_save_update_from_tab2(self, controller):
    # checkbox variable
    controller.particles_at_border = self.particles_at_border
    # current values of sliders
    controller.sigma = self.sigm
    controller.high_threshold = self.high_threshold
    controller.low_threshold = self.low_threshold
    controller.labelLines_parameter = self.labelLines_parameter
```

VIII.6.1.4 Tab 3

```
##############################################
# ########################################## #
# ### Tab 3 - Ellipse areas selection - ### #
# ########################################## #
##############################################

# import libraries
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.backends.backend_tkagg import NavigationToolbar2Tk
from matplotlib.figure import Figure
from matplotlib.pyplot import rcParams
import matplotlib.pyplot as plt
import matplotlib.colors as colors
import numpy as np
import tkinter as tk
from tkinter import ttk
from skimage.draw import ellipse
import math
from PIL import Image, ImageDraw, ImageFont

# global plot parameters
matplotlib.use("TkAgg")
LARGE_FONT = ("Verdana", 12)
GlobFontSize = 10
params = {
    'axes.labelsize': GlobFontSize,
    'font.size': GlobFontSize,
    'axes.titlesize': GlobFontSize,
    'font.family': "Calibri",
    'legend.fontsize': GlobFontSize,
    'xtick.labelsize': GlobFontSize,
    'ytick.labelsize': GlobFontSize,
    'text.usetex': False,
    'figure.constrained_layout.use': True,
    'lines.markersize': 3.0,
    'boxplot.flierprops.markersize': 3
}
rcParams.update(params)


# tab 3 code
class Page_Area_Selection(tk.Frame):
    def __init__(self, parent, controller):
        # get dummys and variables from controller
        self.console_print_on = controller.console_print_on
        self.pxl_ratio = controller.pxl_ratio
        self.ParticleProp = controller.ParticleProp
        self.new_imagesize = controller.new_imagesize  # imagesize is (height, width)
        self.particles_at_border = controller.particles_at_border
        self.ParticleList = controller.ParticleList
        self.ellipse_draw_list = controller.ellipse_draw_list
        self.image_ellipses = controller.image_ellipses
        self.image_numbers = controller.image_numbers
        self.image = controller.image
        self.image_w_axes = controller.image_w_axes.get()
        self.img_x = controller.img_x
        self.img_y = controller.img_y

        # start variables of the tab for the entryBoxes
        self.highest_area = 1000000
        self.minimum_area = 0
        self.highest_major = 10000
        self.minimum_major = 0
        self.highest_minor = 10000
        self.minimum_minor = 0
        self.highest_aspect = 100
        self.minimum_aspect = 0
        self.highest_orientation = 180
        self.minimum_orientation = 0
```

```
# other variables
self.ellipse_color = plt.cm.hsv
self.area_factor = self.pxl_ratio * self.pxl_ratio / 1  # in nm^2
self.reset_ParticleProp_list = None
self.ellpsoids_at_tab3_drawn = 0
self.max_aspect = 0.0  # for polarplot

## create tab layout ##
tk.Frame.__init__(self, parent)
# devide window into 2 frames
self.tab3_left = tk.Frame(master=self)
self.tab3_left.grid(row=1, column=0, sticky="nw")
self.tab3_left.columnconfigure(0, minsize=220)
self.tab3_right = tk.Frame(master=self)
self.tab3_right.grid(row=1, column=1, sticky="ne")

# make tab 3 menu on left container
# checkbutton on left, which value to select
self.rad_but_var = tk.IntVar()
self.R1 = tk.Radiobutton(self.tab3_left, text="Area", variable=self.rad_but_var, value=1,
        command=self.radiobut_chosen)
self.R1.grid(row=0, column=0, sticky="nw")
self.R2 = tk.Radiobutton(self.tab3_left, text="Major", variable=self.rad_but_var, value=2,
        command=self.radiobut_chosen)
self.R2.grid(row=1, column=0, sticky="nw")
self.R3 = tk.Radiobutton(self.tab3_left, text="Minor", variable=self.rad_but_var, value=3,
        command=self.radiobut_chosen)  # lambda:butt3_1.config(state = "normal")
self.R3.grid(row=2, column=0, sticky="nw")
self.R4 = tk.Radiobutton(self.tab3_left, text="Aspect", variable=self.rad_but_var, value=4,
        command=self.radiobut_chosen)  # lambda:butt3_1.config(state = "normal")
self.R4.grid(row=3, column=0, sticky="nw")
self.R5 = tk.Radiobutton(self.tab3_left, text="Orientation", variable=self.rad_but_var,
        value=5, command=self.radiobut_chosen)  # lambda:butt3_1.config(state = "normal")
self.R5.grid(row=4, column=0, sticky="nw")
# button to refresh the histogram with adjusted values
self.butt3_1 = ttk.Button(self.tab3_left, text="Refresh Histogram", command=self.make_hist)
self.butt3_1["state"] = "disable"
self.butt3_1.grid(row=7, column=0, sticky='ew')
# button to plot the ellipses defined by data in histogram
self.butt3_2 = ttk.Button(self.tab3_left, text="> Plot Ellipses from Histogram",
        command=self.create_ellipses_image)
self.butt3_2["state"] = "disable"
self.butt3_2.grid(row=8, column=0, sticky='ew')
ttk.Separator(self.tab3_left, orient='horizontal').grid(row=9, column=0, pady=10,
        sticky="ew")
# button for crop the underlying dataspace
self.butt3_3 = ttk.Button(self.tab3_left, text="Redefine Data Borders",
        command=self.data_list_further)
self.butt3_3["state"] = "disable"
self.butt3_3.grid(row=10, column=0, sticky='ew')
# button for reset dataspace
self.butt3_4 = ttk.Button(self.tab3_left, text="Reset Data Borders",
        command=self.reset_data_list)
self.butt3_4["state"] = "disable"
self.butt3_4.grid(row=11, column=0, sticky='ew')
ttk.Separator(self.tab3_left, orient='horizontal').grid(row=12, column=0, pady=10,
        sticky="ew")
# button for next tab
self.butt3_5 = ttk.Button(self.tab3_left, text=">> Final Analysis",
        command=lambda: self.update_controller_from_tab3(controller))
self.butt3_5["state"] = "disable"
self.butt3_5.grid(row=13, column=0, sticky='ew')
ttk.Separator(master=self.tab3_left, orient='horizontal').grid(row=14, column=0, pady=10,
        sticky="ew")
# variables field to insert directl filter variables
tk.Label(master=self.tab3_left, text='Filter Values').grid(row=15, column=0, sticky='ew')
self.left_preset_box = tk.Frame(master=self.tab3_left, width=120)
self.left_preset_box.grid(row=16, column=0, pady=10, sticky="ew")
self.filterdata = []
labels_filter = ['Area', 'Major', 'Minor', 'Aspect', 'Orientation']
for i in range(0, len(labels_filter)):
    tk.Label(self.left_preset_box, text=labels_filter[i]).grid(row=i, column=0)
self.entryBox = []
```

```
    for i in range(0, 10):
        self.entryBox.append(tk.Entry(master=self.left_preset_box, width=10))
        self.entryBox[i].grid(row=i // 2, column=1 + i % 2)
    self.entryBox[0].insert(0, str(self.minimum_area))
    self.entryBox[1].insert(0, str(self.highest_area))
    self.entryBox[2].insert(0, str(self.minimum_major))
    self.entryBox[3].insert(0, str(self.highest_major))
    self.entryBox[4].insert(0, str(self.minimum_minor))
    self.entryBox[5].insert(0, str(self.highest_minor))
    self.entryBox[6].insert(0, str(self.minimum_aspect))
    self.entryBox[7].insert(0, str(self.highest_aspect))
    self.entryBox[8].insert(0, str(self.minimum_orientation))
    self.entryBox[9].insert(0, str(self.highest_orientation))
    for i in range(0, 10):
        self.filterdata.append(float(self.entryBox[i].get()))
    # button to use the values set in the entryBoxes
    self.butt3_6 = ttk.Button(self.tab3_left, text="Use Filters",
            command=self.create_filtered_particle_list)
    self.butt3_6["state"] = "normal"
    self.butt3_6.grid(row=17, column=0, sticky='ew')
    # button to get the values from the scales into the entryBoxes
    self.butt3_7 = ttk.Button(self.tab3_left, text="Update from Scales",
            command=self.get_filterdata_from_scale)
    self.butt3_7["state"] = "normal"
    self.butt3_7.grid(row=18, column=0, sticky='ew')

    self.tab3_left.update()

# displays the histogram to the chosen variable
def radiobut_chosen(self):
    self.butt3_1.config(state="normal")
    self.scale_for_adjust()
    self.make_hist()
    if self.console_print_on: print("size of tab3_left")
    if self.console_print_on: print(self.tab3_left.winfo_width())

# create scales for adjusting the histogram
def scale_for_adjust(self):
    if self.console_print_on: print("A scaling of histogram was done.")
    # set scale boundaries according to choice
    if self.rad_but_var.get() == 1:
        self.scale_minimum = min([P.area for P in self.ParticleProp]) * self.area_factor * 0.99
        self.scale_highest = max([P.area for P in self.ParticleProp]) * self.area_factor * 1.01
    elif self.rad_but_var.get() == 2:
        self.scale_minimum = min([P.major_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 0.99
        self.scale_highest = max([P.major_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 1.01
    elif self.rad_but_var.get() == 3:
        self.scale_minimum = min([P.minor_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 0.99
        self.scale_highest = max([P.minor_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 1.01
    elif self.rad_but_var.get() == 4:
        try:
            self.scale_minimum = min([P.major_axis_length / P.minor_axis_length for P in
                    self.ParticleProp]) * 10 * 0.99
            self.scale_highest = max([P.major_axis_length / P.minor_axis_length for P in
                    self.ParticleProp]) * 10 * 1.01
        except ZeroDivisionError:
            self.scale_minimum = 1
            self.scale_highest = 100  # not higher as aspect 10 possible
            if self.console_print_on: print("ZeroDevisionError")
    elif self.rad_but_var.get() == 5:
        self.scale_minimum = (min([P.orientation for P in self.ParticleProp]) + math.pi / 2)
            * 180 / math.pi
        self.scale_highest = (max([P.orientation for P in self.ParticleProp]) + math.pi / 2)
            * 180 / math.pi
    else:
        self.scale_minimum = 0
        self.scale_highest = 100

    self.sc_hist_left = tk.DoubleVar()
    self.sc_hist_rght = tk.DoubleVar()

    self.scale_hist_left = tk.Scale(self.tab3_left, variable=self.sc_hist_left,
            from_=self.scale_minimum, to=self.scale_highest, length=200, digits=3,
            orient=tk.HORIZONTAL, resolution=1)
    self.scale_hist_left.set(0)
    self.scale_hist_left.grid(row=5, column=0)

    self.scale_hist_rght = tk.Scale(self.tab3_left, variable=self.sc_hist_rght,
            from_=self.scale_minimum, to=self.scale_highest, length=200, digits=3,
            orient=tk.HORIZONTAL, resolution=1)
    self.scale_hist_rght.set(self.scale_highest)
    self.scale_hist_rght.grid(row=6, column=0)  # , sticky="nw")

    if self.console_print_on: print(self.scale_highest)
    if self.console_print_on: print(self.scale_minimum)

# make the histogram from the variables of the particle/area properties
def make_hist(self):
    self.butt3_2.config(state="normal")
    # count labels
    MaxLabel = max([P.label for P in self.ParticleProp])
    if self.console_print_on: print(self.rad_but_var.get())
    if self.console_print_on: print("max label " + str(MaxLabel))
    self.hist_list = []
    self.area_no = []

    # prepare the histogram
    self.fig_hist2 = Figure(figsize=(self.img_x, self.img_y))
    tab2_hist = self.fig_hist2.add_subplot(111)

    # choose which parameter for picking and write to hist_list
    if self.rad_but_var.get() == 1:
        tab2_hist.set_title("Area")
        tab2_hist.set_xlabel("Area [nm^2]")
        tab2_hist.set_ylabel("Number Count")
        # make list of areas
        self.minimum_area = float(self.scale_hist_left.get())
        self.highest_area = float(self.scale_hist_rght.get())
        for P in self.ParticleProp:
            if P.area * self.area_factor >= self.minimum_area and P.area * self.area_factor
                    <= self.highest_area:
                self.hist_list.append(P.area * self.area_factor)
                self.area_no.append(P.label)

    elif self.rad_but_var.get() == 2:
        tab2_hist.set_title("Major Axis Length")
        tab2_hist.set_xlabel("Major Axis [nm]")
        tab2_hist.set_ylabel("Number Count")
        # make list of major axis
        self.minimum_major = float(self.scale_hist_left.get())
        self.highest_major = float(self.scale_hist_rght.get())
        for P in self.ParticleProp:
            if P.major_axis_length * self.pxl_ratio >= self.minimum_major
                    and P.major_axis_length * self.pxl_ratio <= self.highest_major:
                self.hist_list.append(P.major_axis_length * self.pxl_ratio)
                self.area_no.append(P.label)

    elif self.rad_but_var.get() == 3:
        tab2_hist.set_title("Minor Axis Length")
        tab2_hist.set_xlabel("Minor Axis [nm]")
        tab2_hist.set_ylabel("Number Count")
        # make list of minor axis
        self.minimum_minor = float(self.scale_hist_left.get())
        self.highest_minor = float(self.scale_hist_rght.get())
        for P in self.ParticleProp:
            if P.minor_axis_length * self.pxl_ratio >= self.minimum_minor
                    and P.minor_axis_length * self.pxl_ratio <= self.highest_minor:
                self.hist_list.append(P.minor_axis_length * self.pxl_ratio)
                self.area_no.append(P.label)
    elif self.rad_but_var.get() == 4:
        tab2_hist.set_title("Aspect Ratio")
        tab2_hist.set_xlabel("Aspect Ratio [/10]")
        tab2_hist.set_ylabel("Number Count")
        # make list of aspects
        self.minimum_aspect = float(self.scale_hist_left.get())
        self.highest_aspect = float(self.scale_hist_rght.get())
        for P in self.ParticleProp:
            if (P.major_axis_length / P.minor_axis_length * 10) >= self.minimum_aspect
                    and (P.major_axis_length / P.minor_axis_length * 10)
                    <= self.highest_aspect:
                self.hist_list.append(P.major_axis_length / P.minor_axis_length * 10)
                self.area_no.append(P.label)

    elif self.rad_but_var.get() == 5:
        tab2_hist.set_title("Orientation")
        tab2_hist.set_xlabel("Angle to x-Axis [°]")
        tab2_hist.set_ylabel("Number Count")
        # make list of orientations
        self.minimum_orientation = float(self.scale_hist_left.get())
        self.highest_orientation = float(self.scale_hist_rght.get())
        for P in self.ParticleProp:
            orientation_in_deg = (float(P.orientation) + math.pi / 2) * 180 / math.pi
            if orientation_in_deg >= self.minimum_orientation and orientation_in_deg
                    <= self.highest_orientation:
                self.hist_list.append(orientation_in_deg)
                self.area_no.append(P.label)
    else:
        self.hist_list = []

    # make histogram from hist_list
    tab2_hist.hist(self.hist_list)
    canvas3_1 = FigureCanvasTkAgg(self.fig_hist2, self.tab3_right)
    canvas3_1.get_tk_widget().grid(row=0, column=0, rowspan=1)
    # Create vertical lines on the histogram to adjust
    self.lower_limit_line = tab2_hist.axvline(self.scale_hist_left.get(), color='k')
    self.upper_limit_line = tab2_hist.axvline(self.scale_hist_rght.get(), color='k')
    self.scale_hist_left.config(command=self.update_histo_tab3)
    self.scale_hist_rght.config(command=self.update_histo_tab3)
    # toolbar
    toolbar3_1 = NavigationToolbar2Tk(canvas3_1, self.tab3_right, pack_toolbar=False)
    toolbar3_1.update()
    toolbar3_1.grid(row=1, column=0, rowspan=1)

# when adjust sliders and click button, update histogram
def update_histo_tab3(self, value):
    # Update the position of the vertical lines
    self.lower_limit_line.set_xdata([self.scale_hist_left.get(), self.scale_hist_left.get()])
    self.upper_limit_line.set_xdata([self.scale_hist_rght.get(), self.scale_hist_rght.get()])
    if self.console_print_on: print("updated hist tab3")
    self.fig_hist2.canvas.draw_idle()

# make the ellipse image using the region properties
def create_ellipses_image(self):
    self.butt3_3.config(state="normal")
    self.butt3_5.config(state="normal")
    self.ParticleList = []
    self.ParticleList.append('Partikel_No., posy, posx, minor, major, orientation_in_deg,
            ratio, area')
    self.ellipse_draw_list = []

    # make image of ellipses on blank image
    # np.zero (blank image) needs dimension of height, width
    # --> using new_imagesize in normal order
    image_zero = np.zeros(shape=[self.new_imagesize[0], self.new_imagesize[1], 3])
    image_zero.fill(0)
    if self.console_print_on: print(image_zero.shape)
    if self.console_print_on: print(self.new_imagesize)

    number = 1
    for P in self.ParticleProp:
        if P.label in self.area_no:
            # get all necessary labled region informations
            major = int(round(P.major_axis_length / 2, 0))
            minor = int(round(P.minor_axis_length / 2, 0))
            posx = int(P.centroid[1])
            posy = int(P.centroid[0])

            # convert orientation to draw ellipses orientation
            # -- since skimage v 0.18 necessary to do pi/2
            orientation = float(P.orientation) + math.pi / 2

            # draw ellipses into blank image at given position in color
            # and do not draw outside of imagesize
            rr, cc = ellipse(posy, posx, minor, major, shape=self.new_imagesize,
                    rotation=orientation)
            try:
                image_zero[rr, cc] = colors.to_rgb(self.ellipse_color(abs((orientation)
                    / (math.pi))))
            except IndexError:
                print("one index error with " + str(posy) + "  " + str(posx) + "  "
                    + str(minor) + "  " + str(major))
                print(rr)  # rr is y value
                print(cc)  # cc is x value
            org_no = P.label

            # write drawn ellipse data in ellipse_draw_list using image coordinates
            elementi = [number, posy, posx, minor, major, orientation, org_no, cc, rr]
            self.ellipse_draw_list.append(elementi)
            # write properties of drawn data in ParticleList for saving/export
            aspect_ratio = round(P.major_axis_length / P.minor_axis_length, 3)
            ellipse_area = round(math.pi * minor * self.pxl_ratio * major * self.pxl_ratio, 3)
            element = [number, posy, posx, P.minor_axis_length * self.pxl_ratio,
                    P.major_axis_length * self.pxl_ratio, orientation * 180 / math.pi,
                    aspect_ratio, ellipse_area]
            self.ParticleList.append(element)
            # get highest aspect ratio for polarplot
            if aspect_ratio > self.max_aspect:
                self.max_aspect = aspect_ratio
            number = number + 1
        else:
            if self.console_print_on: print("one data excluded")

    # after drawing every ellipse, safe the image as image_ellipses and number the regions
    self.image_ellipses = image_zero
    self.image_numbers = self.number_region_on_image(self.image_ellipses)
    self.image_ellipses = image_zero

    self.image_ellipses_at_tab3()
    self.make_polarplot()

# add numbers on the ellipses image to mark the regions
def number_region_on_image(self, numparray):
    # get an image
    base = Image.fromarray(numparray, mode='RGBA')
    # turn array of new_imagesize
    size = str(self.new_imagesize[1]) + "," + str(self.new_imagesize[0])
    if self.console_print_on: print(base.size)
    if self.console_print_on: print(self.new_imagesize)
    if self.console_print_on: print(size)
    if self.console_print_on: print("here")
    # make a blank image for the text, initialized to transparent text color
    txt = Image.new('RGBA', base.size, (255, 255, 255, 0))
    # get a font
    fnt = ImageFont.truetype('/usr/share/fonts/TTF/arial.ttf', 20)
    # get a drawing context
    d = ImageDraw.Draw(txt)
    # draw text, full opacity - ((x,y),text,font, fill)
    for i in range(len(self.ParticleList) - 1):
        d.text((self.ParticleList[i + 1][2], self.ParticleList[i + 1][1]), str(i + 1),
            font=fnt, fill=(255, 255, 255, 255))
    return np.array(txt)
```

```
# draw the ellipses on the greyscale image
def image_ellipses_at_tab3(self):
    # check if image is already drawn, create if not, update if yes
    if self.ellpsoids_at_tab3_drawn == 0:
        #image ellipses on original image
        self.figd = Figure(figsize=(self.img_x, self.img_y))
        d = self.figd.add_subplot(111)
        self.ellipses_on_img_tab3 = d.imshow(self.image_ellipses, cmap=plt.cm.nipy_spectral)
        d.imshow(self.image, cmap=plt.cm.gray, alpha=0.5)
        self.ellipse_no_img_tab3 = d.imshow(self.image_numbers, cmap=plt.cm.gray, alpha=1)
        if self.image_w_axes:
            d.set_title("Ellipses on Original Image")
            d.set_xlabel("Horizontal Pixel ID")
            d.set_ylabel("Vertical Pixel ID")
        else:
            d.axis("off")
        canvas3_2 = FigureCanvasTkAgg(self.figd, self.tab3_right)
        canvas3_2.get_tk_widget().grid(row=0, column=1, rowspan=1)
        # toolbar
        toolbar3_2 = NavigationToolbar2Tk(canvas3_2, self.tab3_right, pack_toolbar=False)
        toolbar3_2.update()
        toolbar3_2.grid(row=1, column=1, rowspan=1)
        self.ellpsoids_at_tab3_drawn = 1
    else:
        # update image
        self.ellipses_on_img_tab3.set_data(self.image_ellipses)
        self.ellipse_no_img_tab3.set_data(self.image_numbers)
        self.figd.canvas.draw_idle()
        if self.console_print_on: print("ellipses on tab3 updated")

# create the polarplot
def make_polarplot(self):
    figc = Figure(figsize=(self.img_x, self.img_y))
    angle = []
    yachse = []
    c = figc.add_subplot(111, projection='polar')
    for i in range(len(self.ParticleList) - 1):
        # orientation in polar coordinates as befor in ellipse_draw_list
        angle.append((self.ParticleList[i + 1][5]) / 180 * math.pi)
        yachse.append(float(self.ParticleList[i + 1][6]))  # aspect ratio
        # mirroring the values to 90°-270° as unidentifiable if 0° or 180°
        angle.append(((self.ParticleList[i + 1][5]) + 180) / 180 * math.pi)
        yachse.append(float(self.ParticleList[i + 1][6]))
    c.set_rmax(self.max_aspect)
    c.plot(angle, yachse, 'bo')
    c.grid(True)
    canvas3_3 = FigureCanvasTkAgg(figc, self.tab3_right)
    canvas3_3.get_tk_widget().grid(row=2, column=0, rowspan=1)
    # toolbar, although zoom does not work with polar plot
    toolbar3_3 = NavigationToolbar2Tk(canvas3_3, self.tab3_right, pack_toolbar=False)
    toolbar3_3.update()
    toolbar3_3.grid(row=3, column=0, rowspan=1)
    self.make_angle_hist()

# create the angle histogram
def make_angle_hist(self):
    figE = Figure(figsize=(self.img_x, self.img_y))
    e = figE.add_subplot(111)
    angle = []
    for i in range(len(self.ParticleList) - 1):
        angle.append(self.ParticleList[i + 1][5])
    e.hist(angle, bins=36)  # , rwidth = 5
    if self.image_w_axes:
        e.set_title("Orientation of Ellipses")
        e.set_xlabel("Angle [°]")
        e.set_ylabel("Number Count")
    canvas3_4 = FigureCanvasTkAgg(figE, self.tab3_right)
    canvas3_4.get_tk_widget().grid(row=2, column=1, rowspan=1)
    # toolbar
    toolbar3_4 = NavigationToolbar2Tk(canvas3_4, self.tab3_right, pack_toolbar=False)
    toolbar3_4.update()
    toolbar3_4.grid(row=3, column=1, rowspan=1)
```

```
# crop the datalist when hit button 'Redefine Data Borders' or 'Use Filters'
def data_list_further(self):
    self.second_datalist = []
    if self.reset_ParticleProp_list is None:
        if self.console_print_on: print(locals())
        self.reset_ParticleProp_list = []
        for P in self.ParticleProp:
            self.reset_ParticleProp_list.append(P)
        self.data_excluded = 0
    for P in self.ParticleProp:
        if P.label in self.area_no:
            self.second_datalist.append(P)
        else:
            self.data_excluded = self.data_excluded + 1
    if self.console_print_on: print("ParticleProp: " + str(len(self.ParticleProp)))
    if self.console_print_on: print("reset List: " + str(len(self.reset_ParticleProp_list)))
    if self.console_print_on: print("Excluded: " + str(self.data_excluded))
    Length_list2 = len(self.second_datalist)
    if self.console_print_on: print("New ParticleProp: " + str(Length_list2))
    self.ParticleProp = self.second_datalist
    self.butt3_4.config(state="normal")
    self.scale_for_adjust()
    self.make_hist()

# reset the cropped datalist to the list handed over by tab 2
def reset_data_list(self):
    self.ParticleProp = self.reset_ParticleProp_list
    self.butt3_4.config(state="disabled")
    self.scale_for_adjust()
    self.make_hist()

# filter the particle list of tab 2 according to the entryBox values
def create_filtered_particle_list(self):
    self.area_no = []
    self.filterdata = []
    for i in range(0, 10):
        self.filterdata.append(float(self.entryBox[i].get()))
    if self.console_print_on: print("The following filter data was used: "
            + str(self.filterdata))
    for P in self.ParticleProp:
        orientation_in_deg = (float(P.orientation) + math.pi / 2) * 180 / math.pi
        if P.area * self.area_factor >= self.filterdata[0] and P.area * self.area_factor
                    <= self.filterdata[1]:
            if P.major_axis_length * self.pxl_ratio >= self.filterdata[2]
                    and P.major_axis_length * self.pxl_ratio <= self.filterdata[3]:
                if P.minor_axis_length * self.pxl_ratio >= self.filterdata[4]
                    and P.minor_axis_length * self.pxl_ratio <= self.filterdata[5]:
                    if P.major_axis_length / P.minor_axis_length * 10 >= self.filterdata[6]
                            and P.major_axis_length / P.minor_axis_length * 10
                            <= self.filterdata[7]:
                        if orientation_in_deg >= self.filterdata[8] and orientation_in_deg
                                    <= self.filterdata[9]:
                            self.area_no.append(P.label)
    # write filter in list ['area', 'major', 'minor', 'aspect', 'orientation']
    self.minimum_area = self.filterdata[0]
    self.highest_area = self.filterdata[1]
    self.minimum_major = self.filterdata[2]
    self.highest_major = self.filterdata[3]
    self.minimum_minor = self.filterdata[4]
    self.highest_minor = self.filterdata[5]
    self.minimum_aspect = self.filterdata[6]
    self.highest_aspect = self.filterdata[7]
    self.minimum_orientation = self.filterdata[8]
    self.highest_orientation = self.filterdata[9]
    self.rad_but_var.set(1)
    self.butt3_1.config(state="normal")
    self.data_list_further()
    self.create_ellipses_image()

# update the entryBoxes with the current datalist values (rounded to integer)
def get_filterdata_from_scale(self):
    self.update_min_max()
    # write filter in list ['area', 'major', 'minor', 'aspect', 'orientation']
    self.filterdata[0] = self.minimum_area
    self.filterdata[1] = self.highest_area
    self.filterdata[2] = self.minimum_major
    self.filterdata[3] = self.highest_major
    self.filterdata[4] = self.minimum_minor
    self.filterdata[5] = self.highest_minor
    self.filterdata[6] = self.minimum_aspect
    self.filterdata[7] = self.highest_aspect
    self.filterdata[8] = self.minimum_orientation
    self.filterdata[9] = self.highest_orientation
    # update entry fields
    for i in range(0, 10):
        self.entryBox[i].delete(0, 'end')
        self.entryBox[i].insert(0, str(self.filterdata[i]))

# update the variables min/max values of the current particle/area list
def update_min_max(self):
    self.minimum_area = round(min([P.area for P in self.ParticleProp]) * self.area_factor
            * 0.99, 0)
    self.highest_area = round(max([P.area for P in self.ParticleProp]) * self.area_factor
            * 0.99 + 1, 0)
    self.minimum_major = round(min([P.major_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 1.01)
    self.highest_major = round(max([P.major_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 1.01 + 1, 0)
    self.minimum_minor = round(min([P.minor_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 0.99)
    self.highest_minor = round(max([P.minor_axis_length for P in self.ParticleProp])
            * self.pxl_ratio * 0.99 + 1, 0)
    try:
        self.minimum_aspect = round(min([P.major_axis_length / P.minor_axis_length for P in
            self.ParticleProp]) * 10 * 0.99)
        self.highest_aspect = round(max([P.major_axis_length / P.minor_axis_length for P in
            self.ParticleProp]) * 10 * 0.99 + 1, 0)
    except ZeroDivisionError:
            self.minimum_aspect = 1
            self.highest_aspect = 100
    self.minimum_orientation = round((min([P.orientation for P in self.ParticleProp])
            + math.pi / 2) * 180 / math.pi)
    self.highest_orientation = round((max([P.orientation for P in self.ParticleProp])
            + math.pi / 2) * 180 / math.pi + 1, 0)

# update images and then handover the altered parameters to the controller,
# so they can be used with next tab
def update_controller_from_tab3(self, controller):
    self.get_filterdata_from_scale()
    # variables to controller
    controller.ellipse_draw_list = self.ellipse_draw_list
    controller.ParticleList = self.ParticleList
    controller.ParticleProp = self.ParticleProp
    controller.highest_major = self.highest_major
    controller.minimum_major = self.minimum_major
    controller.image_ellipses = self.image_ellipses
    controller.image_numbers = self.image_numbers
    controller.filter_used_list = self.filterdata
    # next tab 4
    controller.next_step(4)
    controller.show_frame(3)

# handover when saving the parameters
def make_save_update_from_tab3(self, controller):
    #variables to controller
    controller.ellipse_draw_list = self.ellipse_draw_list
    controller.filter_used_list = self.filterdata
```

VIII.6.1.5 Tab 4

```
#####################################
# ################################# #
# ### Tab 4 - Ellipse picking - ### #
# ################################# #
#####################################

# import libraries
import matplotlib
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.backends.backend_tkagg import NavigationToolbar2Tk
from matplotlib.figure import Figure
from matplotlib.pyplot import rcParams
import matplotlib.pyplot as plt
import matplotlib.colors as colors
import numpy as np
import tkinter as tk
from tkinter import ttk
import os
import datetime
import math
import pandas as pd

# global plot parameters
matplotlib.use("TkAgg")
LARGE_FONT = ("Verdana", 12)
GlobFontSize = 10
params = {
    'axes.labelsize': GlobFontSize,
    'font.size': GlobFontSize,
    'axes.titlesize': GlobFontSize,
    'font.family': "Calibri",
    'legend.fontsize': GlobFontSize,
    'xtick.labelsize': GlobFontSize,
    'ytick.labelsize': GlobFontSize,
    'text.usetex': False,
    'figure.constrained_layout.use': True,
    'lines.markersize': 3.0,
    'boxplot.flierprops.markersize': 3
}
rcParams.update(params)


# tab 4 code
class Page_EllipsePicking(tk.Frame):
    def __init__(self, parent, controller):
        # get dummys and variables from controller
        self.console_print_on = controller.console_print_on
        if self.console_print_on: print("statistic Page")
        self.ParticleList = controller.ParticleList
        self.ParticleProp = controller.ParticleProp
        self.image_path = controller.image_path
        self.img_x = controller.img_x
        self.img_y = controller.img_y
        self.ellipse_draw_list = controller.ellipse_draw_list
        self.new_imagesize = controller.new_imagesize
        self.image_w_axes = controller.image_w_axes.get()
        self.image = controller.image
        self.image_numbers = controller.image_numbers
        self.image_rgb = controller.image_rgb
        self.image_ellipses = controller.image_ellipses
        self.pxl_ratio = controller.pxl_ratio
        self.particles_at_border = controller.particles_at_border
        self.analyse_orientations_img = controller.analyse_orientations_img

        # variables of the tab
        self.was_not_clicked = True
        self.image_ellipsoids_at_tab4_drawn = 0
        self.var_feld = []
        # copy ellipse_draw_list to workingList
        self.DeleteList = []
        self.WorkingList = []
        for e in self.ellipse_draw_list:
            self.WorkingList.append(e)
        self.final_ParticleList = []
        self.old_ParticleList = []
        # create final images
        self.finalfigure = Figure(figsize=(self.img_x, self.img_y))
        if self.console_print_on: print(self.new_imagesize)
        # attention: figure needs (width, height), new_imagesize is (height, width)
        # --> need to be turned
        self.finalfig_orgdim = Figure(figsize=(self.new_imagesize[1] / 100,
                self.new_imagesize[0] / 100))

        ## create tab layout ##
        tk.Frame.__init__(self, parent)
        # devide window into 2 frames
        self.tab4_left = ttk.Frame(master=self)
        self.tab4_left.grid(row=0, column=0, sticky="nw")
        self.tab4_left.columnconfigure(0, minsize=220)
        self.tab4_right = ttk.Frame(master=self)
        self.tab4_right.grid(row=0, column=1, sticky="nsew")

        # make tab 4 menu on left container
        # toggle numbers on image
        self.number_show = tk.BooleanVar()
        self.number_show.set(True)
        self.check_numbers = ttk.Checkbutton(self.tab4_left, text="Switch ON Numbers",
                variable=self.number_show, command=self.image_ellipsoids_at_tab4)
        self.check_numbers.grid(row=0, column=0, sticky='ew')
        # button for save/export data
        self.butt4_3 = ttk.Button(self.tab4_left, text=">> Save/Export Data",
                command=lambda: controller.write_datafile())
        self.butt4_3.grid(row=2, column=0, sticky='ew', pady=5)
        ttk.Separator(self.tab4_left, orient='horizontal').grid(row=3, column=0, pady=10,
                sticky="ew")
        # button for the orientation plot of the visible image
        self.butt4_1 = ttk.Button(self.tab4_left, text="Plot Orientations",
                command=self.plot_orientations, state="enabled")
        self.butt4_1.grid(row=4, column=0, sticky='ew')
        # button for a distance plot
        self.butt4_4 = ttk.Button(self.tab4_left, text="Plot Distances",
                command=lambda: self.plot_distances(controller), state="enabled")
        self.butt4_4.grid(row=5, column=0, sticky='ew')
        # button for the angle analysis
        self.butt4_6 = ttk.Button(self.tab4_left, text="Analyse Angles",
                command=self.analyse_orientations)
        self.butt4_6.grid(row=7, column=0, sticky='ew')
        ttk.Separator(self.tab4_left, orient='horizontal').grid(row=8, column=0, pady=10,
                sticky="ew")
        # button for clearing all lables on image
        self.butt4_7 = ttk.Button(self.tab4_left, text="Clear Labels on Image",
                command=self.Button_deleteAllonImage, state="enabled")
        self.butt4_7.grid(row=9, column=0, sticky='ew')

        # start final evaluation
        self.finall_ellipsoids_image()
        self.create_final_ParticleList()

    # create window displaying the orientation plot
    def plot_orientations(self):
        angle = []
        yaxe = []
        fig_pol = Figure(figsize=(self.img_x, self.img_y))
        if self.console_print_on: print(self.img_x)
        if self.console_print_on: print(self.img_y)
        if self.console_print_on: print(fig_pol.dpi)
        c = fig_pol.add_subplot(121, projection='polar')
        for i in range(len(self.final_ParticleList) - 1):
            # orientation in polar coordinates as befor in ellipse_draw_list
            angle.append((self.final_ParticleList[i + 1][5]) / 180 * math.pi)
            yaxe.append(float(self.final_ParticleList[i + 1][6]))  # aspect ratio
            # mirroring the values to 90°-270° as unidentifiable if 0° or 180°
            angle.append(((self.final_ParticleList[i + 1][5]) + 180) / 180 * math.pi)
            yaxe.append(float(self.final_ParticleList[i + 1][6]))
        c.plot(angle, yaxe, 'b.')
        c.grid(True)
        c.set_rmax(np.max(yaxe) + 0.5)
        c.set_title('Orientation of Ellipses on Image')
        c.set_rlabel_position(-22.5)  # get radial labels away from plotted line

        # add a histogram
        angle_deg = []
        for i in angle:
            angle_deg.append(i * 180 / math.pi)
        h_orient = fig_pol.add_subplot(122)
        h_orient.hist(angle_deg, bins=120, range=(0, 360))
        h_orient.set_xlabel("Angle [°]")
        h_orient.set_ylabel("Number Count")
        h_orient.set_xlim(0, 360)
        h_orient.set_title("Angle Histogram")

        self.window_pol = tk.Tk()
        self.window_pol.geometry(str(int(self.img_x * fig_pol.dpi)) + "x" + str(int(self.img_y
                * fig_pol.dpi) + 50))
        canvas_polar = FigureCanvasTkAgg(fig_pol, self.window_pol)
        canvas_polar.get_tk_widget().grid(row=0, column=0, rowspan=1)
        # toolbar
        toolbar_polar = NavigationToolbar2Tk(canvas_polar, self.window_pol, pack_toolbar=False)
        toolbar_polar.update()
        toolbar_polar.grid(row=1, column=0, rowspan=1)
        self.final_polar_plot=fig_pol

    # clear the lables on the image
    def Button_deleteAllonImage(self):
        self.DeleteList = []
        for P in self.ellipse_draw_list:
            self.DeleteList.append(P[6])
        self.WorkingList = []
        self.was_not_clicked = False
        self.finall_ellipsoids_image()
        self.create_final_ParticleList()

    # event handler when clicked on image
    def onclick(self, event):
        if self.was_not_clicked:
            self.DeleteList = []
            self.WorkingList = []
            for e in self.ellipse_draw_list:
                self.WorkingList.append(e)
            if self.console_print_on: print(len(self.ellipse_draw_list))
            self.was_not_clicked = False
        # when there was a click on the image area
        if not event.xdata == None and not event.ydata == None:
            if self.console_print_on:
                print('%s click: button = %d, x = %d, y = %d, xdata = %f, ydata = %f' %
                ('double' if event.dblclick else 'single', event.button,
                event.x, event.y, round(event.xdata, 0), round(event.ydata, 0)))

            # ellipse draw list element [number, posy, posx, minor, major, orientation, org_no,
                array(cc), array(rr)]
            for i in self.ellipse_draw_list:
                if [int(round(event.xdata, 0)), int(round(event.ydata, 0))]
                        in list(map(list, zip(i[7], i[8]))):
                    if self.console_print_on: print("clicked on particle region")
                    # click is on Particle -> delete from workingList
                    if not i[6] in self.DeleteList:
                        self.DeleteList.append(i[6])
                        for el in self.WorkingList:
                            if i[6] == el[6]:
                                self.WorkingList.remove(el)
                                if self.console_print_on:
                                        print("one particle deleted from WorkingList")
                            else:
                                if self.console_print_on: print("is not particle to be removed")
                    # Particle was formerly deleted -> restore
                    else:
                        self.WorkingList.append(i)
                        for eli in self.DeleteList:
                            if i[6] == eli:
                                if self.console_print_on:
                                        print("one particle deleted from DeleteList")
                                self.DeleteList.remove(eli)
                        if self.console_print_on: print("Particle restored")

            if self.console_print_on: print(self.DeleteList)
            # update image and list
            self.finall_ellipsoids_image()
            self.create_final_ParticleList()

        # when click was not on image area
        else:
            if self.console_print_on: print("out of region click")

    # update particle list
    def create_final_ParticleList(self):
        if self.console_print_on: print("start final particle list")
        self.old_ParticleList = self.ParticleList
        new_list = []
        new_list.append(self.ParticleList[0])
        # working list has draw infos but not actual properties,
        # so that compare with original particle list is necessary
        for x in range(0, len(self.WorkingList), 1):
            for k in range(1, len(self.ParticleList), 1):
                if self.WorkingList[x][0] == self.ParticleList[k][0]:
                    new_list.append(self.ParticleList[k])
        self.final_ParticleList = new_list
        if self.console_print_on: print("new_list ready")

    # create image of colorful ellipses with given coordinates in working list
    # WorkingList with elements [number, posy, posx, minor, major, orientation, org_no, array(cc),
                                array(rr)]
    def finall_ellipsoids_image(self):
        self.ellipse_color = plt.cm.hsv
        # create black image to draw ellipses
        # -> np.zeros need shape like new_imagesize (height, width)
        image_zero = np.zeros([self.new_imagesize[0], self.new_imagesize[1], 3])
        image_zero.fill(0)
        # draw colorful ellipses from workinglist with the cc,rr coordinate arrays at
        # position 7 (x-coord) and 8 (y-coord) - note that y,x coords from np array
        for i in range(len(self.WorkingList)):
            image_zero[self.WorkingList[i][8], self.WorkingList[i][7]]
                = colors.to_rgb(self.ellipse_color(abs((self.WorkingList[i][5]) / (math.pi))))
        self.image_ellipses = image_zero
        self.image_ellipsoids_at_tab4()

    # image ellipsoids on original image with or without numbers
    def image_ellipsoids_at_tab4(self):
        # if no image is drawn yet
        if self.image_ellipsoids_at_tab4_drawn == 0:
            self.fig_el = Figure(figsize=(self.img_x * 2, self.img_y * 2 + 0.4))
            ellipse_img = self.fig_el.add_subplot(111)
            self.ellipsoids_on_img_tab4 = ellipse_img.imshow(self.image_ellipses,
                                                        cmap=plt.cm.nipy_spectral)
            ellipse_img.imshow(self.image_rgb, cmap=plt.cm.gray, alpha=0.6)
            # draw numbers if checkbox is ON
            if self.number_show.get():
                self.number_show_img = ellipse_img.imshow(self.image_numbers,
                        cmap=plt.cm.gray, alpha=1)
            if self.image_w_axes:
                ellipse_img.axis("on")
                ellipse_img.set_title("Ellipses on Original Image")
                ellipse_img.set_xlabel("Horizontal Pixel ID")
                ellipse_img.set_ylabel("Vertical Pixel ID")
            else:
                ellipse_img.axis("off")

            canvas4 = FigureCanvasTkAgg(self.fig_el, self.tab4_right)
            # create mouse click event to choose ellipses
            canvas4.mpl_connect('button_press_event', self.onclick)
            canvas4.get_tk_widget().grid(row=0, column=0, sticky="nsew", columnspan=1)
```

```
            self.image_ellipsoids_at_tab4_drawn = 1
        # if image is present -> update
        else:
            # update image
            self.ellipsoids_on_img_tab4.set_data(self.image_ellipses)
            if self.number_show.get() == False:
                self.number_show_img.set_alpha(0)
            else:
                self.number_show_img.set_alpha(1)
            self.fig_el.canvas.draw_idle()
            if self.console_print_on: print("ellipsoids on tab3 updated")

        # adress image on display for later save
        self.finalfigure = self.fig_el

        # create image with orginial dimensions, for saving later on
        # attention: figure needs (width, height), new_imagesize is (height, width)
        # --> need to be turned
        fig_el_orgdim = Figure(figsize=(self.new_imagesize[1] / 100, self.new_imagesize[0] / 100))
        ellipse_img_orgdim = fig_el_orgdim.add_subplot(111)
        ellipse_img_orgdim.imshow(self.image_ellipses, cmap=plt.cm.nipy_spectral)
        ellipse_img_orgdim.imshow(self.image_rgb, cmap=plt.cm.gray, alpha=0.6)
        ellipse_img_orgdim.axis("off")
        self.finalfig_orgdim = fig_el_orgdim
        if self.console_print_on: print('the dimensions of image prior save: {}'
                                .format(self.finalfig_orgdim.get_size_inches()))
        # set (width, height), but from shape get (height, width),
        # do use original pixel size, else devide 100
        self.finalfig_orgdim.set_size_inches(self.new_imagesize[1], self.new_imagesize[0])

    # update images and then handover the altered parameters to the controller
    def update_controller_from_tab4(self, controller):
        self.plot_orientations()
        controller.final_ParticleList = self.final_ParticleList
        controller.ParticleList = self.ParticleList
        controller.finalfigure = self.finalfigure
        controller.finalfig_orgdim = self.finalfig_orgdim
        controller.image_ellipses = self.image_ellipses
        controller.final_polar_plot=self.final_polar_plot
        controller.analyse_orientations_img = self.analyse_orientations_img
        if self.console_print_on: print("Controller updated")

    # handover when saving the parameters
    def make_save_update_from_tab4(self, controller):
        if controller.save_angle_dist.get(): self.plot_orientations()
        controller.final_ParticleList = self.final_ParticleList
        controller.finalfigure = self.finalfigure
        if controller.save_angle_dist.get(): controller.final_polar_plot=self.final_polar_plot
        controller.finalfig_orgdim = self.finalfig_orgdim
        controller.image_ellipses = self.image_ellipses
        controller.analyse_orientations_img = self.analyse_orientations_img

###########################
## ##################### ##
## # Statistic windows # ##
## ##################### ##
###########################
    # calculate distances from particles
    def create_distance_list(self, particles):
        dist_list = []
        p = particles
        # create list: x coordinate (in particleList[element][0]), y, distance, angle
        # (not used) input-particle list style: "Partikel_No.","posy", "posx", "minor", "major",
                                                "orientation_in_deg", "ratio"
        # input with working list [number, posy, posx, minor, major, orientation, org_no]
        for P1 in range(0,len(p),1):
            for P2 in range(P1+1,len(p),1):
                # diameter in nm
                d = math.sqrt((p[P2][2]-p[P1][2])**2 + (p[P2][1]-p[P1][1])**2) * self.pxl_ratio
                try:
                    deg = math.degrees(-math.atan((p[P2][1] - p[P1][1]) / (p[P2][2] - p[P1][2])))
                except:
                    deg = 90
                # orientation in polar coordinates as befor in ellipse_draw_list
                element = [p[P1][0], p[P2][0], d, deg]
                dist_list.append(element)
                # double orientation in polar coordinates as befor in ellipse_draw_list
                element2 = [p[P1][0],p[P2][0],d, deg + 180]
                dist_list.append(element2)
        if self.console_print_on: print("distance list created")
        return dist_list

    # show the distances of the particles
    def plot_distances(self, controller):
        particle_distance = self.create_distance_list(self.WorkingList)
        xdist_achse = []
        angle_dist_achse = []
        for i in range(len(particle_distance)):
            # write distances into drawlist
            xdist_achse.append(particle_distance[i][2])
            # write angles into draw list
            angle_dist_achse.append((particle_distance[i][3]) / 180 * math.pi)
        # plot histogram and polarplot
        self.window_distances = tk.Tk()
        self.window_distances.geometry("1000x800")
        figt = Figure(figsize=(self.img_x, self.img_y))
        be = figt.add_subplot(121)
        be.hist(xdist_achse, bins=100)
        be.set_title("Distance")
        be.set_xlabel("Distance [nm]")
        be.set_ylabel("Numbers")
        de = figt.add_subplot(122, projection='polar')
        de.plot(angle_dist_achse, xdist_achse, marker=".", linestyle="None", markersize=1)
        canvas_polar2 = FigureCanvasTkAgg(figt, self.window_distances)
        canvas_polar2.get_tk_widget().grid(row=0, column=1, rowspan=1)
        # toolbar
        toolbar_polar2 = NavigationToolbar2Tk(canvas_polar2, self.window_distances,
                pack_toolbar=False)
        toolbar_polar2.update()
        toolbar_polar2.grid(row=1, column=1, rowspan=1)
        # save the distance figure imediately to folder
        time = datetime.datetime.now().strftime("%Y-%m-%d %H-%M-%S")
        split_name = os.path.splitext(str(controller.image_path[1]))
        figt.savefig((str(controller.image_path[0]) + "\\" + str(split_name[0]) + '_distancePlot_'
                        + time + ".png"), dpi=100, bbox_inches="tight", pad_inches=0)

    # orientation correlation
    def analyse_orientations(self):
        dist_list = []
        p = self.final_ParticleList[1:]
        # create list: x coordinate (in particleList[element][0]), y, distance, angle
        # (not used) input-particle list style: "Partikel_No.","posy", "posx", "minor", "major",
                "orientation_in_deg", "ratio"
        # input with working list [number, posy, posx, minor, major, orientation, org_no]
        for P1 in range(0,len(p),1):
            for P2 in range(P1+1,len(p),1):
                d = math.sqrt((p[P2][2]-p[P1][2])**2 + (p[P2][1]-p[P1][1])**2)
                if p[P1][5]>p[P2][5]:
                    compare_angle = round((p[P1][5]-p[P2][5]), 0)
                else:
                    compare_angle = round((p[P2][5]-p[P1][5]), 0)
                # print('angle 1 '+str(p[P1][5])+ '  angle 2 '+str(p[P2][5]))
                # the angles inbetween the ellipses vectors are independent of direction,
                # just angles of 0-90° are possible
                if compare_angle>180:
                    compare_angle = 360 - compare_angle
                if compare_angle>90 and compare_angle<=180:
                    compare_angle = 180 - compare_angle
                try:
                    deg = math.degrees(-math.atan((p[P2][1] - p[P1][1]) / (p[P2][2] - p[P1][2])))
                except:
                    deg = 90
                # orientation in polar coordinates as befor in ellipse_draw_list
                element = [p[P1][0], p[P2][0], d, deg, compare_angle]
                dist_list.append(element)
```

```
xdist_achse = []
compare_angle_dist = []
for i in range(len(dist_list)):
    # write distances into drawlist
    xdist_achse.append(dist_list[i][2])
    # write angles into drawlist
    compare_angle_dist.append(dist_list[i][4])

dataframe = pd.DataFrame(xdist_achse, columns=['distance_pxl'])
dataframe['compare_angle'] = compare_angle_dist
binsize = 10
dataframe['bins'] = dataframe['distance_pxl'] // binsize
mean_dataframe = dataframe.groupby('bins').mean()
# create figure
analyse_orientations_img = Figure(figsize=(3, 2.2))
ada = analyse_orientations_img.add_subplot()
if len(dataframe.index) < 10:
    ada.plot(mean_dataframe['distance_pxl'] * self.pxl_ratio, np.cos(2 / 180 * np.pi
                * mean_dataframe['compare_angle']), 'ro', label='mean')
else:
    ada.plot(mean_dataframe['distance_pxl'] * self.pxl_ratio, np.cos(2 / 180 * np.pi
                * mean_dataframe['compare_angle']), 'r', label='mean')
ada.set_title('Evaluation of Angles')
ada.set_ylabel('Mean Angle cos(2*({0}(0)-{0}(r)))'.format(chr(952)))
ada.set_xlabel('Distance r [nm]')
ada.set_ylim(-1.05, 1.05)
self.analyse_orientations_img = analyse_orientations_img
# show figure
self.window_poly = tk.Tk()
canvas_ada = FigureCanvasTkAgg(analyse_orientations_img, self.window_poly)
canvas_ada.get_tk_widget().grid(row=0)
# toolbar
toolbar_polar = NavigationToolbar2Tk(canvas_ada, self.window_poly, pack_toolbar=False)
toolbar_polar.update()
toolbar_polar.grid(row=1)
```

VIII.6.2 Code to Chapter IV.4.1 – Correcting Slice Positions

Condensed Python source code for the correction calculations to create the .info file. The data used is taken from Label Analysis of the separated fiducials. The created .info file is for the particle reconstruction of the same image stack.

```
# import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import math

# define parameters:
# zBinSize = distance of slices counting to same bin (here <1 fiducial diameter)
# imgZBin = original slice height
# no_Images = no of images after export of 3D volume to tif
zBinSize = 0.300  # µm
imgZBin = 0.040  # µm
no_Images = 300

# read table extracted from AMIRA Label Analysis
data_Zero = pd.read_excel('datafile.xlsx', sheet_name='Sheet')

# kick zeros from filtering + reindex
data = data_Zero.loc[data_Zero['Length3d (µm) '] > 0].copy().reset_index(drop=True)

# Create group number of z-Bin by floor division as integer
# '//' not used (see 'rounding errors in floor division':
# https://stackoverflow.com/questions/38588815/rounding-errors-in-python-floor-division )
data['zBin'] = 0
for i in range(0, len(data.index)):
    data.at[i, 'zBin'] = math.floor(float(data.at[i, 'BaryCenterZ (µm) ']) /
                                zBinSize)

# define the diameter for the correction as the smallest from x or y bounding box (float)
i = 0
for x, y in zip(data['BoundingBoxDx (µm) '], data['BoundingBoxDy (µm) ']):
    if x < y:
        data.at[i, 'UsedDiameter'] = float(x)
    else:
        data.at[i, 'UsedDiameter'] = float(y)
    i += 1

# Calculate the correction factor using the defined diameter
data['zCorrector'] = data['BoundingBoxDz (µm) '] / data['UsedDiameter']

# Get median for each zBin and each column and write to new dataframe medianValues
# index is column zBin
medianValues = data.groupby('zBin').median()

# Assign a correction factor to each bin for later multiplication
# assign also correction to bins with smaller z value then first fiducial
# --> value is first (nearest) correction value
maxZ = no_Images * imgZBin
bins = 1 / medianValues['zCorrector']
bins = bins.rename('Correction')
bins = bins.reindex(np.arange(0, maxZ / zBinSize, 1), method='nearest')
```

```
# Next, the .info-file for reloading to AMIRA is prepared
# create imagelist with z position with steady distortion
# of original slice height (imgZBin = 0.04 µm)
# --> round() needed because of python value inaccuracy;
# *1000 as values for slices in nm (previous µm)
imagePositions = pd.DataFrame(range(0, no_Images), columns=['slice_no'])
imagePositions['org_height'] = round(imagePositions['slice_no'] * imgZBin *1000, 2)

# make group numbers of z-Bin by floor division (int)
imagePositions['zBin'] = 0
for i in range(0, len(imagePositions.index)):
    imagePositions.at[i, 'zBin'] = math.floor(float(imagePositions.at
                                    [i, 'org_height']) / (zBinSize * 1000))

# assign correction values to respective image slice
imagePositions = imagePositions.join(bins, on='zBin')

# calculate the new z location of image slice (float)
imagePositions['newheight'] = 0.0
for i in range(0, len(imagePositions.index) - 1):
    imagePositions.at[i + 1, 'newheight'] = imagePositions.at[i, 'newheight'] +
      imgZBin * 1000 * float(imagePositions.at[i, 'Correction'])

# set the last slice to a value for constant grid interpolation
constant_grid = 10  # nm
imagePositions.at[len(imagePositions.index) - 1, 'newheight'] =
  round(imagePositions.at[len(imagePositions.index) - 1, 'newheight'] /
  constant_grid, 0) * constant_grid

# write .info-file
infoFileName = 'imagefiles.info'
info_file = open(infoFileName, 'w')
# write header of .info file
info_file.write('# AMIRA Stacked Slices Reconstruction by min of XorY and Z-BoundingBox \n'
                '\n'
                '# bin height: ' + str(zBinSize) + 'µm \n'
                '\n'
                '# Filename of imagefiles \n'
                'pathname I:/AMIRA Processed/Final3DCorrectionXorY-Z/\n'
                '\n'
                '# Pixel size in x- an y-direction (nm) \n'
                'pixelsize 10 10 \n'
                '\n'
                'offset 0 0 \n'
                '\n'
                '# Imagelist with z-positions (nm) \n')
# write image name and new z-position (in nm)
# # example line: "img001.tif"  40
img_name = 'img'
for i in range(0, no_Images):
    j = '"' + img_name + str(i).zfill(3) + '.tif" '
        + str(imagePositions.at[i, 'newheight'])
    info_file.write("%s\n" % j)

info_file.close()
```

VIII.6.3 Code to Chapter V.1.5 – 2D Positions Dataset Generation

The following code snippets can generate the depicted images of close-packed ellipses. The input parameters are *xB* and *yB* as image bounding box (image size), *sp* as spacer pixels around ellipses, *r* as starting radius, *asp* as AR, and *ori* as orientation of the stretching.

First, the parameters are handed to the class and the new long and short axis after stretching are calculated by volume conservation:

```
def parameters(self, xB, yB, sp, r, asp, ori):
    # write variables to class
    self.xBox = xB
    self.yBox = yB
    self.spacer = sp  # pxl size of black rim around the particle
    # variables of particle
    self.start_r = r
    self.aspectratio = asp  # major/minor
    orientation_deg = ori

    # radius of minor
    self.minor = int(self.start_r / (self.aspectratio**(1 / 3)))
    # radius of major
    self.major = self.aspectratio * self.minor  # same as: int(start_r * (aspectratio**(2/3)))

    self.stretch_x = self.major / self.start_r
    self.stretch_y = self.minor / self.start_r

    self.phi = orientation_deg / 360 * 2 * np.pi  # deg to rad
    self.ellipse_color = plt.cm.hsv
```

Then a square or hex pattern field of point coordinates is created as a starting lattice:

```
# create square pattern coordinates
def squarePattern(self):
    sqr_array = []
    k = 0
    a = 2 * (self.start_r + self.spacer)
    h = a
    for i in range(int(self.xBox / 50)):
        for j in range(int(self.yBox / 50)):
            point = [i * a, j * h, self.start_r, self.start_r, self.phi]
            sqr_array.append(point)
            k += k
    return sqr_array


# create hexagonal close-packed pattern coordinates
def ellipseHexPatternAffine(self):
    hex_array = []
    k = 0
    a = 2 * (self.start_r + self.spacer)
    h = (self.start_r + self.spacer) * (3**0.5)
    for j in range(int((self.yBox) / h * self.aspectratio) + 4):
        for i in range(int((self.xBox) * self.aspectratio / a) + 4):
            if j % 2:
                hex_point = [a / 2 + i * a, (j * h), self.start_r, self.start_r, self.phi]
            else:
                hex_point = [i * a, (j * h), self.start_r, self.start_r, self.phi]
            hex_array.append(hex_point)
            k += k
    return hex_array
```

After the lattice with particle positions is created, the point coordinates are transferred to new coordinates by rotating, transfer and back rotation:

```
def affineTransformation(self, input_array):
    turn_array = []
    stretched_array = []
    turn_bck_array = []
    shift_array = []
    final_array = []

    # turn points coordinates
    for i in range(len(input_array)):
        x = input_array[i][0]
        y = input_array[i][1]
        # x' = x* np.cos(phi)- y*np.sin(phi)
        # y' = x* np.sin(phi) + y*np.cos(phi)
        turn_point = [x * np.cos(self.phi) - y * np.sin(self.phi),
                      x * np.sin(self.phi) + y * np.cos(self.phi),
                      self.start_r, self.start_r, 0]
        turn_array.append(turn_point)

    # stretch point coordinates
    for i in range(len(turn_array)):
        x = turn_array[i][0]
        y = turn_array[i][1]
        new_x = x * self.stretch_x
        new_y = y * self.stretch_y
        stretched_array.append([new_x, new_y, self.minor, self.major, 0])

    # turn back point coordinates
    if((stretched_array[1][0] - stretched_array[0][0]) == 0):
        self.phi_reverse = np.pi / 2
    else:
        self.phi_reverse = np.arctan((stretched_array[1][1] - stretched_array[0][1]) /
                                     (stretched_array[1][0] - stretched_array[0][0]))
    for i in range(len(stretched_array)):
        x = stretched_array[i][0]
        y = stretched_array[i][1]
        bck_turn_point = [x * np.cos(-self.phi_reverse) - y * np.sin(-self.phi_reverse),
                          x * np.sin(-self.phi_reverse) + y * np.cos(-self.phi_reverse),
                          self.minor, self.major, self.phi_reverse]
        turn_bck_array.append(bck_turn_point)

    # find center of particle field
    minVal = np.amin(turn_bck_array, axis=0)
    maxVal = np.amax(turn_bck_array, axis=0)
    shift_x = -minVal[0] - (maxVal[0] - minVal[0]) / 2 + self.xBox / 2
    shift_y = -minVal[1] - (maxVal[1] - minVal[1]) / 2 + self.yBox / 2

    # shift center particle field to center of image
    for i in range(len(turn_bck_array)):
        x = turn_bck_array[i][0]
        y = turn_bck_array[i][1]
        shift_point = [x + shift_x, y + shift_y, self.minor, self.major, self.phi_reverse]
        shift_array.append(shift_point)
    for i in range(len(shift_array)):
        if 0 - self.major <= shift_array[i][0] <= self.xBox + self.major
          and -self.major <= shift_array[i][1] <= self.yBox + self.major:
            final_array.append(shift_array[i])

    return final_array  # list of [x, y, minor, major, orientation]
```

The image of the ellipses was then created with the data of the *final_array*.

```
    def image_ellipses(self, data):
        image_zero = np.zeros([self.yBox, self.xBox, 3])
        shape = (self.yBox, self.xBox)
        image_zero.fill(0)
        # draw colorful ellipses from workinglist
        for i in range(len(data)):
            # draw ellipses into blank image at given position in color
            rr, cc = ellipse(data.at[i, 'y'], data.at[i, 'x'], data.at[i, 'minor'],
                             data.at[i, 'major'], shape=shape, rotation=data.at[i, 'orientation'])
            try:
                image_zero[rr, cc] = colors.to_rgb('white')
            except:
                None
        image_ellipses = image_zero

        return image_ellipses
```

VIII.6.4 Code to Chapter V.1.6.1 – $g(r)$ Calculation

The code for calculating the $g(r)$ includes Dr. Bernd A.F. Koperas code calculating the $g(r)$ from particle positions in 2D. The code was adjusted by myself to be used with the particle positions separately extracted from microscope images using the python program of Chapter III.3.

The general steps are:

First, the file with the particle data (`ellipseDetectFile`) is loaded in the app. To use the code of Dr. Bernd A.F. Kopera, the particle data is stored in objects (`Particles`), containing the x/y-coordinates and the particle diameter, via an `ParticleSample` class (from Dr. Bernd A.F. Koperas code, not shown here).

Second, for each particle object the local $g(r)$ is calculated.

Third, a global $g(r)$ is made, the values exported into an excel file and then displayed in a diagram.

The following code snippet displays the calculation of the $g(r)$ function.

```
# start the calculation application
def start_calculate(self):
    # get image path from opened particle position file
    image_path = os.path.split(os.path.abspath(self.ellipseDetectFile))
    # Experiment folder used for saving files
    self.expFolder = Path(image_path[0])
    expName = "2D Single Frame Test"
    Px = 1  # voxel size in x-direction
    Py = 1  # voxel size in y-direction
    # PixelSizes = {"x": Px, "y": Py}
    # logging and plotting configuration
    # fontsize, plotsize
    confMatplotlib(10, [4.5 * 1.1, 4.5])
    self.Sentinel = getCustomLogger(self.expFolder, expName)
    self.Results = AnalysisData(expName, self.expFolder)
```

```
        # data frame to get next neighbours with: bin id, amount of particle in local bin, aso.
        self.number = pd.DataFrame(columns=['RIdx', 'Local_Gr[RIdx]', 'ShellBoxArea', 'Local_Density'])
        self.data = pd.DataFrame(columns=['x', 'y', 'minor', 'major', 'orientation'])

        csvFile = Path(self.ellipseDetectFile)
        print(self.rawdata)
        print(csvFile.name)
        print(self.expFolder)
        for i in range(len(self.rawdata)):
            if 0 < self.rawdata['x'].iloc[i] < self.xBox and 0 < self.rawdata['y'].iloc[i] < self.yBox:
                self.data.loc[len(self.data.index)] = self.rawdata.iloc[i]

        if self.test_start_parameter:
            self.data['minor'] += self.spacer
            self.data['major'] += self.spacer
            # particle diameter in pxl - input radius
            self.PartDiam = float(2 * self.data['minor'].mean()) / self.scale
        # use the minor diameter for smallest particle contact from SEM image evaluation
        else: # particle diameter in pxl - input diameter
            self.PartDiam = self.part_diam_SEM / self.scale
        sample = ParticleSample(csvFile.name, self.expFolder, [self.data.x, self.data.y],
                                Diameter=self.PartDiam)

        self.Sentinel.info('Loaded {} as a sample'.format(csvFile))
        self.Results.PSampleData.append(sample)
        self.Results.ExpID.append(sample.ExpID)

        scatter = plt.scatter(self.data.x, self.data.y, c='red', marker='.')
        ax = scatter.axes
        # invert yaxis, necessary as evaluated images have their orign
        # on upper left corner and py.plot has it lower left
        ax.invert_yaxis()
        ax.set_aspect('equal')
        name = str(csvFile.name) + "_scatter.png"
        plot_folder_path = self.expFolder / "Plots"
        plot_folder_path.mkdir(parents=False, exist_ok=True)
        plt.tight_layout()
        plt.show(block=False)
        plot_path = plot_folder_path / name
        plt.savefig(plot_path)

        # set max radus of evaluation
        maxR = (int((self.PartDiam * self.times_diameter) / self.pxl_bin) + 1) * self.pxl_bin
        print(maxR)

        if self.test_start_parameter:
            ellipse_testdata.plot_test_image(self, imgdata=self.rawdata, dpi=300,
                                DestPath=self.ellipseDetectFile, to_save=True)
        self.compute_gr(MaxR=maxR, binNum=int(maxR / self.pxl_bin), binWidth="tight", useKDtree=False)
        self.Results.PlotRDF(300)
        self.Results.saveResults()

        addUpPath = self.expFolder / "Plots" / (image_path[1] + "_addup.csv")
        self.addUp.to_csv(addUpPath, sep=',', columns=['RIdx', 'Local_Max_Particle_No',
                                'Local_Density'])

    # Wrapper of computing the radial distribution function
    def compute_gr(self, MaxR, binNum, binWidth="tight", useKDtree=False):
        """
        Parameters:
        MaxR : Maximal radial distance
        binNum : Number of equidistant bins between 0 and Rmax
        binWidth : either "tight" (default) or float
            if "tight" then the bins touch each other
            if float then all bins have the specified with and may overlapp or miss
        useKDtree: Use a KDtree for efficient removal of particles outside the maximal radius
        """
        for idx, Sample in enumerate(self.Results.PSampleData):
            [r, Gr, dr, meanD] = self.RDF_Analytic2DNorm(Sample, MaxR, binNum, binWidth)
            print(str(dr) + ' is binWidth')
            self.Results.GrResults.append([r, Gr, dr, meanD])
            self.Sentinel.info("Computed g(r) from Image: {}".format(Sample.ExpID))
```

```
# Computes the 2D radial distribution function from all particles.
def RDF_Analytic2DNorm(self, Sample, MaxR, binNum, binWidth="tight", useKDtree=False):
    """
    Parameters:
    Sample: Contains individual particle positions and the boundary
    MaxR: Maximal radial distance
    binNum: Number of equidistant bins between 0 and Rmax
    binWidth: either "tight" (default) or float
        if "tight" then the bins touch each other
        if float then all bins have the specified width and may overlap
            or show gaps between bins
    useKDtree: Use a KDtree for efficient removal of particles outside the maximal radius

    Returns:
    r : Radial bin center positions
    Gr : Values for the radial distribution function in each bin
    """
    if Sample.PartNum < 300:
        self.Sentinel.warning("There are only {} particles in your sample."
                              " Be careful when interpreting g(r)."
                              "".format(Sample.PartNum))
    if MaxR <= 0:
        ErrorMsg = "Negative radii are not allowed."
        self.Sentinel.error(ErrorMsg)
        raise ValueError(ErrorMsg)
    if binNum <= 0:
        ErrorMsg = "Negative bin numbers are not allowed."
        self.Sentinel.error(ErrorMsg)
        raise ValueError(ErrorMsg)
    if binWidth != "tight" and binWidth <= 0:
        ErrorMsg = "Negative bin widths are not allowed."
        self.Sentinel.error(ErrorMsg)
        raise ValueError(ErrorMsg)

    # compute the left boundary values
    r = np.linspace(0, MaxR, binNum, endpoint=False)

    # compute the bin width if "tight" is choosen
    if binWidth == "tight":
        dr = r[1]
    else:
        try:
            dr = int(binWidth)
        except ValueError:
            ErrorMsg = "binWidth: {} not recognized.".format(binWidth)
            self.Sentinel.error(ErrorMsg)
            raise ValueError(ErrorMsg)
    self.Sentinel.info("The bin width is set to: {}".format(dr))

    # Create boundaries to center values
    r_boundaries = r + dr / 2

    # Create a list with bin boundaries squared for histograms
    # This increases performance for bin assignment, because no sqrt()
    # is required
    # add 0 in front of array, first box is: 0 to dr/2
    squaredBinBoundaries = np.append(r[0], r_boundaries)**2

    # maximal radial distance
    MaxDist = r[-1] + dr / 2

    # Gr averaged over all particles
    Global_Gr = np.zeros(binNum)

    # create a list with [(X1, Y1),....] coordinates
    # --> in Dr. Bernd A. F. Koperas code this is the variable 'Particles'
    ParticleCoords = np.array(Sample.getParticleList()).transpose()

    if useKDtree is True:
        ParticleTree = spatial.cKDTree(ParticleCoords)

    # use every particle as the center once
    # CentralP contains the x,y from the particle, k is list number
```

```
        for k, CentralP in enumerate(ParticleCoords):
            # create list of neighbours id number --> different to Dr. Bernd A. F. Kopera
            if useKDtree is True:
                NeighbourIdx = ParticleTree.query_ball_point(CentralP, MaxDist, p=2, eps=dr)
            else:
                NeighbourIdx = np.arange(len(ParticleCoords))

            # Don't use the central particle itself --> different to Dr. Bernd A. F. Kopera
            NeighbourIdx = np.setdiff1d(NeighbourIdx, [k], True)
            Neighbours = ParticleCoords[NeighbourIdx]

            # Calculate g(r)
            Local_Gr = self.computeLocalGr2D(CentralP, Neighbours, r, dr, squaredBinBoundaries, Sample)
            Global_Gr += Local_Gr

            # show progress
            if k % 10 == 0 and k > 0:
                self.Sentinel.info("Finished g(r) at particle "
                                   "{} of {}".format(k, Sample.PartNum))

        # get average number of particles in bin
        print(self.number)
        # get all filled bins
        uniqueRIdx = np.unique(self.number['RIdx'].to_numpy(), return_counts=False)
        print(uniqueRIdx)
        # create result dataframe
        self.addUp = pd.DataFrame(columns=['RIdx', 'Local_Max_Particle_No', 'Local_Density'])
        counter = 0
        for u in uniqueRIdx:
            max_no = 0
            addup_value = 0
            for i in range(len(self.number)):
                if self.number['RIdx'].iloc[i] == u:
                    # is there more neighbours?
                    if self.number['Local_Gr[RIdx]'].iloc[i] > max_no:
                        max_no = self.number['Local_Gr[RIdx]'].iloc[i]
                    addup_value += self.number['Local_Density'].iloc[i]
            self.addUp.loc[len(self.addUp.index)] = [u, max_no, addup_value]
            if counter % 10 == 0 and counter > 0:
                self.Sentinel.info("Addup finished at bin no. " "{} of {}".format(counter,
                                    len(uniqueRIdx)))
            counter += 1
        print(self.addUp)
        # normalize by the mean particle density and amount of particles
        # --> same as Dr. Bernd A. F. Koperas Code
        MeanDensity = Sample.PartNum / Sample.Volume
        Global_Gr = Global_Gr / MeanDensity
        Global_Gr = Global_Gr / Sample.PartNum

        self.Sentinel.info("Computed g(r) from sample: {}"
                           "".format(Sample.ExpID))
        #return the normData, globData, new dr and meanDensity
        return [r, Global_Gr, dr, MeanDensity]

    # Computes the local g(r) around one particle.
    def computeLocalGr2D(self, CentralP, Neighbours, r, dr, squaredBinBoundaries, Sample):
        """
        Parameters:
        CentralP: Coordinates of the central particle
        Neighbours: list of Particles [(X, Y),...]
        r: List with the radial distance bins centers
        dr: Width of the radial distance bins
        squaredBinBoundaries : The squared bin boundaries. Comparing the squared distance between
            particles with this list leads to faster lookup of the correct bin.
        Sample: Contains the boundaries of the sample
        """
        # Calculate the squared distance to each neighbour
        sqDistances = ((Neighbours - CentralP)**2).sum(axis=1)
        # new: make sqDistance integer as just total pxl possible,
        # avoids rounding errors in calculation,
        # e.g. radius 51 and bin borders diectly on pxl numbers
        # (former adjustment): diagonal distance 102.0000000000002 would be
        # in bin 102-103 not in 101-102
        # horizontal distance 102 so bin 101-102
        if self.test_start_parameter:
            sqDistances = np.round(sqDistances, 0)

        # Check how many neighbours belong to each bin
        Local_Gr = np.histogram(sqDistances, bins=squaredBinBoundaries)[0]
        Local_Gr = Local_Gr.astype('float64')

        # change the origin such that the CentralP is at the origin
        Sample.setCOSYCenter(CentralP)

        # normalize with the shell volume
        for RIdx in range(0, len(r), 1):  # old:from 1 to maxR-1 new: 0 to maxR-1
            # Skip if no particles are in the bin --> New to Dr. Bernd A. F. Koperas code
            if Local_Gr[RIdx] == 0:
                continue

            # Calculate the shell area
            assert Sample.Dimension == 2
            # the following is new, as previous versions had
            # r[RIdx - 1] + (dr / 2),r[RIdx] + (dr / 2)
            # which is the same when starting at 1, missing 0 position
            ShellBoxArea = self.CircShellVolume(r[RIdx] - (dr / 2),
                                                r[RIdx] + (dr / 2), Sample)
            # new
            # exclude data if shell area is smaller than 5
            # just necessary as then, one particles in small area has extremly large impact on Local_Gr
            # normally not used when just half size of sample box area is used for binning
            if ShellBoxArea < 5:
                self.Sentinel.info("Exclude bin {} from particle with coords:
                                    {} because of area beeing just: {}"
                                    .format(RIdx, CentralP, ShellBoxArea))
                Local_Gr[RIdx] = 0
            else:
                self.number.loc[len(self.number.index)]=[RIdx, Local_Gr[RIdx], ShellBoxArea,
                                                            Local_Gr[RIdx]/ ShellBoxArea]
                Local_Gr[RIdx] /= ShellBoxArea

        return Local_Gr

    # Computes the intersection area between a circle and a box.
    # adapted from Dr. Bernd A. F. Kopera
    def CircleArea(self, Rc, LocalBox):
        """
        Parameters:
        Rc : Radius of the circle placed around the origin in [m].
        LocalBox : Contains information about the boundary box

        Returns:
        area : Intersection area between a circle and a box
        """
        # New
        # the circle is split into 4 quadrants.
        # Each is checked if a box border cuts it.
            # if not, the full quadrant is added.
            # if yes, the cutQuadrantArea function is started
        area = 0.0
        for xb in [abs(LocalBox.getXmin()), abs(LocalBox.getXmax())]:
            for yb in [abs(LocalBox.getYmin()), abs(LocalBox.getYmax())]:
                # if no boundary cuts into the quadrant add full quadrant
                if xb >= Rc and yb >= Rc:
                    area += 1 / 4 * np.pi * Rc**2
                # else boundary cut into quadrant
                else:
                    area += self.CutQuadrantArea(Rc, xb, yb)
        return area

    # Computes the intersection area between a circle quadrant and a box boundary.
    def CutQuadrantArea(self, Rc, xb, yb):
        """
        Parameters:
        Rc : Radius of the cicle placed around the origin.
        xb, yb : Intersections between the box corner planes and the coordinate axes.
```

```
        Returns:
        area : Intersection area between the sphere quadrant and box corner
        """
        if Rc <= 0 or xb < 0 or yb < 0:
            ErrorMsg = "Only positive values are " +\
                       "allowed: {}.".format([Rc, xb, yb])
            self.Sentinel.error(ErrorMsg)
            raise ValueError(ErrorMsg)

        # two boundary intersect eachother = box fully inside the quadrant
        if (xb**2 + yb**2) <= Rc**2:
            return xb * yb

        # else: start with a full Quadrant
        else:
            area = 1 / 4 * np.pi * Rc**2
            # if a boundary intersects, subtract the respective part
            for B in [xb, yb]:
                if B < Rc:
                    # 360° = 2*pi
                    area -= np.pi / (2 * np.pi) * Rc**2 * np.arccos(B / Rc)
                            - B / 2 * np.sqrt(Rc**2 - B**2)
            return area

    # Compute the intersection Volume of a circular shell and a box.
    def CircShellVolume(self, Rmin, Rmax, LocalBox):
        """
        Parameters:
        Rmin, Rmax : Boundaries of the circular shell
        LocalBox : Contains information about the boundary box

        Returns:
        SArea: area of the circular shell box intersection.
        """
        # calculate the area of the inner circle
        if Rmin < 0:   # negative radius -> no inner circle
            InnerCircle = 0
            self.Sentinel.info("The inner shell had a negative"
                               " radius: {}. I set the intersection"
                               " volume to zero.".format(Rmin))
        # new
        elif Rmin == 0:
            InnerCircle = 0
        else:
            InnerCircle = self.CircleArea(Rmin, LocalBox)

        OuterCircle = self.CircleArea(Rmax, LocalBox)
        SArea = OuterCircle - InnerCircle

        return SArea
```

VIII.6.5 Code to Chapter V.1.7 – $g_2(r)$ Calculation

The following code snippet calculates the values of angle correlation function of individual datasets (`particle_file`) from the python program of Chapter III.3. The returned `compare_angle` in the parameter list is the result of $\theta(0) - \theta(r)$. For the angle correlation function, the cosin of the angle is plotted against the distance (see equation V.1-18, Chapter V.1.7).

```
def calc_orientations(particle_file):
    rawdata = pd.read_csv(particle_file, skiprows=1, sep=',',
        names=['file', 'Partikel_No.', 'y', 'x', 'minor', 'major', 'orientation', 'ratio', 'area'])
    dist_list = []
    p = rawdata
    # create list: x coordinate (in particleList[element][0]), y, distance, angle
    for P1 in range(0, len(p), 1):
        for P2 in range(P1 + 1, len(p), 1):
            d = math.sqrt((p.at[P2, 'x'] - p.at[P1, 'x'])**2 + (p.at[P2, 'y'] - p.at[P1, 'y'])**2)
            if p.at[P1, 'orientation'] > p.at[P2, 'orientation']:
                compare_angle = round((p.at[P1, 'orientation'] - p.at[P2, 'orientation']), 0)
            else:
                compare_angle = round((p.at[P2, 'orientation'] - p.at[P1, 'orientation']), 0)
            # the angles inbetween the ellipses vectors are independent of direction,
            # just angles of 0-90° are possible
            if compare_angle > 180:
                compare_angle = 360 - compare_angle
            if compare_angle > 90 and compare_angle <= 180:
                compare_angle = 180 - compare_angle
            if (p.at[P2, 'x'] - p.at[P1, 'x']) != 0:
                deg = math.degrees(math.atan((p.at[P2, 'y'] - p.at[P1, 'y']) /
                                             (p.at[P2, 'x'] - p.at[P1, 'x'])))
            else:
                deg = 90
            # orientation in polar coordinates as befor in ellipse_draw_list
            element = [p.at[P1,'Partikel_No.'], p.at[P2,'Partikel_No.'], d, deg, compare_angle]
            dist_list.append(element)
        # show progress
        if P1 % 10 == 0:
            print("Finished compare at particle {} of {}".format(P1, len(p)))
    xdist_achse = []
    compare_angle_dist = []
    for i in range(len(dist_list)):
        # write distances into drawlist
        xdist_achse.append(dist_list[i][2])
        # write angles into draw list
        compare_angle_dist.append(dist_list[i][4])
    dataframe = pd.DataFrame(xdist_achse, columns=['distance_pxl'])
    dataframe['compare_angle'] = compare_angle_dist
    binsize = 1
    dataframe['bins'] = dataframe['distance_pxl'] // binsize
    count_part_bins = dataframe['bins'].value_counts(sort=False).sort_index()
    mean_dataframe = dataframe.groupby('bins').mean()
    mean_dataframe = mean_dataframe.join(dataframe['bins'].value_counts().to_frame('counts'))
    # return list [bins, distance_pxl, compare_angle, counts]
    return mean_dataframe
```

VIII.6.5.1 Test data for Angular Evaluation

The test datasets of Chapter V.1.5 will not give much information in angular evaluation, as these test sets were all oriented in on direction. New test data with particles of different orientations are generated using e.g. open hexagonal lattice positions and randomly oriented particles or, as an example of the perpendicular orientation, a square lattice with alternating oriented ellipsoids (see Figure V-13).

Chapter IX. Acknowledgements

First of all, I want to thank my dear wife Lisa for all the sound discussions and help so that I was able to finish this biggest project in my scientific life. You helped me not just by pushing me to finalize the thesis but also with your thorough corrections. Thank you that you covered my back for all the other topics arose during this research project so I could concentrate on this. Without you, I might not have succeeded and would have not been able to persevere.

Second, I want to thank my Professor Markus Retsch who gave me the opportunity to come to his group and work on this interesting topic. You always had an open ear when questions arose. Your good mood and visionary ideas helped a lot. I am thankful that I was able to follow my own ideas within these topics but always was certain to get help from you if needed.

Next, I want to thank Hendrik Bargel for the fruitful discussions and help with the measurements of Chapter IV. It was nice to spend so much time with you at the SEM. I thank you so much for always starting the AMIRA program when it crashed again, so I could do most of the reconstruction work remotely.

I thank my research assistants and internship students. It was fun guiding you. Without you, I would not have had the opportunity to do these multiple purpose investigations and this thesis would have much fewer pages.

In that sense, I want to thank explicitly Friederike Ganster, my research assistant for many years. It was a wonderful time exploring the depth of this project with such a skilled crew-member. Thank you that you wanted to stay with me for this long time and helped me with all this cumbersome washing (particles, glass slides and lab dishes).

Then I want to thank all the members of the group of Physical Chemistry I. I enjoyed working with you. So thank you Alexander Berger, Tanja Feller, Argyrios Georgiadis, Kai Herrmann, Patrick Hummel, Ina Klein, Bernd A.F. Kopera, Tobias Lauster, Flora Lebeda, Kishin Matsumori, Anna Neuhöfer, Nelson W. Pech-May, Stefan Rettinger, Sabine Rosenfeldt, Marius Schöttle, Qimeng Song, Daewoo Suh, Maximilian Theis and Thomas Tran.

Finally, I thank my parents Andrea and Helmut. You did a good job in equipping me with enough patience to succeed in such a long-term project. I also want to thank my sisters Franziska and Desirée for being my sisters and my dear grandma Maria for being with us.

Chapter X. Eidesstattliche Versicherung

§8 Satz 3, Promotionsordnung der Fakultät für Biologie, Chemie und Geowissenschaften (BCG), Universität Bayreuth, gültig ab dem 30.03.2023.

Hiermit versichere ich eidesstattlich, dass ich die Arbeit selbstständig verfasst und keine anderen als die von mir angegebenen Quellen und Hilfsmittel benutzt habe. (vgl. Art. 97 Abs. 1 Satz 8 BayHIG).

§8 Satz 3, Promotionsordnung Fakultät BCG

Hiermit erkläre ich, dass ich die Dissertation nicht bereits zur Erlangung eines akademischen Grades eingereicht habe und dass ich nicht bereits diese oder eine gleichartige Doktorprüfung endgültig nicht bestanden habe.

§8 Satz 4, Promotionsordnung Fakultät BCG

Hiermit erkläre ich, dass ich Hilfe von gewerblichen Promotionsberatern bzw. -vermittlern oder ähnlichen Dienstleistern weder bisher in Anspruch genommen habe, noch künftig in Anspruch nehmen werde.

§8 Satz 7, Promotionsordnung Fakultät BCG

Hiermit erkläre ich mein Einverständnis, dass die elektronische Fassung der Dissertation unter Wahrung meiner Urheberrechte und des Datenschutzes einer gesonderten Überprüfung unterzogen werden kann.

§8 Satz 8, Promotionsordnung Fakultät BCG

Hiermit erkläre ich mein Einverständnis, dass bei Verdacht wissenschaftlichen Fehlverhaltens Ermittlungen durch universitätsinterne Organe der wissenschaftlichen Selbstkontrolle stattfinden können.

Aalen, 04.11.2024 ____________________
Dominik Benke, M.Sc.