Multi-View Reconstruction of Unknown Objects in the Presence of Known Occlusions

Stefan KUHN and Dominik HENRICH
Lehrstuhl für Angewandte Informatik III (Robotik und Eingebettete Systeme)
Universität Bayreuth, D-95440 Bayreuth, Germany
{Stefan.Kuhn | Dominik.Henrich}@uni-bayreuth.de

***Abstract*—We present a general method for reconstructing unknown objects (e.g. humans) within a known environment (e.g. tables, racks, robots) which usually has occlusions. These occlusions have to be considered since parts of the unknown objects might be hidden in some or even all camera views. Besides grayscale and color cameras also depth sensors are considered. In order to avoid cluttered reconstructions, plausibility checks are used to eliminate reconstruction artifacts which actually do not contain any unknown object. One application is a supervision/surveillance system for safe human/robot-coexistence and –cooperation. Experiments for a voxel-based implementation are given.**

Keywords – Inferred Visual Hull; Multi-View Reconstruction; Occlusions; Visibility; Static Environment; Dynamic Environment; Computer Vision; Color cameras; Depth sensors; 3D Scene Analysis

I. INTRODUCTION

The *visual hull* [14] of an object is defined as the maximal silhouette-consistent volume in the limit of an infinite number of cameras. The visual hull for a finite number of cameras is called *inferred* visual hull [17]. It can be described as intersection of the general cones obtained by the back projection of the object's silhouettes of all camera images in a calibrated camera network [16]. If the object is completely visible in each camera, the inferred visual hull always contains the overall object. Several methods for constructing the inferred visual hull have been published in the past. Therefore different representations like voxels [4], [13], [18] or polyhedrons [15], [7] have been used. The main focus has been efficiency, accuracy and robustness.

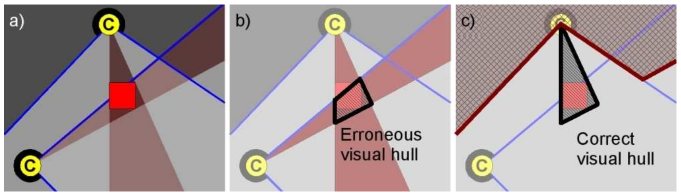

Figure 1 Illustration of a scene with two cameras and an object (a), the erroneous inferred visual hull construction, when only considering the object cones (b) and the correct visual hull, using the complement of the free space [7].

In the presence of occlusions, the silhouette might be erroneous in some cameras, such that the inferred visual hull does not contain the overall object. The same problem arises when the object resides partly outside the camera frustum (Figure 1b). Background subtraction algorithms (e.g. [5] and [8]) will produce such erroneous silhouettes if the object is partly or completely occluded. To overcome this problem, occlusions must be considered. Recently, some approaches have been published, which deal with two different kinds of occlusion problems.

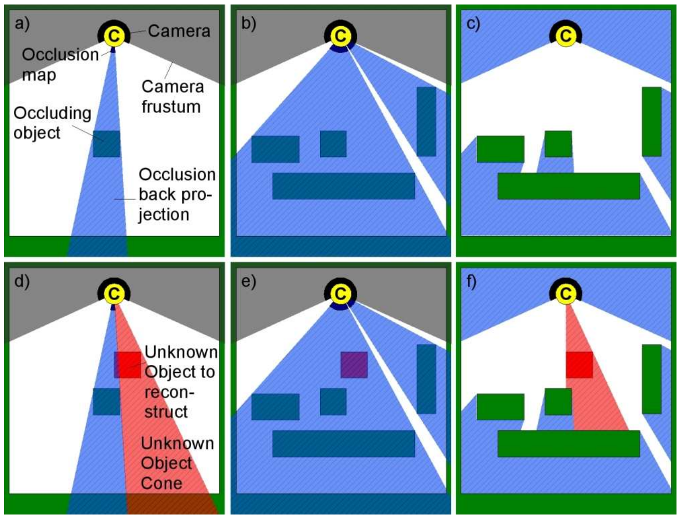

Figure 2 Illustration of the occlusions in a scene with one camera using an occlusion map (a) and (b) and the resulting detection capabillities (d) and (e). In Figure (c) the visibility depth of the known environment is utilized as described in our approach. The detection capability of an object is illustrated in (f). The region outside the camera frustum is additionaly interpreted as occlusion. All light gray, light blue and light red regions may contain objects and have to be interpreted as reconstruction here.

The first occlusion problem is caused by the camera frustum. If the object resides outside this frustum in at least one camera, the inferred visual hull will not contain the overall object. This problem has been tackled by [7]. In principle, they do not calculate the intersection of object cones, but the complement of the fused free space (Figure 1).

The second occlusion problem is caused by an environment which might occlude the objects to be reconstructed in some camera views. In [3] and [12] dynamic known objects like robots are considered. A masking of these in the images is proposed. Later this method was applied to static occluding objects [9], [13]. While [3] calculates the occlusion map online based on the known geometry and configuration of a robot, [9] determines a static occlusion map by a learning method. Having static objects, this occlusion map has the disadvantage, that a whole cone is classified as occlusion, although the camera could see up to the occluding objects surface (Figure 2 a, b and d, e). Using several cameras, as described in [10] and [11], this effectively results in an inferred visual hull of the static objects (Figure 3 a). In the case of several static objects, also pseudo objects may arise (Figure 3 b). There is neither information about the geometry of the static objects nor about the free space within the inferred visual hulls. Since unknown objects which finally have to be reconstructed may reside within these static object hulls, the inferred visual hulls have to be added to the reconstruction. The observation of a scene containing active objects can reveal the inferred visual hulls of the occluding objects. [11] provides a probability of an occlusion for a voxel in a given view, while [10] reconstructs the occluding object. As said above – the best approximation of a static object is its inferred visual hull, resulting in the same issue as the pure occlusion maps.

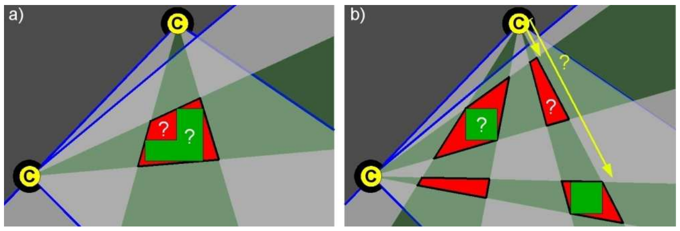

Figure 3 Illustration of the inferred visual hull of the learned static environment. We do not have information about the actual geometry and the free space within the inferred visual hull. Thus, we have to assume unknown objects which have to be reconstructed within this inferred visual hull. Furthermore, we do not know the actual depth a camera can see at a specified pixel (b).

Compared to the learning approach of the static occluding objects, we consider here the occlusion problem from a slightly different view. We already know the complete environment containing static objects (e.g. tables, racks, etc.) and dynamic objects (e.g. robots) except the unknown dynamic objects we want to reconstruct (e.g. human) (Figure 2 c, f). Thus, the question is how to obtain the best reconstruction of these unknown objects using different kinds of sensors (color cameras, depth sensors), different kinds of objects (known static and dynamic) and different kinds of background subtraction approaches. Furthermore, what limitations exist having these different conditions and how to eliminate reconstruction artifacts which actually do not contain any object?

We provide an analysis of the above questions and a general model of the occlusions and visibility. Furthermore, we present a voxel-based algorithm and experiments regarding computation times.

One main application are surveillance systems for safe human/robot coexistence and cooperation. The combination of the strength of both human and robot promises a higher efficiency and flexibility in production. In the past, humans and robots were strictly separated by fences and other safety equipment. Recently, engineer standards were extended for allowing vision systems under certain conditions. In short, the human has to be safely detected by the vision system and the robot speed is restricted.

The remainder of the paper is organized as follows. In Section II, the method to cope with occlusions is described in a general fashion. A multitude of alternatives is discussed, e.g. the use of depth images instead of usual color images. In Section III, a reconstruction algorithm is described and experiments in Section IV provide results regarding computation times. Section V closes with a conclusion and an outlook to future investigations.

II. Occlusions and Visibility

This section comprises a theoretical look at the problem of occlusions and visibility in a multi-camera setup having different pre-conditions, and the possibilities to perform a reconstruction of the objects using plausibility checks to revise pseudo objects, i.e. reconstruction artifacts which actually do not contain an object. In short, the aim is to reconstruct as accurate as possible the unknown but detectable parts (e.g. humans) within a known environment (e.g. tables, robot), which typically has occlusions.

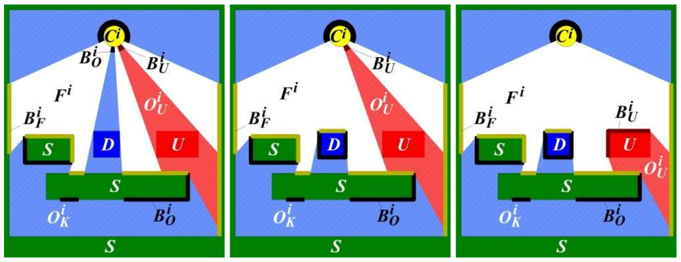

Figure 4 Illustration of the differences regarding visibility and occlusions using color or grayscale cameras with common change detection methods (a), extended change detection methods (b), or using a depth camera, with a suitable change detection method (c). Nomenclature: C^i: Camera *i*; F^i: Free in camera *i*; *S*: Static known object; *D*: Dynamic known object; *U*: Unknown object; O^i_K: Known occlusion in *i*; O^i_U: Unknown occlusion in *i*; B^i_F: Free boundary in *i*; B^i_O: Occlusion boundary in *i*; B^i_U: Unknown boundary in *i*.

At first, the types of objects that are contained by surveyed scenes are described in Subsection A. Two different kinds of cameras are used to detect the introduced type of objects which is referred as *unknown* (e.g. humans and objects with initial unknown position, orientation, etc.). The camera types, its detection capabilities and the different occlusions caused by the different objects are detailed in Subsection B. The simultaneous use of several cameras with different perspectives onto the surveyed scene which results in a cluttered reconstruction is described in Subsection C. The plausibility checks to revise this cluttered reconstruction are discussed in Subsection D.

A. Object Types

The surveyed scene contains static and dynamic objects. *Static* objects *S* are racks, tables etc. The geometry, position and the appearance of those objects are *known* and do not change over the time (apart from possibly occurring shadows and from illumination changes caused by the dynamic objects). *Dynamic* objects are robots, conveyor belts, humans etc. This group must be divided into two subgroups. The first subgroup contains *known* dynamic objects *D*, with changing geometry, position and the appearance but in a known manner. The robots and conveyor belts pertain to this subgroup. The second subgroup contains dynamic objects U with *unknown* changing geometry, position and appearance, e.g. humans. In the majority of cases approximate information about size, volume or similar can be provided. TABLE I. summarizes the identified object types. Note that static unknown objects do not exist. The *free space* F of a surveyed scene does not contain any known object but may contain unknown objects. Figure 4 a-c illustrate the introduced object types. In summary, the following three equations hold true:

$S \cup D \cup F = \mathbf{E}^n$, with $\mathbf{E}^n$: Euclidian Space

$S \cap D = S \cap F = D \cap F = \varnothing$ and $U \subseteq F$

TABLE I. OBJECT TYPES AND EXAMPLES WITH ABBREVIATION

	Static	Dynamic
Known	Tables, Racks, … (Abbr.: S)	Robots, Conveyor belts, … (Abbr.: D)
Unknown	----	Humans, … (Abbr.: U)

B. *Camera Types and Detection Capabilities*

A number of N calibrated cameras with focal points $C^i \in \mathbf{E}^n$, $i \in \{1, \ldots, N\}$ and a frustum $L^i = \{x \in \mathbf{E}^n |\ x$ is projected via C^i onto the image plane of camera $i\}$, are used to detect and finally reconstruct the unknown objects which reside in-between the known objects as accurate as possible. Therefore, two types of cameras are considered:

- *Color and Grayscale cameras*: The images of these sensors contain color and intensity information for each pixel, respectively.
- *Depth cameras*: The images contain depth information for each pixel (for example stereo vision or time-of-flight cameras).

One fundamental characteristic of these sensors is that they can only see up to the surface of the nearest opaque object per viewing direction (e.g. pixel center direction). Thus, occlusions always occur at the rear side of an opaque object. Moreover, the visibility of these sensors is limited by the frustum L^i. Outside this frustum, the sensor is not able to see anything. Thus, these parts need to be interpreted as occlusions as well.

Now, the terms *visibility* and *occlusions* have to be introduced and detailed (Figure 4). The *visibility* V^i of a camera i is the region of the free space F where a camera is able to detect unknown objects. The *known occlusion* O_K^i of a camera i is the region of the free space F where a camera can not detect unknown objects due to occlusions caused by known objects. Thus, it can be stated that $O_K^i \cap V^i = \varnothing$ and $O_K^i \cup V^i = F$ for each camera i. The *unknown occlusion* O_U^i of a camera i is the region of the visible space V^i where an unknown object has to be assumed, due to the evaluation of a camera image by a background subtraction method. The free space seen by camera i F^i is defined by $F^i = V^i \backslash O_U^i$. It can be stated, that $O_K^i \cap O_U^i = O_K^i \cap F^i = O_U^i \cap F^i = \varnothing$ and $O_K^i \cup O_U^i \cup F^i = F$ for each camera i. The concrete structures of the sets V^i, O_K^i, O_U^i depend directly on the used camera type with its detection capabilities and are described later.

In order to describe our formalism, we use the concepts of rays and segments in the Euclidean Space. A ray(S, E) and a segm(S, E) are point sets and are defined by

$$\text{ray}(S,E) := \{S + t\cdot(E-S) \mid t \in \mathbf{R}_0^+, S, E \in \mathbf{E}^n\}$$

$$\text{segm}(S,E) := \{S + t\cdot(E-S) \mid t \in \mathbf{R} \wedge 0 \le t \le 1,\quad S, E \in \mathbf{E}^n\}$$

Furthermore, a distance function dist(P, Q), with $P, Q \in \mathbf{E}^n$ exists, since the Euclidean Space is a metric space. Having the visibility and the occlusions, sets containing the most distant visible points and the nearest occluded points (per viewing direction) can be specified by

$$B_F^i = \{x \in V^i \mid \text{dist}(C^i, x) = \max_{y \in V^i \cap \text{ray}(C^i,x)} \{\text{dist}(C^i, y)\}$$

$$B_O^i = \{x \in O_K^i \mid \text{dist}(C^i, x) = \min_{y \in O_K^i \cap \text{ray}(C^i,x)} \{\text{dist}(C^i, y)\}$$

$$B_U^i = \{x \in O_U^i \mid \text{dist}(C^i, x) = \min_{y \in O_U^i \cap \text{ray}(C^i,x)} \{\text{dist}(C^i, y)\}$$

Dependent on the camera type, different change detection methods have to be used to detect the unknown objects. In the following, the concrete visibility and occlusions of a camera i depending on its type and a suitable change detection method are formalized. Here, we distinguish between color/grayscale cameras in conjunction with a conventional background subtraction (1) and an improved background subtraction method (2) and depth images (3).

1) *Basic Color/Grayscale Images*

Using color or grayscale cameras, ordinary change detection methods can be utilized. These change detection methods segment an image into foreground and background based on the known appearance and a current image of the surveyed scene. If an unknown object resides in the scene and is not occluded by the known environment, it is marked as foreground in the segmented image. But usually the dynamic known objects are also – if not occluded – identified as foreground in the segmented image. Thus, the detection of unknown objects *in front* of a dynamic known object is not possible. Since the change detection method is not able to decide whether the cone between the camera and the dynamic known object is free or contains unknown objects, it has to be interpreted as a known occlusion. Thus, the visibility is described by

$$\begin{aligned} V^i = \{x \in L^i \mid\ & \text{segm}(C^i, x) \subseteq F \wedge \\ & (\text{ray}(C^i, x) \cap D = \varnothing \vee \\ & (\text{ray}(C^i, x) \cap D \neq \varnothing \wedge \text{ray}(C^i, x) \cap S \neq \varnothing \wedge \\ & \exists_{y \in S \cap \text{ray}(C^i,x)} \forall_{z \in D \cap \text{ray}(C^i,x)} \text{dist}(C^i, y) < \text{dist}(C^i, z)))\} \end{aligned}$$

As detailed above, the known occlusions are formulated by

$$O_K^i = F \setminus V^i$$

Since depth values are not available for unknown objects with this sensor type, unknown occlusions start at the camera:

$$O_U^i = \{x \in V^i \mid \text{ray}(C^i, x) \cap U \cap V^i \neq \varnothing\}$$

Figure 4 a illustrates the occlusions using a color or grayscale camera and a conventional background subtraction method.

2) *Improved Color/Grayscale Images*

In the second case, an improved change detection method is able to detect unknown objects even in front of a dynamic object. Two simple approaches come to mind:

First, a database which contains several appearances of different configurations and positions of the dynamic known objects as images can be used to project the dynamic known objects into the current image before using it within an ordinary change detection method [2]. The disadvantage of this method is the vast amount of needed memory to store the images if the dynamic object has many configurations like an industrial robot.

Second, the configuration, position and texture of the dynamic known object can be rendered into the current image before using it within the ordinary change detection. Two issues can be discovered. First, the rendered dynamic known object probably causes a classification into foreground, since it is not exact enough. Second, dependent on the geometry complexity of the known dynamic objects, it might take too much time to render the dynamic known object.

The visibility and occlusions of the extended background subtraction method are described by:

$$V^i = \{x \in L^i \mid \text{segm}(C^i, x) \subseteq F\}$$

$$O_K^i = F \setminus V^i$$

The unknown occlusions $O_U{}^i$ are the same as in the previous case, using a conventional background subtraction method.

Figure 4 b illustrates the visibility and occlusions using a color or grayscale camera and an extended change detection method. (The known occlusion cone in front of the known dynamic object in Section I now can be classified as visible.)

Here we consider only the consequences of using those two different background subtraction methods. A further discussion of these methods is outside the focus of the paper.

3) *Depth Images*

Background subtraction methods using depth cameras could compare the known depth with a current depth for each pixel to determine not-occluded unknown objects [6]. Besides foreground/background segmentation, the result additionally contains depth information. Thus, a free cone between the camera and a not-occluded unknown object can be guaranteed. Also, a free cone between the camera and a not-occluded known dynamic object can be guaranteed (Figure 4 c). In comparison to the improved background subtraction for color/grayscale cameras, only the definition of the unknown occlusions changes:

$$O_U^i = \{x \in V^i \mid \text{segm}(C^i, x) \cap U \cap V^i \neq \varnothing\}$$

C. *Simultaneous Use of Several Cameras*

Using several cameras with different perspectives onto the surveyed scene, each camera, regardless the type that is used provides a different occlusion and visibility situation, as discovered in the previous section that now has to be merged.

For every camera *i*, a collection of sets can be provided describing the known and unknown occlusions as well as the surveyed free space.

$$Q^i = \{O_K^i, O_U^i, F^i\}$$

The partitioning of the free space via the reconstruction step can be described by

$$R = \{A = q^1 \cap \ldots \cap q^N \mid q^i \in Q^i \wedge$$
$$\forall x, y \in A : \exists f : [0,1] \to A, f(0) = x \wedge f(1) = y\}$$

In words, all different labeled regions of all cameras are intersected among each other. Furthermore, the resulting intersections are grouped into connected components. All these connected components are contained by R (Figure 5, upper row). Again, it can be stated that $\cup_{A \in R} A = F$.

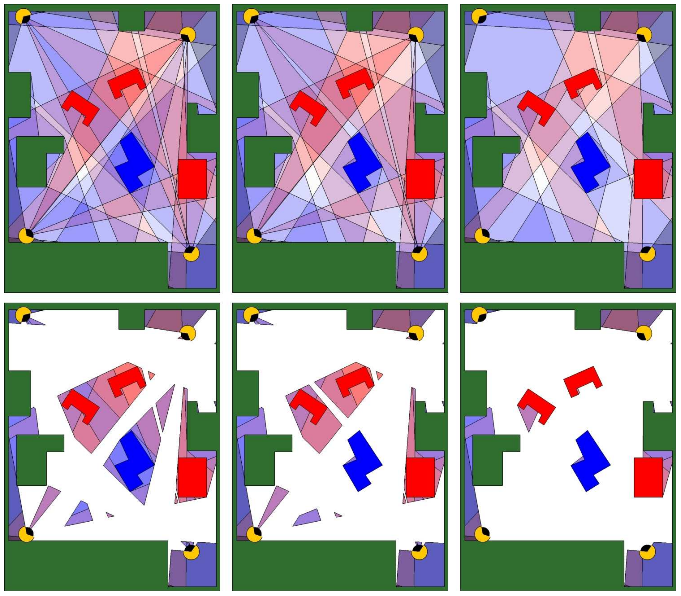

Figure 5 Illustration of the use of several cameras. In the first row, the occlusion situation is simply overlaid. The second row, regions are set to free, if at least one camera sees free. In the first column, color/grayscale cameras and a simple change detection method are used. In the second column, color/grayscale cameras and an extended change detection method are used. In the last column, depth cameras are used.

Volumes in the free space of the surveyed scene which are actually seen as free (cf. F^i) by at least one camera are not further considered, since no unknown object can reside there. This results in (Figure 5, second row):

$$R' = \{x \in R \mid \forall i = \{0, \ldots, N\} : x \notin F^i \wedge x \neq \varnothing\}$$

To each set of R' a tuple (o, u) can be assigned, containing the number of seen known occlusions and unknown occlusions.

The number of known and unknown occlusions is represented by the mixture of the colors red and blue in Figure 5.

D. *Plausibility Checks*

In the majority of applications some information like size, volume, etc. about the unknown objects is available. Several sets of *R'* actually cannot contain an unknown object. Thus, plausibility checks are used to eliminate those occlusions which do not contain unknown objects. The mentioned plausibility checks aim for a *quasi-static* consideration. Another kind of plausibility checks can utilize *temporal* considerations, like “an unknown object can not suddenly appear in and surrounded by free space”.

Note plausibility checks can apply to the whole scene or only to a part of the scene. In the following, a couple of quasi-static plausibility checks are discussed.

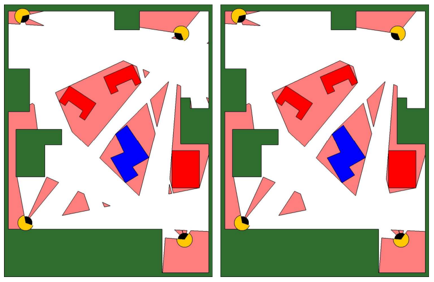

Figure 6 Illustration of the *Minimum Volume* plausibility check for color/grayscale with conventional change detection. Left: Before removing small regions; Right: After removing small regions.

Minimum Volume: If only unknown objects like humans with a typical volume of 0.075 m³ should be detected, one might set a maximum volume for occlusions to be eliminated to 0.05 m³. Thus, all connected sets of *R'* obtained as described in the previous section with a volume smaller than 0.05 m³ can be safely removed. Only unknown objects with a specified minimum volume remain (Figure 6). Note, a maximum volume plausibility check is obviously inapplicable, since larger occlusions may contain smaller unknown objects.

Maximum Distance to Ground: Typically, objects do not hover but have contact with the ground. If this can be guaranteed, all connected sets of *R'* with no contact to the ground *S* or *D* can be eliminated. More general, all objects with a distance larger than a specified maximum distance to the ground can be eliminated. Thus, setting the maximum distance to 1 m also a jumping human can be detected and is not removed by this plausibility check.

Surveillance Zones: In most cases, only certain parts of the whole surveyed scene are actually interesting so that unknown objects outside this part can be eliminated.

Occlusion parameter θ: If it can be guaranteed, that an unknown object can be completely occluded by the known environment in a maximum number of θ cameras, all regions where more than θ cameras see a known occlusion can be eliminated if no other region is connected to it with equal or less than θ cameras which see a known occlusion. As shown in Figure 7 the mixed occlusions are binarized by θ. Afterwards, connected regions are adapted. The identified remaining occlusions which do not contain an object can be safely removed. For more details about θ, see [12].

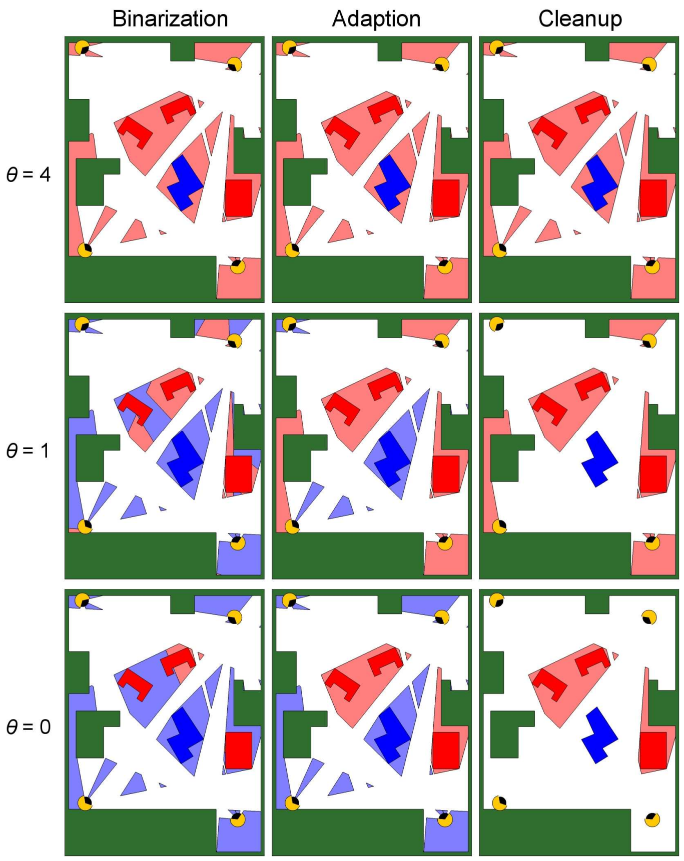

Figure 7 Binarisation, adaption and cleanup for the three different thetas

III. Reconstruction Algorithm

In this section we provide a voxel-based reconstruction algorithm, which works on the surfaces of the objects and is capable to deal with occlusions. The algorithm presented here is limited to color and grayscale cameras in combination with a conventional background subtraction approach like [5] and [8]. Regarding the camera model, we only assume a two-dimensional field of connected pixels, with their back projected volumes also connected. Furthermore, the position and geometry of the pixels and the back projected volumes have to be known. Thus, we assume neither a pinhole camera model nor undistorted images.

A. *Surface voxel determination*

Given several calibrated cameras images segmented into *free*, *known* and *unknown* and a voxel space, surface voxels can be determined by the following algorithm. Then the result is a voxel space with voxels marked according to the occlusions of all perspectives and a list containing all these voxels. At first, the needed functions are explained.

The classification value (*free, known* or *unknown*) of a pixel *P* is provided by the function *classification (P)*.

Assuming that two adjacent pixels are separated by a pixel edge *E*, a list of voxels that are intersected by the back projection of this pixel edge E down to its visibility depth is provided by the function *voxelList(E)*. The function *neighborClassification(E)* for a pixel edge E provides the value *unknown*, if one of the two pixels is classified as *unknown*. It provides *known*, if one pixel is classified as *known* and the other as *free*. In all other cases, it provides *free*.

For each voxel, the pixels it projects to in all cameras are needed. For simplification, here we only use the center of the voxels with the consequence, that objects that are smaller than a voxel side may be reconstructed incorrectly. Thus, it is necessary to choose an appropriate small voxel size. (Another voxel-like but camera centric-representation called conexels [1] could be applied, which avoids this drawback). The pixel of the projection of the voxel center V into a camera image C is provided by the function *projectVoxelCenter(V, C)*.

The distance for a voxel center V to a camera C is provided by the function *distance(V, C)*.

Per pixel P, the visibility depth (distance to B^i_F) and the occlusion depth (distance to B^i_O) as described in Section II is provided by the functions *visibleDepth(P)* and *occlusionDepth(P)*.

The function *markVoxelAndAddToList(V, O_K, O_U)* marks the voxel V in voxel space by the two counter variables O_K, O_U representing the number of known and unknown occlusions respectively, and adds it to a list containing all surface voxels.

```
foreach camera $C$ do
  foreach silhouette pixel edge $E$ do
    foreach voxel $V$ in voxelList($E$) do
      counter $O_U$ = 0, $F$ = 0, $O_K$ = 0
      if neighborClassification($E$) == unknown
        $O_U$ = 1
      else if neighborClassification($E$) == known
        $O_K$ = 1
      endif
      foreach camera $C'$ != $C$ do
        pixel P = projectVoxelCenter($V$, $C'$)
        if classification($P$) == known
           or distance($V$, $C'$) $\geq$ occlusionDepth($P$)
          $O_K$++
        else if classification($P$) == unknown
           and distance($V$, $C'$) $\leq$ visibleDepth($P$)
          $O_U$++
        else
          $F$++
        endif
      done
      if $F$ == 0
        markVoxelAndAddToList($V$, $O_K$, $O_U$)
      endif
    done
  done
done
```

As described above, after the execution, a voxel list containing the determined surface voxels is available. Additionally, these voxels are marked in voxel space by the tuple (O_K, O_U).

Since the surface is not necessarily closed at the known objects, one may use a constrained flood fill algorithm to close it. Furthermore, completely occluded regions exclusively caused by the static environment are not revealed by this algorithm but can be determined in an initialization step by testing each voxel for visibility against the static environment in all cameras.

Having the surface voxels, partitions of related voxels, i.e. voxels with the same known and unknown occlusion counter can be built. Then, the sorted plausibility checks can be applied according to the costs and success probability. Dependent on the plausibility check additional information, like volume has to be calculated.

B. *Memory Consumption*

Some of the used functions can be implemented as look-up tables to enable fast calculations. In order to give a memory consumption estimation M of these look-up tables, the following variables are introduced: A voxel space with dimensions X, Y and Z is used and the resolution of N cameras is provided by W and H.

The visibility depth and occlusion depth per pixel has a memory consumption for all images of:

$$M_1 = 2 \cdot N \cdot W \cdot H$$

The memory consumption for the voxel lists per pixel edges and for all cameras can be estimated by:

$$\begin{aligned} M_2 \leq N \cdot [\quad & ((H + W + 2)) \cdot E \\ + & ((W + 1) \cdot (H - 1) + (W - 1) \cdot (H + 1)) \cdot G\ ], \\ & \text{with } G = X + Y + Z \text{ and } E = Z \cdot Y + Z \cdot X, \quad X \leq Y \leq Z \end{aligned}$$

Furthermore, the distances for each voxel to all cameras results in a memory consumption of:

$$M_3 = 2 \cdot N \cdot X \cdot Y \cdot Z$$

Thus, the overall memory consumption is bounded by $M \leq M_1 + M_2 + M_3$. As an example the parameters are set to $N = 4$, $W = 320$, $H = 240$ and $X = Y = Z = 100$, with a typical camera placement and voxel-, pixel-addresses and floating point variables of 4 bytes, results in an upper bound of $M \leq 907$ MB and actually of 411 MB.

IV. Experiments

In order to evaluate our methods and algorithms, we set up two test environments. The first one is an industrial robot work cell (Figure 8, left) with eight color cameras and four depth sensors mounted at the ceiling. Each camera has its own computing unit for preprocessing steps. The second environment (Figure 8, middle) is a smaller one, with five color cameras mounted around the scene to survey. Only one computer is used here. The latter one is available in a virtual simulation environment, too (Figure 8, right).

Figure 8 Two test environments: Left: Robot work cell with eight color cameras and four depth cameras mounted at the ceiling; Middle: Small demonstrator with five color cameras. Right: Virtual test environment: Five calibrated cameras; Yellow wire frame: Surveillance zone; Light blue: Static known environment; Red Sphere: Virtual unknown object.

In the following, the hardware and software configuration of the second test environment is listed. Then the performance results of the reconstruction step are presented having already segmented images within the virtual environment of the second test environment.

A. Hardware and Software Configuration

The used computer contains an Intel Core™2 Quad CPU, with 2 GHz, 6 MB Cache and 4 GB RAM, but currently only one core of the CPU is utilized by our implementation. The graphics card is an NVIDIA GeForce 9600 GT with 512 MB and it is CUDA enabled. The operating system is a SUSE 11.0, with the gcc/g++ compiler suite version 4.3.1. The cubical volume of the test environment is 76 cm × 76 cm × 76 cm. Five Unibrain FireWire Fire-i™ Digital Board Cameras with 15 and 30 fps and a resolution of 640x480 Bayer Pattern are used.

The calibration results, obtained by [16] for the images with a resolution of 640x480 have a low 3D position back projection error (mean deviation < 1.6 pixels and standard deviation < 1 pixel).

The following performance tests for the reconstruction uses the virtual test environment, based on the real test environment providing a virtual object (here: sphere with a radius of 6 cm) and its segmented camera images. Additionally, a surveillance zone and a static object are included (Figure 8, right).

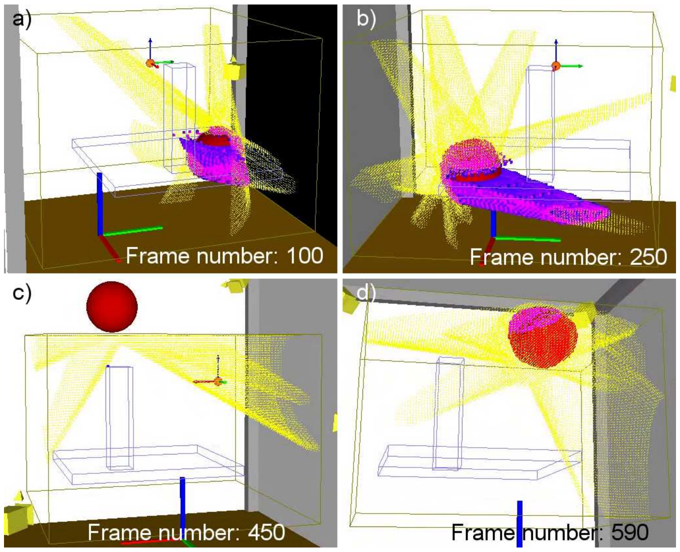

Figure 9 Images of the simulated environment of four different frames of the recorded sequence. Here, the static object is illustrated by its wire frame. The yellow dots represent the voxels which have been tested for being surface voxels within a predefined surveillance zone. The surface voxels are shown using a mixture of red and blue dependent on the visibility.

B. Performance tests

The unknown object in the virtual test environment is moved on a circular path around and through the static known object in the middle of the scene. The virtual object is projected into all camera images simulating a conventional background subtraction. The segmented images are used to reconstruct the voxel-based unknown object within the predefined surveillance zone and in consideration of the occlusions. Two cycles of this movement with a total of 1200 frames have been recorded. Figure 9 illustrates four interesting frames of the recorded sequence. The yellow dots represent voxels which have been tested for being surface voxels. The resulting surface voxels are shown using a mixture of the colors red and blue, dependent on the visibility.

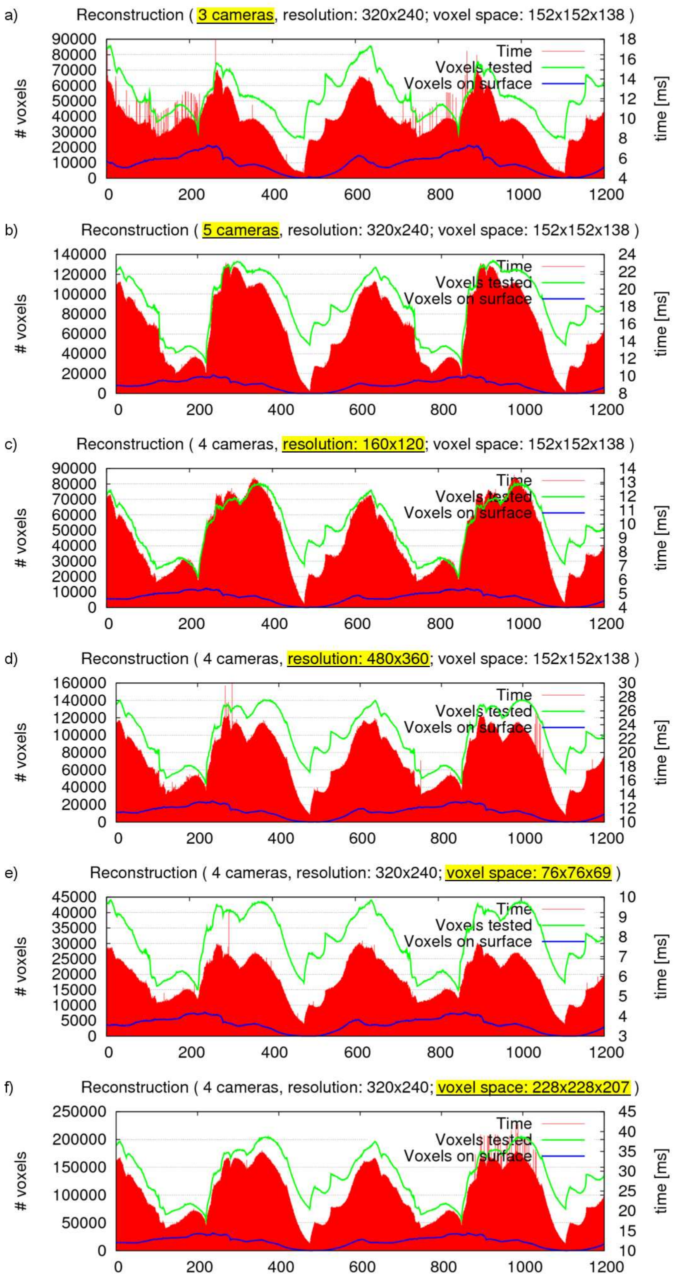

Figure 10 Pairwise comparison of different settings regarding number of cameras (a),(b), number of pixels (c),(d) and number of voxels (e),(f). Green line: Number of voxels tested. Blue: Number of surface voxels tested. Red area: Needed time in milliseconds (right scale).

Dependent on the position of the unknown object, different numbers of pixels and voxels are marked and thus, different computation times are needed.

The diagrams, depicted in Figure 10, compare the number of potential surface voxels (green line), actual surface voxels (blue line) and the time (red area) consumed for different numbers of cameras (a, b), different camera resolutions (c, d) and different voxel space resolutions (e, f). Obviously, the calculation time corresponds to the number of potential surface voxels which have to be tested for each camera. Furthermore, the number of actual surface voxels must always be smaller or equal to the number of potential surface voxels.

The calculation time is high, if the unknown object is seen by all cameras, such that many potential voxels have to be tested (frame# 250). Although the unknown object may be outside the surveillance zone, potential surface voxels have to be tested because of the absent depth information of this unknown object. Only the number of actual surface voxels is zero (frame# 450).

In Figure 9 b the lower part of the sphere is only seen by the rightmost camera. Thus, the complete cone of potential surface voxels within the surveillance zone caused by that camera actually results in surface voxels. In this case, the ratio between actual surface voxels and potential surface voxels is relatively high.

TABLE II. summarizes the measured computation times by comparing the average values of the diagram pairs. The quintessence of this diagram is that although multiplying the number of pixels or voxels by a factor, the average time increases slower. This behavior is due to the consideration of surfaces and silhouettes instead of volumes and areas, respectively.

TABLE II. SUMMARY AND ANALYSIS OF THE RECORDED SEQUENCE

	# cameras	**Avg. number of potential surface voxel tested**	**Avg. number of actual surface voxel tested**	**Avg. time [ms]**
b	5	89021	8281	15.74
a	3	52488	9168	9.88
b/a	1.667	1.696	0.903	1.593
	# pixel			
d	172800	99095	11536	19.06
c	19200	51821	6166	9.01
d/c	9	1.91	1.87	2.11
	# voxel			
f	10760688	134968	15717	24.43
e	398544	31175	3674	5.9
f/e	27	4.33	4.28	4.14

V. CONCLUSIONS

For the first time, a general and consistent formalism for describing the visibility and occlusions within a camera surveyed scene with a known environment is provided. To do so, objects are classified as known/unknown and static/dynamic. Furthermore, this formalism abstracts from the camera type and the background subtraction method. Thus, depth sensors or grayscale/color cameras can both be used or even mixed.

A voxel-based algorithm reconstructing the unknown objects, which works on surfaces using grayscale/color cameras in combination with a conventional background subtraction method, has been presented. The experimental results show that the computation time for the reconstruction step depends mainly on the number of tested surface voxels. Additionally, the measurements show that the computation time increases slower than the camera resolution and voxel space resolution, due to the surface and silhouette consideration.

In the future, the plausibility checks especially the temporal ones will be considered more intensively, since these promises a valuable enhancement in the reconstruction of unknown objects. The plausibility checks will be integrated into the voxel-based algorithm. Furthermore, the presented algorithm can be parallelized, such that potential surface voxels are tested simultaneously. For this, NVIDIAs CUDA seems to be suited. In addition, non-voxel-based approaches will be investigated.

ACKNOWLEDGMENT

This work has been supported by the German Research Foundation (DFG) under the project name “Sicherheitsstrategien für die Mensch/Roboter-Koexistenz und -Kooperation” (SIMERO).

REFERENCES

[1] J. Casas and J. Salvador, “Image-Based Multi-view Scene Analysis using ‘Conexels’”, ACM Proceedings of the HCSNet workshop on use of vision in human-computer interaction, Vol. 56, Canberra, Australia, 2006.

[2] Daimler AG, A. Franke, L. Krüger and C. Wöhler, „Kamerabasierte Überwachung bewegter Maschinen und/oder beweglichen Maschinenelementen zur Kollisionsverhinderung“, published patent application, DE 10 2006 048 163 A1 2008.02.07, Germany, 2008.

[3] D. Ebert and D. Henrich, “Safe Human-Robot-Cooperation: Image-based Collision Detection for Industrial Robots”, In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Lausanne, 2002.

[4] F. Cailette and T. Howard, “Real-Time Markerless Human Body Tracking with Multi-View 3-D Voxel Reconstruction”, In Proc. British Machine Vision Conference (BMVC), pp. 597-606, Oxford, 2004.

[5] A. Elgammal, D. Harwood and L. Davis, “Non-parametric Model for Background Subtraction”, Computer Vision – ECCV, Vol. 1843, 2000.

[6] M. Fischer and D. Henrich, “3D Collision Detection for Industrial Robots and Unknown Obstacles using Multiple Depth Images”, In: German Workshop on Robotics (GWR), Braunschweig, Germany, 2009.

[7] J.-S. Franco and E. Boyer, “Efficient polyhedral modeling from silhouettes”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 31, Is. 3, pp. 414-427, 2009.

[8] S. Fukui, Y. Iwahori and R.-J. Woodham, “GPU Based Extraction of Moving Objects without Shadows under Intensity Changes”, IEEE Congress on Evolutionary Computation. (IEEE World Congress on Computational Intelligence), Hong Kong, 2008.

[9] L. Guan et al. “Visual Hull Construction in the Presence of Partial Occlusions”, In Proc, of the 3rd International Symposium on 3D Data, 2006.

[10] L. Guan, J.-S. Franco and M. Pollefeys, “3D Occlusion Inference from Silhouette Cues”, In Proc. Comp. Vis. And Pattern Rec. (CVPR), 2007.

[11] M. Keck and J.W. Davis, “3D Occlusion Recovery using Few Cameras”, In: Conf. on Computer Vision and Pattern Recognition (CVPR), 2008.

[12] S. Kuhn, T. Gecks and D. Henrich, "Velocity control for safe robot guidance based on fused vision and force/torque data", In: IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems", Heidelberg, Germany, 2006.

[13] A. Ladikos, S. Benhimane and Nassir Navab, "Efficient Visual Hull Computation for Real-Time 3D Reconstruction using CUDA", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Anchorage, Alaska (USA), June 2008.

[14] A. Laurentini, "The Visual Hull Concept for Silhouette-Based Image Understanding" IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 16, pp. 150-162, 1994.

[15] S. Lazebnik, Y. Furukawa and J. Ponce, "Projective Visual Hulls", International Journal of Computer Vision, Vol. 74, Nr. 2, August 2007.

[16] T. Svoboda, D. Martinec and Tomas Pajdla, "A Convenient Multi-Camera Self-Calibration for Virtual Environments", In: Presence: Teleoperators and Virtual Environments, Vol 14, Issue. 4, 2005.

[17] G. Slabaugh, B. Culbertson, T. Malzbender, and R. Shafer, "A survey of methods for volumetric scene reconstruction from photographs", In Intl. WS on Volume Graphics, 2001.

[18] R. Szeliski, "Rapid Octree Construction from Image Sequences", CVGIP: Image Understanding, 58(1):23-32, 1993.