In: Third ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 2009), Aug 30 – Sep 02, 2009, Como/Italy

Surveillance of Robots using Multiple Colour or Depth Cameras with Distributed Processing

Markus FISCHER and Dominik HENRICH
Lehrstuhl Angewandte Informatik III (Robotik und Eingebettete Systeme)
Universität Bayreuth D-95440 Bayreuth
{Markus.Fischer | Dominik.Henrich}@uni-bayreuth.de

***Abstract*—We introduce a general approach for surveillance of robots using depth images of multiple distributed smart cameras which can be either standard colour cameras or depth cameras. Unknown objects intruding the robot workcell are detected in the camera images and the minimum distance between the robot and all these detected unknown objects is calculated. The surveillance system is built as master-slave architecture with one slave per distributed camera. The level of distributed processing on the slaves (from pure image acquisition up to minimum distance calculation) controls the remaining computations on the master and thus the quality of approximation of the detected unknown objects. The calculated minimum distance and the asymptotic overall surveillance cycle time are evaluated in experiments.**

Keywords - surveillance; collision detection; multi-view reconstruction; depth cameras; colour cameras

I. INTRODUCTION

In todays industrial applications, human and robot workspaces are completely separate and it is forbidden for them to share a common workspace. A robot without surveillance of its workspace is unaware of unexpected changes of its environment and can not react properly, e. g. a human entering the robot workspace and crossing the robots trajectory. If the robot proceeds its program, it could collide with the human leading to severe injury or death. Thus, the robot workspace must be surveilled to detect unknown objects in it. The robot velocity must be controlled using the minimum distance between the robot and any detected unknown object in the robot workspace leading to slow down or stop the robot, if it approaches an unknown object. There are many applications for coexistence of robot and human. Robots would require less space if no physical barriers are needed to isolate them or a technician could perform maintenance of other machines in the workcell.

Some definitions are necessary for a formal description of the surveillance problem. For any compact geometry of a real *object* o_r and each plane through the *focal point* $f^{\,i}$ of a colour or depth camera i within the three-dimensional space, two sectors of this plane can be defined (Figure 1). The *front sector* $F^i(o_r)$ of o_r is defined by the part of its surface, which is visible to the camera. The rightmost and leftmost points of the front side of o_r define the boundary to the occluded *back sector* $B^i(o_r)$ of o_r (including o_r).

In the image of a colour camera, only the silhouette of o_r is detected. Since no distance information is available, the complete *cone* $c^i(o_r)$ from $f^{\,i}$ into the *visible volume* V^i of camera i, which includes the silhouette of o_r, has to be regarded as *detected object* $o^i_d(o_r)$ (Figure 1 left, blue cone). This is the best conservative approximation of the geometry of o_r of colour images. In the image of a depth camera, the distance between $f^{\,i}$ and the surface of o_r in $F^i(o_r)$ can be measured, too. Since only the part of $c^i(o_r)$ in $B^i(o_r)$ is not visible by camera i, the area of $c^i(o_r)$ in $B^i(o_r)$ needs to be regarded as $o^i_d(o_r)$ (Figure 1 right). Colour cameras can be treated as depth cameras with binary depth measurement. If an object is detected, the measured distance is zero, else infinite. This allows the integrated usage of colour and depth cameras with a common representation of their measured image data.

The distance between a known geometry and o_r is $d(o_r)$. Since the geometry of o_r is approximated as $o^i_d(o_r)$, $d(o^i_d(o_r))$ is the approximated distance between the known geometry and o_r detected from an image. The distance in $F^i(o_r)$ can be approximated much better by depth than by colour images (Figure 1). Within $B^i(o_r)$, there is no difference between both sensors, since no geometrical information of o_r in $B^i(o_r)$ can be measured.

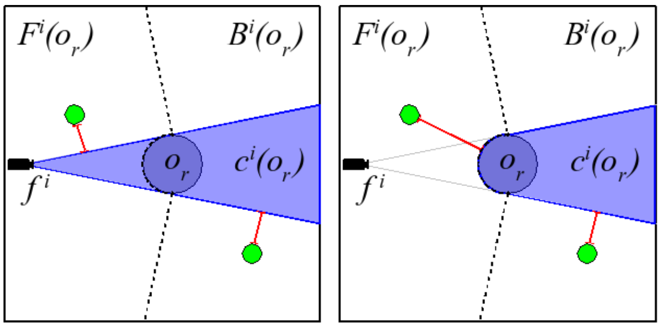

Figure 1 Illustration of the object detection (blue cones) of a real object geometry o_r in colour (left) and depth images (right) and the distance determination (red lines) to known geometries (green).

The presented surveillance approach is based on minimum distance calculation between a defined *safety zone S* of known geometry (e.g. a robot) and the detected unknown objects $o^i_d(o_r)$ (e.g. humans) by multiple simultaneous acquired camera images. The here introduced method integrates both camera types and follows a common surveillance rather collision detection approach [3]. Minimum distance calculation leads to a generalized collision detection, since a calculated minimum distance of zero indicates a detected collision. Smart cameras can calculate the computational steps of the surveillance algorithm, which must be done for each image, by themselves. Only the data fusion is not separable of the master. The contribution of this paper is a general approach for surveillance of safety zones with an arbitrary level of distributed processing, which controls the used method of data fusion (detected objects data or collision detection results of all cameras) in the surveillance algorithm.

The remainder of the paper first summarizes the state of the art in the related fields of research (Section II). In the following section (Section III), the presented surveillance approach and used distributed processing model is described. Afterwards, in the experimental section (Section IV), the performance using different camera types, collision detection methods and distribution levels in a demonstrator with the distributed surveillance algorithm is analyzed in detail.

II. Related Work

A. *Reconstruction and Computer Vision*

In computer graphics, 3D reconstruction of images from different perspectives is a prominent field of research with well known approaches.

The visual hull concept [10] uses object silhouettes in various colour images for a limited number of cameras. The intersection of the cones, which include the silhouettes, define the volume of the visual hull.

In [9], a stereo algorithm is used to improve the volumetric approximation of the visual hull. The presented system calculates 4 frames per second for a scene that contains a single object.

A mesh-based CAD model of an object based on multiple depth images provided by laser scans from different perspectives is presented in [12]. Merging these images result in a precise CAD mesh model. Constructing the mesh and merging each perspective at a resolution of 110x128 *pixels* requires about 6 *min*.

These algorithms target reconstruction of single objects in empty space and are not suitable for fast fusion of multiple depth images containing multiple objects to obtain polyhedral representations. But they describe different approximations of an object of colour and depth images.

In computer vision, several approaches for human detection are presented [6]. Most of those approaches use models for object recognition, but object models limit the capability to detect arbitrary unknown objects.

B. *Surveillance and Safety Systems*

Existing safety strategies provide 2½D or 3D surveillance, depending on the type and number of sensors used.

In [13] a single laser range finder acquires 1½D distance information within a plane just above the floor of the workcell. Any dynamic obstacles detected are assumed to be standing humans and are approximated by a vertical cylinder. The smallest distance between this cylinder and the robot limits the maximum robot velocity. However, other obstacles or humans in a stooping posture may not be correctly approximated. A second human behind the detected one is not detected.

In [14][15] a camera system acquires color images of the shared workspace from above. This enables a human and a SCARA robot to cooperatively assemble small work pieces. The human’s hands and neck are detected based on the characteristic color and texture features of the skin. The distance between human body parts and the robot is used to limit the maximum robot velocity. However, collisions with other parts of the human, other obstacles or larger work pieces are not detected. This system was extended by a stereo vision system acquiring 2½D distance information from above. However, it is unclear whether this method can cope with larger work pieces or obstacles other than exposed human body parts.

Global surveillance of a robot workcell and collision detection using multiple color cameras is subject of the SIMERO system [5]. By means of pixel classification, it is possible to separate humans or other dynamic unknown objects from the background of the workcell since they are not part of the static environment. These differences are used to calculate distances in one image [7] or between the visual hulls [4] of all obstacles in the surveilled space. Since no depth information can be measured in color images, only differences with respect to a reference image can be used to detect a dynamic unknown object. A dynamic object like a conveyor belt causes differences, too.

SafetyEye [11] calculates 2½D data about the surveilled space of a single stereo image and uses them to check for violation of predefined safety zones. These zones are defined as polygons with a given height. A violation of such a safety zone leads to stop or deceleration of the robot. During active surveillance, these zones are static and cannot be changed. There is no robot model integrated in the surveillance, so the robot workspace itself has to be masked.

In [16] an approach for surveillance of a robot workspace with a single PMD camera is presented. A robot model is used, which suppresses the robot data in the acquired depth image. For each link of the robot, a safety zone is defined. The size of this safety zone depends on the current speed of the robot. Now, the depth image can be checked for intersections with the safety zones. The performance of this system is 200 *ms* per collision detection cycle and additional time for acquiring the depth image of the camera.

In summary, all existing safety systems use multiple colour cameras or one depth sensor, but none of them use more than one depth sensor or both camera types for surveillance, especially including the fusion of their image data.

III. Multi Camera Distributed Surveillance

In this section, we summarize the main issues of the introduced approach. At first, the achievable quality of approximation of the minimum distance calculation is analyzed regarding the method of data fusion. Then the steps of the algorithm are described. The introduced model of distributed processing controls the sequence of the processed steps on the slaves and the master. The performance of the surveillance algorithm is controlled by this distribution level leading to an overall cycle time given by the maximum computation time of all camera slaves and additional computation time of the master. For theoretical comparison of the different distribution levels, the complexity of the computation times of all steps and the corresponding data complexity transferred between slaves and master is estimated.

A. Approximation Quality of Distance Measurement

For multiple colour or depth images, the approximation of o_r can be improved by intersecting the $o^i_d(o_r)$ of the images of all cameras i. For colour images, this results in the *visual hull* $h_V(o_r) = \cap_i o^i_d(o_r) = \cap_i c^i(o_r)$ (Figure 2 upper left). For depth images this intersection is called here *stereo hull* $h_S(o_r) = \cap_i o^i_d(o_r) = (\cap_i c^i(o_r)) \cap (\cap_i B^i(o_r))$ in the following (Figure 2 lower left). For all parts of the surface of o_r, which are in $F^i(o_r)$ of at least one input image, $h_S(o_r)$ is more accurate than $h_V(o_r)$. Only within the intersection of $B^i(o_r)$ of all input images, the quality of approximation of $h_S(o_r)$ and $h_V(o_r)$ is the same (Figure 2 right). Thus, a visual hull is a special case of the stereo hull concept, using binary depth measurement (zero, infinite). For a single camera image, the detected object and the hull depending on the camera type are the same.

Since o_r is approximated the more accurate by these hulls the more cameras are used, the distance $d(o_r)$ between safety zone S and o_r can be approximated the more accurate, too. In case of colour images, the equation

$$d(o_r) \geq d(h_V(o_r)) \geq d(o^i_d(o_r))$$

holds true in general. For depth images, these equations are

$$d(o_r) = d(h_S(o_r)) = d(o^i_d(o_r)) \text{ for } S \text{ in } F^i(o_r),$$

$$d(o_r) > d(h_S(o_r)) = d(h_V(o_r))) > d(o^i_d(o_r)) \text{ for } S \text{ in } \cap_i B^i(o_r).$$

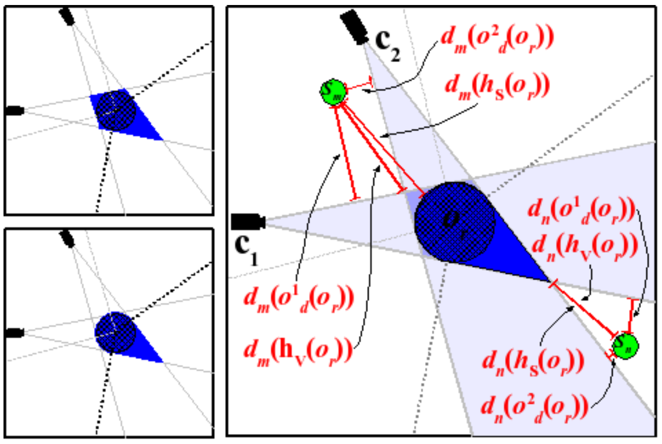

Figure 2 Left: Illustration of the object approximation (blue) of multiple colour images as visual hull (upper) or of multiple depth images as stereo hull (lower). Right: Illustration of the determined distances between known geometries (green) and detected object cones and hulls.

At first, the surveillance algorithm calculates the distances between S and each $o^i_d(o_r)$. The distance can be determined in two ways (Figure 3). The multi-camera image-based distance (a, b) is the maximum distance in all images $max_i\{d(o^i_d(o_r))\}$. The fusion-based distance (c, d) is $d(h_S(o_r))$ or $d(h_V(o_r))$. The collision detection for the complete workspace determines the minimum distance of all the distances between the detected unknown objects and S:

$$min\{max\{d(o^i_d(o_r))\}\} \text{ or } min\{d(h_S(o_r))\} \text{ or } min\{d(h_V(o_r))\}.$$

The most accurate result of the minimum distance calculation is achieved by fusion of the detected object data and calculating the minimum distance respective the resulting hulls (Figure 3c and 3d). If only depth cameras are used, the benefit of the fusion of detected object data of multiple cameras decreases with increasing number of depth cameras, since the intersection of $B^i(o_r)$ shrinks. If both camera types are used simultaneously, the benefit of fusion decreases for increasing number of depth cameras because of the same reason.

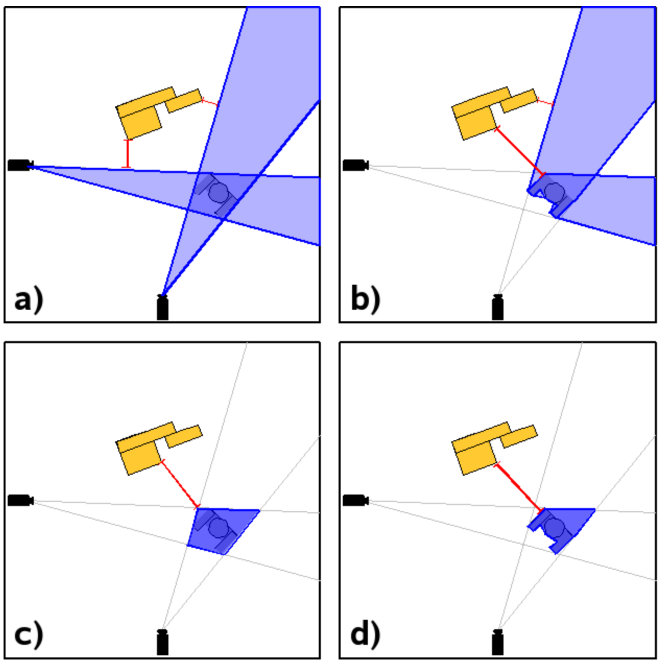

Figure 3 Illustration of the distance determination for multiple images of colour (a, c) or depth cameras (b, d). The image-based distance (a, b) is the maximum distance of each image. The distance based on fused object data (c, d) is the distance to the object hull resulting of the intersection of all images.

B. General Surveillance Approach

The surveillance algorithm cycle consists of three major computational steps, which are described in the following. At first, the detected objects of all camera images are calculated and fused. Second, *known objects* o_k, e. g. the robot, which are also detected in the depth images, must be suppressed in the minimum distance calculation. Therefore, detected objects of the o_k are generated virtually for all camera images and fused. All parts of the detected objects calculated in the first step, which are not within the generated objects of the second step, are *unknown detected objects*. Finally, the minimum distance between S and all detected unknown objects is calculated [2]. These major steps are algorithmically divided into sub-steps. Most sub-steps can be processed on both, master or slaves. Only the fusion sub-step must be processed on the master, since the used star network topology does not allow data exchange between different slaves. The processing distribution level controls the sequence of sub-steps, which are processed on either slave or master (fusion is done either for the object data or for the collision detection results of the slaves).

The objects are represented as convex polyhedrons in the algorithm. *Convex hulls* $h_C(o^i_d(o_r))$ are a conservative approximation of the detected objects of the real measurements. In computational geometry, efficient algorithms for convex polyhedrons are known [1]. Since real sensor values include error pixels without correct distance values, one characteristic of the convex hull is very useful: If a single point is measured above the real surface, the convex hull is extended by this point, which is a conservative estimation of the correct hull. If a single point is measured behind the real surface, it is "absorbed", since it lies within the convex hull.

To deal with noise in real sensor measurements, filters are used. The acquired depth images are filtered using smoothing filters (Median and Averaging) and the segmentation images are filtered using morphological filters (Erosion, Dilation, Open and Close).

The computation time complexity of the sub-steps is given by different parameters. They are estimated by parameters of a single camera image

- Image resolution of NxM *pixels*
- Number of detected objects O
- Maximum number of pixels in a detected object P
 and parameters for surveillance cycle in general
- Number of cameras C
- Number of detected unknown objects U

The computational steps (A-C) and their sub-steps (a-e) of the surveillance algorithm are:

A. *Object Detection*
Measured objects in the current camera image are detected, resulting in all $h_C(o^i_d(o_r))$ of a camera image.

a. *Object pixel Segmentation*
Using a background modeling method a binary difference image is calculated.
Complexity: $O(N*M)$

b. *Object pixel sets calculation*
Connected parts of the segmented pixels and their silhouettes in the image are calculated.
Complexity: $O(N*M)$

c. *Object point sets calculation*
For each object pixel in the calculated sets, its position in 3d space is calculated.
Complexity: $O(O*P)$

d. *Object hulls calculation*
The convex hulls of all object point sets are calculated, resulting in the detected objects.
Complexity: $O(O*(P \log(P)))$

e. *Object Data Fusion (only Master)*
The detected objects of all cameras are fused.
Complexity: $O(O^C)$

B. *Object Generation*
Using the known geometric occupation of the robot as o_k, a virtually detected $h_C(o^i_d(o_k))$ of the robot are generated.

a. *Virtual Object Detection*
The robot model is projected into an empty depth image and virtually detected as measured object.
Complexity: $O(P)$

b. *Generated Object Data Fusion (only Master)*
The generated objects of all images are fused.
Complexity: $O(C)$

C. *Collision Detection*
Of the two former calculated object sets, the minimum distance between S, here the robot model, and the detected unknown objects is calculated.

a. *Safety zone calculation*
The robot model for the current robot configuration is used as safety zone S.
Complexity: $O(1)$

b. *Minimum Distance Calculation*
Detected objects not within generated objects are classified as detected unknown objects. Their minimum distance to S is calculated.
Complexity: $O(U)$

c. *Minimum Distance Data Fusion (only Master)*
The maximum of all minimum distances in all cameras is calculated.
Complexity: $O(C)$

The single image based sub-steps of the surveillance algorithm must be processed C times, leading to an overall complexity of

$$C*(2*O(N*M)+O(O*P)+O(O*(P \log(P)))$$
$$+O(O^C)+C*O(P)+O(C)+O(1)+O(U)+O(C).$$

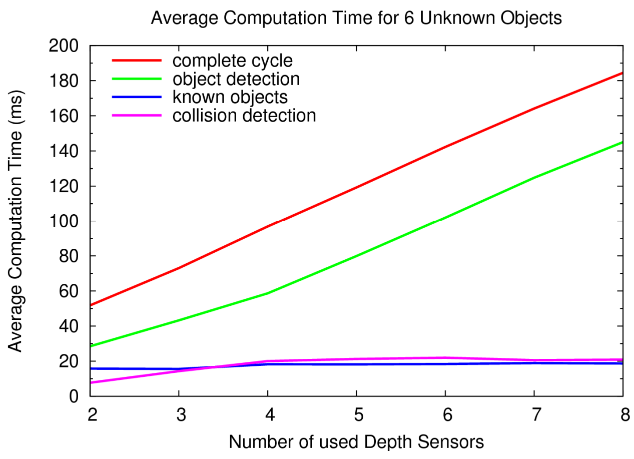

Figure 4 Diagram showing the average surveillance cycle time for a sequence of 1500 simulated input depth images of a workcell containing a robot and 6 unknown objects for an increasing number of depth cameras.

The algorithm was tested with simulated depth cameras in a simulated workcell without distributed processing (Figure 4). Since all sub-steps of the object detection must be processed once per camera and all detected objects of all cameras have to be fused, the asymptotic surveillance cycle time increases significantly in the number of used cameras. The time increases with a constant factor C for almost all sub-steps, only the fusion of object data increases exponential in C. Up to O^C possible intersections in the object data fusion sub-step must be calculated in the worst case. Some of the intersections can be excluded by bounding box intersection tests, but fusion complexity has the greatest fraction of surveillance cycle time.

C. Distributed Processing Model

The presented approach allows for an arbitrary level of distributed processing of the complete surveillance cycle. All distributed processing levels tested in experiments are described in TABLE I. The tested distributed processing levels are given by the sub-steps and named as the corresponding transferred data type between slaves and master.

TABLE I. DISTRIBUTED PROCESSING MODEL

Distributed Processing Level		**Processing Time and Data Transfer Complexity**		
		Sub-Steps on Slaves	***Sub-Steps on Master***	***Submitted Data from each Slave***
0	Raw Image		A: a, b, c, d, e B: a, b C: a, b	$N*M$ Float
1	Segmentation Image	A: a	A: b, c, d, e B: a, b C: a, b	$N*M$ (Float+Bool)
2	Object Pixels	A: a, b	A: c, d, e B: a, b C: a, b	$O*P$ (2xByte+Float)
3	Object Points	A: a, b, c	A: d, e B: a, b C: a, b	$O*P$ (3xFloat)
4	Object Vertices	A: a, b, c, d	A: d, e B: a, b C: a, b	$O*P$ (3xFloat)
5	Minimum Distance	A: a, b, c, d B: a C: a, b	C: c	Float

The current tested distributed processing levels are strongly related to sub-steps of the object detection, because the object detection is camera image centered and data exchange is only directed from slaves to master. Since the object generation is strongly linked to the following collision detection and has only a little fraction of computation time complexity of the surveillance algorithm, it is not reasonable to separate the processing of object generation and collision detection. For each distributed processing level except for the image-based collision test (level 5), the transferred object data can be fused on the master resulting in the detected objects in the workcell. This object data is used for collision detection as described in Figure 3c and 3d. The image-based collision detection method (Figure 3a and 3b) is only used for complete distributed processing.

The distributed processing levels can be classified in three categories, depending on their processed and transferred data. The first two distributed processing levels (0 and 1) process image data and transfer complete images. The network traffic is constant and little preprocessing is done on the slaves. The three following levels (2, 3 and 4) process object data of these images and lead to high variable network traffic. The last level (5) processes the complete surveillance algorithm respective one current image leading to minimum network traffic and minimum master centered processing.

0. Raw Image: The acquired images are directly transferred to the master (Figure 4). The data transferred via network is constant in each algorithm cycle.

1. Segmentation Image: The transferred size of data is constant, but the processed segmentation image and additionally the depth image in case of a depth camera must be transferred.

2. Object Pixels: For each detected object pixel, its image coordinate and in case of depth cameras, the depth value must be transferred. The size of transferred data ($O*P$) is not constant, but smaller than $N*M$ as complete images in general.

3. Object Points: Instead of transferring the pixels and their depth values to the master, their position in 3d space given by the camera calibration is transferred. While in former levels at least partial depth image information is available on the master, from this level on there is only processed object data. The size of transferred data is comparable to the former level as $O*P$.

4. Object Vertices: For calculated object hulls on the slaves, only the minimum set of points, which create the object hull, must be transferred. Since these optimized point sets only consists of the points on the convex hull, the hull must be calculated twice. The topological data of the hull (edges, faces) and bounding boxes or normals are lost during transfer. The tradeoff between computation time for input of optimized point sets in the convex hull calculation and reduced transfer time for less point data determines the quality of this distributed processing level.

5. Minimum Distance: The complete surveillance cycle is processed on data of the current camera, leading to the transfer of already usable collision detection results. The computation time of sub-steps on the master is minimized and almost constant in the number of cameras of the complete surveillance algorithm. The transferred data size is also constant and very small. As in other levels, the computation time of the slowest slave determine the overall algorithm cycle time, but in this case, it is almost the complete cycle time.

The used collision test method leads to some issues for the whole surveillance algorithm, which must be considered in the determination of the current distributed processing level. If the object data is fused (level 0 to 4), only objects in the intersection of all V^i of the cameras in the surveillance algorithm are considered in the collision detection. An object o_r, which is not in V^i of a camera, cannot be detected in this camera leading to no $o^i_d(o_r)$ an thus an empty intersection for $\cap_i\ o^i_d(o_r)$. For image-based collision detection (level 5), an object in any V^i leads to $o^i_d(o_r)$ and to $d(o^i_d(o_r))$ in this image and thus it is considered in the whole surveillance, even it is not detected in any other image. Another problem results of occlusions from known objects. Given enough perspectives of cameras in the algorithm, the intersection of occlusions caused by known objects is too small for unknown objects such as humans. These occlusions are much greater in single images in general. A more complex determination of the resulting minimum distance for image-based collision tests than the maximum of all minimum distances of the slaves may include plausibility checks via analysis of the current size of those occlusions. These plausibility checks are not focus of this paper and not discussed further.

IV. EXPERIMENTAL RESULTS

An experimental setup was built in an exemplary robot workcell using a colour-camera-based multi-image collision detection and a depth-camera-based collision detection with fused object data for the surveillance algorithm [3]. To evaluate the introduced distributed processing model, experiments were made using multiple camera slaves and a central master. The performance of the slaves, the master and the overall system was estimated and its asymptotic performance for an increasing number of cameras.

A. *Experimental Setup*

The prototype setup is built in a workcell containing a Stäubli RX130 robot (Figure 5, left). The CAD model of the robot is known and a direct connection between the robot control and the master is established. At any time the current robot configuration can be requested from the robot control and a velocity for the current trajectory segment can be sent to the robot control. Up to four PMD cameras [8] and up to eight colour cameras are used in the prototype setup. The used depth camera model vision 19k has a resolution of 120x160 *pixels* and an aperture angle of 40° and the colour cameras have a resolution of 480x640 *pixels* Bayer pattern and an aperture angle of 90°. Up to 10 *fps* of the depth camera and 30 *fps* of the colour cameras can be provided using a Firewire connection. Since PMD cameras are active sensors, each of them must use another modulation frequency. Multiple PMD cameras using a single modulation frequency would cause interferences within the measured depth images. Each camera slave has an AMD Sempron 3000+ processor and 512 MB RAM, the master has an INTEL Core2 Quad 2.66 GHz processor and 4 GB RAM (currently the algorithm only uses one core).

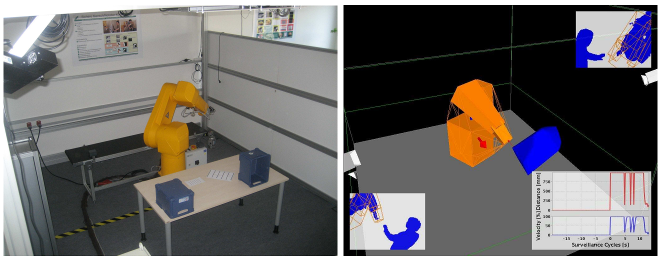

Figure 5 Left: Prototype setup in a workcell containing an industrial robot and depth and colour cameras in the upper corners of the workcell. Right: Visualisation of the surveillance algorithm output. The small images display the results of the object detection in the input images. The fused unknown objects (blue) and known objects (orange) are shown as 3D representations. The minimum distance (red line) is used for velocity control (diagram).

A CAD model of the robot workcell is used as background in the segmentation step. The robot model for the current robot configuration is generated and its 3D representation is displayed as orange volume and the generated object as orange grid (Figure 5 right). The minimum distance between the robot and the current nearest detected unknown object is calculated in the collision detection step and further used to limit the maximum velocity of the robot (Figure 5 right).

B. *Experiments for Distance Approximation Quality*

In the first experiment, several sequences of surveillance cycles were logged. In the first diagram (Figure 6), one camera is used for surveillance of a human moving within the robot workspace. In the second diagram (Figure 7), a static robot workcell is surveilled by two depth cameras. The distance between the robot and a box, which is standing in front of the robot, is calculated from the cameras and their fused data.

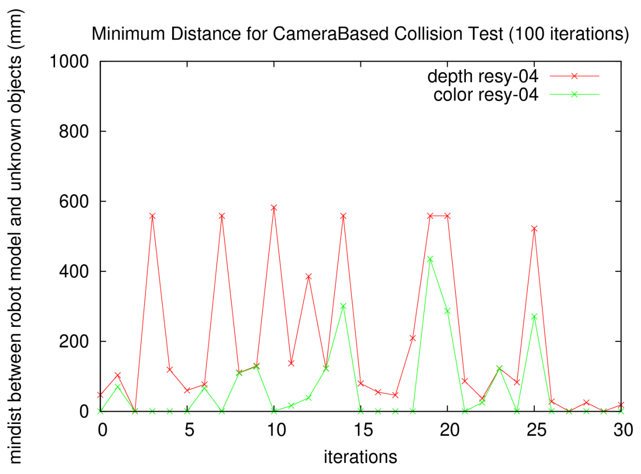

Figure 6 Diagram showing the different calculated distances between the robot and an object for used colour or depth information of the same image.

In Figure 6, a depth camera was used for image acquisition. Since two different cameras always have different external camera calibration parameters, their detected object of a real object will differ, too. To get comparable detected object data, the segmentation image of the depth camera is used as colour image (as described above). Using the depth information, the calculated distance is always greater than only using the segmentation data. This leads to a significantly better mobility of the robot.

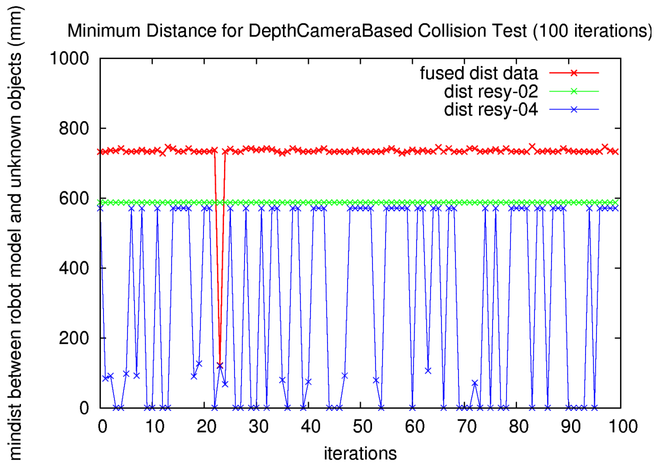

Figure 7 Diagram showing the different calculated distances between the robot and an object for two depth cameras. The image-based distances always are smaller than the fused object data based distance.

In Figure 7, the images of camera resy-04 are very noisy, leading to very small distance approximations. While the calculated distance between robot and box is approximately 600 mm in both depth images, the object data fusion of both depth images results in approximately 750 mm. Also, the noise of the second camera is almost completely vanished. In this way, the fusion provides both, better resolution and noise reduction properties. If only the segmentation images of both cameras are used, the determined distances are very similar, because in box is within the intersection of the $B^i(o_r)$. In the surveillance algorithm, the image-based distances are only used for level 5. For the remaining distributed processing levels, the object data fusion is used.

C. *Experiments for Distributed Processing Levels*

While the images for the experiments were acquired, the workcell was static, the robot was not moving and no human was walking through the workcell. In this way, the experimental results can be compared for each distributed processing level, since the same raw data was used.

1) *Overall Surveillance Algorithm Cycle*

In this experiment, the surveillance algorithm cycle time (Figure 8) for two used depth cameras and all distributed processing levels was tested (1000 cycles logged per level). There was no additional traffic in the network.

Concerning the slave computation time, the sub-steps processed on the slaves increase monotonously with increasing distributed processing level. Since the image acquisition and the segmentation computation time are constant in the image resolution, the computation time of the distributed processing levels 0 and 1 are almost the same for all slaves and constant in the logged surveillance cycles. The computation time of the remaining sub-steps (levels 2 to 5), which are all dependent on the number and pixel size of detected objects, differs significantly for different slaves, since the number and pixel size in the camera image is highly dependent on the perspective of the camera.

Concerning the master computation time, the remaining sub-steps of the surveillance algorithm, which must be processed on the master, decrease monotonously as its computation time. The sub-step of the object data fusion, which cause the main fraction of the algorithm cycle time on the master, cannot be saved for all distributed processing levels besides level 5, however. The sub-step of object hull calculation does not benefit from the preprocessing of points on the convex hull as minimum input point set on the slaves. The computation time on the master remains the same (level 4). The computation time of the master for the distributed processing level 5 is almost independent of the number of cameras, since only few arithmetic operations on float values must be calculated. The computation time is approximately zero.

Concerning the overall surveillance algorithm cycle time (including master and slaves), some practical issues are shown in the experiment. The calculation of the segmentation images and especially the transfer of both complete images is more expensive than the calculation of the segmentation images on the master (level 1). For calculated object data, there is only a slight difference between transferred object pixels and points in the overall surveillance algorithm cycle time (level 2 and 3). This corresponds to the theoretical result of the former section. The calculation of convex hulls on both, slaves and master, leads just to double computation time of this sub-step (level 4). The network transfer time and the calculation time of the master is the same as for transferred object points. Because of this, the overall time is higher than for level 3. For level 5, the overall computation time is only dependent on the slowest slave.

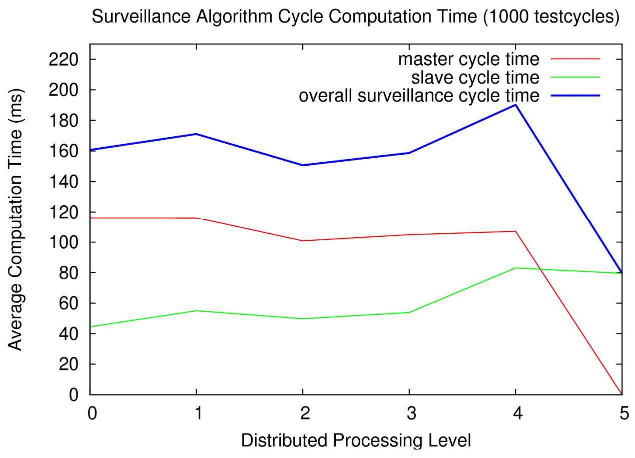

Figure 8 Diagram showing the average computation time of the surveillance algorithm cycle for two cameras.

Complete distributed processing (level 5) leads to minimum network transfer and minimum master computation time. Especially for networks, which are used concurrently by more applications, the network transfer can be a much more relevant factor. As shown in the diagram, the time for two cameras is improved by half the time of the collision detection for fused object data.

2) *Multi Camera Surveillance Cycle Time*

The surveillance algorithm was tested for an increasing number of depth and colour cameras in the experimental setup (Figure 9). The computation time was logged for 1000 surveillance cycles per distributed processing level.

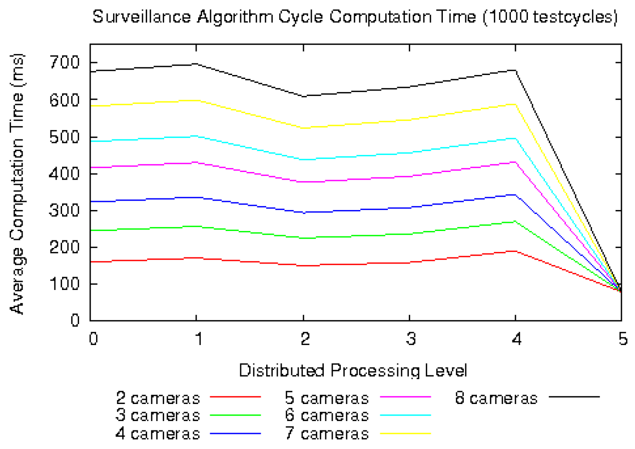

Figure 9 Diagram showing the average computation time of the surveillance algorithm cycle for an increasing number of cameras.

Only the computation time of the master increases for an increasing number of cameras in the surveillance algorithm. While the computation time of the object data fusion sub-step increases significantly, the computation time of the collision detection is almost constant, since the number of fused detected objects remains the same.

The percentage of computation times of the different distributed processing levels of the complete surveillance algorithm cycle for an increasing number of cameras remains similar to the former experiment with two cameras. This leads to reasonable distributed processing levels of transferred object pixels/points for collision detection for fused object data as in the former experiment. The surveillance algorithm computation time for image-based collision detection is constant in the number of cameras. Real-time restrictions of the surveillance application may require image-based collision detection, since surveillance cycle algorithm times of 500 *ms* are far to long for online surveillance of industrial robots. An increasing number of cameras leads to smaller intersections of $B^i(o_r)$ and an decreasing difference to the theoretical better resolution of collision detection for fused object data, too.

V. Conclusions

This paper introduces a general surveillance approach for colour and depth cameras including five possible levels of distributed processing. Dependent on the level of distribution, two different methods of data fusion are applied resulting in different asymptotic surveillance cycle times and minimum distance approximation qualities. The contribution of this paper is a model for distributing of the computational steps of a general surveillance approach using visual sensors. The distributed processing level model evaluates the distribution levels for a varying number of cameras, network performance, and processors on either slaves or master by estimating the asymptotic overall surveillance algorithm cycle time.

The model for distributed processing levels of the surveillance algorithm is evaluated by experiments in an prototype robot workcell. If the object data should be fused for collision detection, the best level of distribution is given by the maximum processing of image-based object data without convex hull computation (level 3). As the asymptotic surveillance cycle time is crucial for an increasing number of cameras, the benefit of object data fusion decreases, especially for depth cameras, as shown in Section 3A. For a surveillance application, which requires anytime capability, image-based collision detection leads to a valid result, once the first camera communicates its result (level 5). This result can be improved by the remaining cameras afterwards.

As camera types for the algorithm, colour and depth cameras are discussed. For example, infrared cameras provide the same geometrical information of their images as colour cameras. The main difference is in another background modeling and segmentation scheme for object detection in the image data. Any sensor, which provides at least the geometric information as the calibration of a colour camera, can be used.

Further research will address analyzing the difference in minimum distance approximation of both data fusion methods used in the surveillance algorithm and for different numbers of depth and colour cameras used in more detail. For more than two cameras, the distributed fusion of object data of subsets of the cameras can improve the overall surveillance cycle time. Therefore, another network topology as the current used star network with master as centre must be implemented.

Acknowledgment

This work was supported by the German Research Foundation (DFG) under the project name “Sicherheits-strategien für die Mensch/Roboter-Koexistenz und –Kooperation” (SIMERO) [5].

References

[1] C. B. Barber, D. P. Dobkin and H. T. Huhdanpaa, “The Quickhull algorithm for convex hulls,” ACM Transactions on Mathematical Software, 22(4): 469-483, Dec 1996, http://www.qhull.org.

[2] S. Cameron, “Enhancing GJK: Computing Minimum and Penetration Distances between Convex Polyhedra,” IEEE International Conference on Robotics and Automation, April 1997.

[3] M. Fischer and D. Henrich, “3D collision detection for Industrial Robots and Unknown Obstacles using Multiple Depth Images,” German Workshop on Robotics, June 2009.

[4] T. Gecks and D. Henrich, “Multi-Camera Collision Detection allowing for Object Occlusions,” 37th International Symposium on Robotics (ISR 2006) / 4th German Conference on Robotics (ROBOTIK 2006).

[5] D. Henrich and T. Gecks, “Multi-Camera collision detection between known and unknown objects,” IEEE International Conference on Distributed Smart Cameras, 2008.

[6] W. Hu, T. Tan, L. Wang and S. Maybank, “A survey on visual surveillance of object motion and behaviors,” IEEE Transaction on Systems, Man and Cybernetics, 34(3), 334-352, 2004.

[7] S. Kuhn and D. Henrich, “Fast Vision-Based Minimum Distance Determination Between Known and Unknown Objects,” IEEE International Conference on Intelligent Robots and Systems, 2007.

[8] H. Kraft et al., “3D-Camera of High 3D-Frame Rate, Depth Resolution and Background Light Elimination Based on Improved PMD (Photonic Mixer Device)-Technologies,” OPTO 2004, AMA Fachverband, http://www.pmdtec.com.

[9] M. Li, H. Schirmacher, M. Magnor and H-P. Seidel, “Combining Stereo and Visual Hull Information for On-Line Reconstruction and Rendering of Dynamic Scenes,“ IEEE Workshop on MMSP, pp9-12, 2002.

[10] W. Matusik, C. Buehler and L. McMillan, „Polyhedral Visual Hulls for Real-Time Rendering,“ Proceedings of the 12th Eurographics Workshop on Rendering, pp. 116-126, 2001.

[11] Patent DE 10 2006 057 605 A1 “Verfahren und Vorrichtung zum Überwachen eines dreidimensionalen Raumbereichs,” PILZ GmbH & Co. KG, 2006, http://www.safetyeye.com.

[12] M. K. Reed and P. K. Allen, “3-D Modeling from Range Imagery: An Incremental Method with a Planning Component,” Image and Vision Computing 17(2), pp99-111, 1999.

[13] F. Som, “Sichere Steuerungstechnik für den OTS-Einsatz von Robotern,” 4.Workshop für OTS-Systeme in der Robotik, IPA 2005.

[14] S. Thiemermann, “team@work - Mensch-Roboter-Kooperation in der Montage,“ 2. Workshop für OTS-Systeme in der Robotik, IPA 2003.

[15] S. Thiemermann, “Direkte Mensch-Roboter-Kooperation in der Kleinteilmontage mit einem SCARA-Roboter,“ IPA-IAO-Bericht, 2005, ISBN 978-3-936947-50-2.

[16] B. Winkler, “Safe Space Sharing Human-Robot Cooperation Using a 3D Time-of-Flight Camera,“ International Robots and Vision Show, 2007