In: 2nd ACM/IEEE International Conference on Distributed Smart Cameras – ICDSC 2008, Sept 7–11, Stanford/USA

MULTI-CAMERA COLLISION DETECTION BETWEEN KNOWN AND UNKNOWN OBJECTS

Dominik Henrich and Thorsten Gecks

ABSTRACT

Today, real-time collision detection is a basic demand for many applications. While collision tests between known (modeled) objects have been around for quite a while, collision detection of known objects with dynamic, unknown (sensor-detected) objects remains a challenging field of research, especially when it comes to real-time requirements. The collision test described in this paper is based on several stationary, calibrated video cameras, each supervising the entire 3-dimensional space shared by unknown and known objects (e.g. humans and robots). Based on their images, potential collisions of the known objects in any of their (future) configurations with a priori unknown dynamic obstacles are detected. Occlusions caused by known objects (such as the robot or machinery set-up within the workspace) are detected and addressed in a safe manner by exploiting the geometrical information of the known objects and the epipolar line geometry of the calibrated cameras in a decision fusion process. The algorithm can be parameterized to adapt to different application demands. Experimental validation shows that real-time behaviour is possible in the presence of highly dynamic unknown obstacles as they occur when humans and robots share the same workspace for the accomplishment of a shared task. In effect, the vision-based collision test can safely be used for human-robot cooperation, intrusion detection, velocity damping, or obstacle avoidance.

Index Terms—Vision, Collision Detection, Multi-Camera Image Fusion

1. INTRODUCTION

Real-time collision tests with a priori unknown (sensor-detected) objects are a basic precondition for numerous robot applications. For example, the basic robot skill of gross motions must be capable of taking the robot from the current configuration to a target configuration without colliding with dynamic obstacles present in the workspace, which provides safe coexistance with human operators.

To achieve this goal, information about the current state of the shared workspace is required. Multiple video cameras are convenient sensors, as they are widely available and cost-effective with regard to various criteria, such as resolution and update rate. The problem is extracting a representation of objects in the environment from the camera images.

The general problem can be specified as follows: The input is the geometry and kinematics of an known object (such as the robot) in a static 3-dimensional (3D) environment that may contain work pieces and static obstacles with known geometry. Additionally, the on-line images of stationary, calibrated cameras are given, each supervising the complete workspace shared by robot and other dynamic obstacles (unknown objects) such as humans. For a specific set-up, the maximum number of cameras in which the dynamic obstacles can simultaneously be occluded by the robot is known. Finally, the desired gross motions (transfer or pick-and-place) are provided. Output is the decision whether there is a collision between the robot and human or any other dynamic obstacle for any specific (future) configuration or along a complete motion segment.

This type of on-line collision test can be used in various applications. It provides the robot with additional capabilities, such as intrusion detection, velocity damping, or obstacle avoidance such as collision-free path planning. Most of them request the state of a configuration, after which the collision test returns either free if the robot can move safely into that configuration or occupied if a collision would occur. The collision test presented can determine the state of all possible robot configurations and not just the neighboring configurations of the current one, all of which is possible using a single set of images taken simultaneously.

In the following, we discuss related work in the area of sensor-based collision detection (Section 2). Then, we present our approach for detecting collisions during transfer motions (Section 3), which can be enhanced to detect collisions for pick-and-place motions (Section 4). Both approaches are validated by the results of our experiment, which applied them in a simple path planner (Section 5).

2. RELATED WORK

In the past, several approaches have been discussed for sensor-based collision detection and avoidance. They can be distinguished according to their sensing domain, i.e. whether global 3D information or global non-3D information about the environment.

2.1. Using global non-3D environment information

Non-3D global environment information can be acquired by stationary range finders or vision systems. The sensors commonly used acquire either 1½D, 2D or 2½D information.

In [28] a laser range finder acquires 1½D distance information within a plane just above the floor of the work cell. 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.

In [31] [29] a tri-ocular camera system acquires 2D color information 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 these 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.

In [27] [30] the above system was extended by a tri-ocular stereovision system acquiring 2½D distance information from above. However, it is unclear whether this method can cope with larger work pieces or obstacles other than bare human body parts.

In summary, it is possible to apply simplified collision detection strategies with non-3D global environment information. However, for robots moving in all three dimensions or for manipulation of large work pieces, 3D environment information is necessary.

2.2. Using global 3D environment information

Global 3D environment information can be acquired by multiple cameras, with each camera monitoring the entire scene form different viewpoints. This technique is widely used in computer graphics to determine the physical extent of an object by fusing multiple images of a scene with the back-projection method (also called shape-from-silhouette or volume-intersection) [9] [10]. However, they focus on the precise reconstruction of an object in 3D space, including texture information. A detailed analysis of the unwanted enlargement of reconstructed objects as a side effect of the back-projection method can be found in [23]. However, neither high precision nor texture information is necessary to detect dynamic obstacles. Furthermore, this technique is only capable of reconstructing single objects correctly, while scenarios of human/robot cooperation contain at least two objects.

To distinguish multiple objects in a 3D environment, the object information can be encoded by color. The following are examples of this approach: First, in [24] the back-projection approach is extended by assuming that the correspondence between pixels of different cameras can be resolved based on the object color. This assumption may hold true for small camera distances, but they in turn lead to large reconstruction errors. Second, in [19] the edges of the robots in a multi-robot system have a special color so they can be detected easily in the camera images. To detect collisions between the robots, the edges projected into the cameras are tested for intersections. However, even if the edges of other obstacles can be marked, this is hardly applicable to humans. Third, in [2] the robot and floor of the work cell are black while obstacles are white; the smallest distance between the projected robot tool center point and the obstacles in the images is used to detect collisions.

The restrictive object color-coding technique can be avoided by applying a difference image approach in multiple cameras. For each camera, the pixel-wise difference between the current image and the reference image is calculated. In [21] [22] [1] three grayscale cameras establishing multiple passive light barriers are used to detect collisions for a portal robot. Evenly distributed 3D points in the workspace are mapped to monitored pixels in each camera. If the features of the pixel differ from the given background values beyond a certain threshold, the system assumes that the corresponding point in space is occupied by an obstacle. However, when multiple objects are present in the workspace, non-existent (pseudo) obstacles can be falsely detected, a problem that will be discussed in Section 3.4. Additionally, it is assumed that the robot is not visible in the cameras, otherwise it will appear as an obstacle hindering its own motion. In [4] this assumption is abandoned and instead, the pixels covered by the current robot configuration are identified with a mapping table and treated separately. To test whether a future robot configuration collides, a second table maps all (obstacle) pixels to robot configurations. Even if the table look-up is fast, however, the table construction can be time- and memory-consuming. A more efficient approach based on a geometric robot model is discussed in Section 3.3.

In conclusion, 3D environment information must be acquired to exploit the full potential of robots moving in 3D, such as a complete gross motion collision test or collision-free path planning. The back-projection method can be extended to reconstruct multiple objects with color-coding. To detect collisions, an enhanced difference image approach with a separate treatment for robot pixels is sufficient.

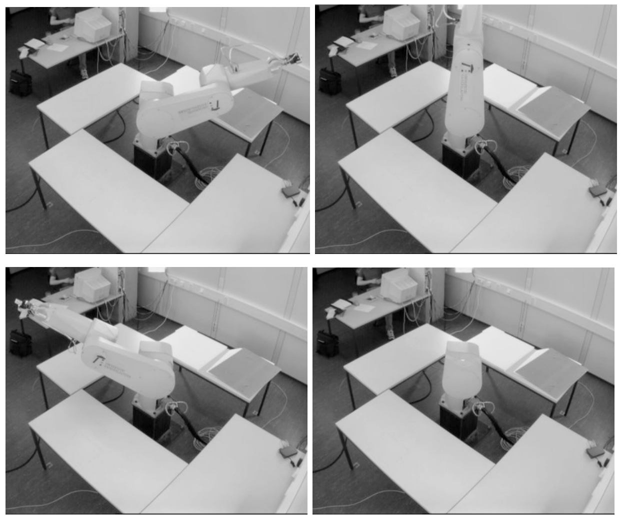


Fig. 1: To calculate the reference image for one camera, the three images (top-left, top-right, and bottom-left) showing the robot in different configurations are acquired. Then, the reference image (bottom-right) is calculated using the intensity median of all images for each pixel.

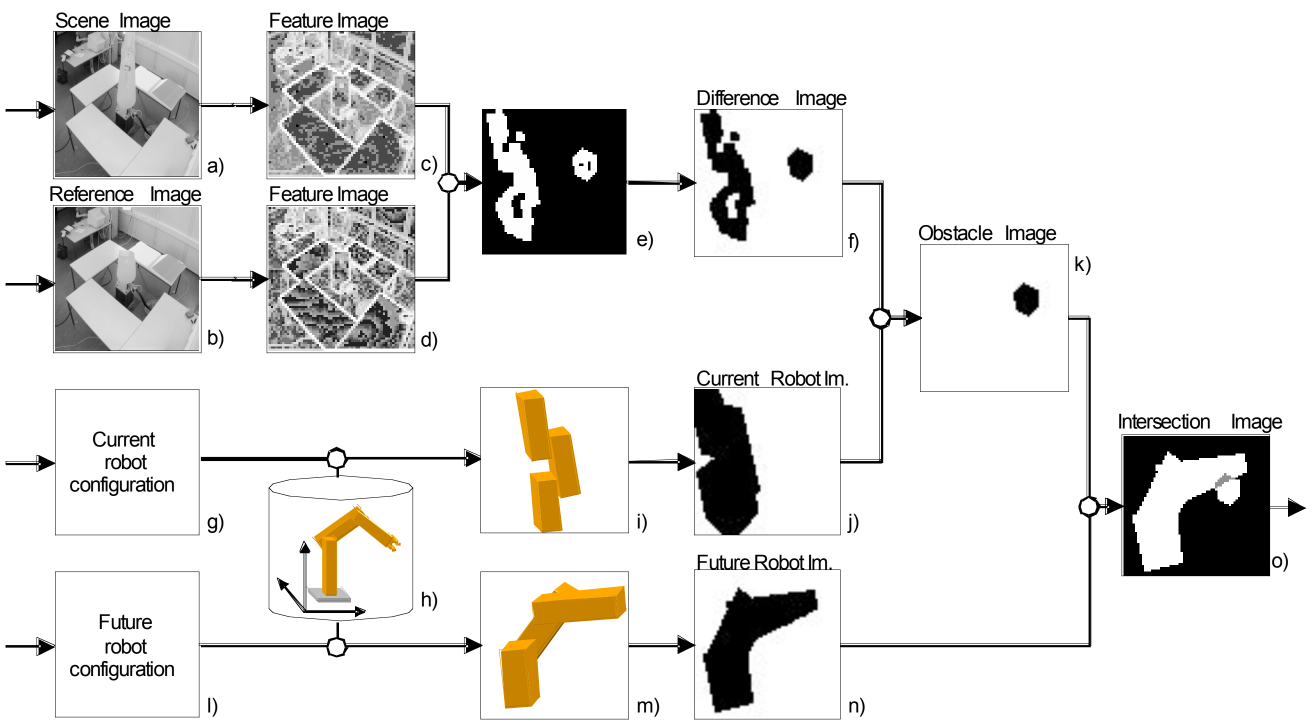


Fig. 2: Pictograms illustrating the data flow for detecting collision between the robot and dynamic obstacles with a single camera

3. COLLISION TEST FOR TRANSFER MOTIONS

The hardware of the collision detection system presented consists of several stationary video cameras, each monitoring the entire workspace shared by robot and unknown objects. The cameras are connected to a standard personal computer that processes their images. Additionally, the computer receives information about the current robot position from the robot controller via network.

The software generates a reference image of the workspace for each camera in a preliminary setup step (Section 3.1). Then an enhanced difference image method is applied, which compares the current workspace image with the reference image of the empty workspace (Section 3.2). This representation of the current workspace is then used in the basic collision detection algorithm (Section 3.3), which can be enhanced in three ways: First, the incomplete obstacle information can be refined with epipolar line information (Section 3.4). Second, the collision test itself can be enhanced by a refined collision condition (Section 3.5). Third, the collisions can be detected on a complete robot motion path by swept volumes (Section 3.6).

TABLE I:
PIXEL MAPPING FUNCTION TO CONSTRUCT THE OBSTACLE IMAGE

Difference Image	Current Robot Image	Obstacle Image
Background	Background	Background
Foreground	Background	Obstacle
Foreground	Robot	Unknown
Background	Robot	Error

Difference Image:

B	B	B	B	B	B	F	F
B	B	B	B	B	B	B	B
B	B	B	F	F	F	F	B
B	B	F	F	F	B	B	B
B	F	B	F	B	B	F	F
F	F	F	B	B	B	F	F
F	F	F	B	B	B	B	B
F	F	F	B	B	B	B	B

Current Robot Image:

B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B
B	B	B	R	R	R	R	B
B	B	R	R	R	B	B	B
B	R	R	R	B	B	B	B
R	R	R	B	B	B	B	B
R	R	R	B	B	B	B	B
R	R	R	B	B	B	B	B

Obstacle Image:

B	B	B	B	B	B	O	O
B	B	B	B	B	B	B	B
B	B	B	U	U	U	U	B
B	B	U	U	U	B	B	B
B	U	E	U	B	B	O	O
U	U	U	B	B	B	O	O
U	U	U	B	B	B	B	B
U	U	U	B	B	B	B	B

Fig. 3: The generated images for an example scene. The Difference Image and Current Robot Image are used to calculate the single-camera Obstacle Image according to Table I.

3.1. Reference image generation

Here, the *reference image* for a camera comprises the workspace containing only static obstacles, i.e. without robot and unknown objects. To avoid removal of the robot from the work cell, the reference image for each camera is calculated from multiple images including the robot [5]. For that purpose, images of the robot in various configurations are captured. The robot configurations are selected in such a manner that in each camera image and for all configurations, the robot covers as few common pixels as possible. Then, the reference image for each camera is created by calculating the median of the corresponding pixel values for all camera images. The median operation selects the pixel value present in the majority of the images, which represents the background. Thus, the mobile parts of the robot disappear in the reference image because the robot is visible only once in all configurations. The immobile robot base remains as a static obstacle (Fig. 1). Additionally, the cameras need to be calibrated so that the internal (focal length, distortion) and external parameters (position, orientation) are known. The calibration is necessary to project the robot model into the cameras (Sections 3.3, 3.6) and to exploit the epipolar line information (Section 3.4). To achieve an automatic calibration process, the system needs to be able to detect known 3D positions within the cameras. A wide variety of camera calibration methods exist; here, we used a blinking light bulb fixed in the robot gripper, which was then moved to the desired 3D positions and could easily be found in the camera images [20].

3.2. Enhanced difference image calculation

To calculate a robust difference image of the workspace, the current captured image (Fig. 2a) is evenly subdivided into non-overlapping tiles on a grid, so that each tile contains several pixels. This subdivision is also applied to the corresponding reference image (Fig. 2b). Each tile corresponds to a pixel in the difference image, which is thus of lower resolution. For each tile, several scalar features are calculated based on the pixel values in the tile (Fig. 2c, d). Examples for characteristic features are pixel value average, variance and contrast. Here, we used the average gray value and the y-coordinate of the gray value balance point, since they provide a good silhouette while using little computation time.

These features are then compared to the corresponding features of the reference images in the following steps: First, the metric defined for each feature is used to measure the distance between each tile of the scene and the reference image. Then, this distance vector is used to classify each tile as *background pixel* if no significant changes to the reference image exist and as *foreground pixel* if significant changes do exist (Fig. 2e). Several classification methods and automatic classification parameter optimization have been investigated [3]. Linear and statistical Bayes classification can achieve correct classification of ca. 97% of foreground tiles. Finally, the morphological operator Close is applied in a 4-neighborhood of all foreground tiles, resulting in the *difference image* (Fig. 2f).

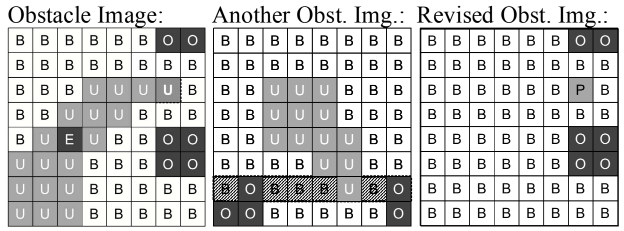
3.3. Basic collision detection algorithm

The collision detection requires a difference image containing only obstacles, because the robot itself is not considered to be an obstacle. Thus, we have to eliminate any foreground pixels caused by the robot from the current set of foreground pixels in each camera. Therefore, the current joint positions and an identifier of the gripped work piece is used (Fig. 2g). The 3D geometric and kinematic robot model (Fig. 2h) is parameterized by the current configuration and transformed to the current robot geometry (Fig. 2i). By means of the calibrated camera parameters, this geometry is then projected into each camera view, yielding the *current robot image* (Fig. 2j). This robot view is used to mask the robot within the difference image, resulting in an image containing only obstacles, called the *obstacle image* (Fig. 2k).

This masking operation is done according to Table I and

TABLE III:
PIXEL MAPPING FUNCTION TO REVISE THE OBSTACLE IMAGE
(ASSUMING $\Theta = 1$)

Obst. Image	Another Obstacle Img.	Revised Obst. Img.
Background		Background
Obstacle		Obstacle
Unknown	Epipolar line ∩ Obst. = ∅	Background
Unknown	Epipolar line ∩ Obst. ≠ ∅	Possible obstacle
Error	Epipolar line ∩ Obst. = ∅	Background
Error	Epipolar line ∩ Obst. ≠ ∅	Possible obstacle


Obstacle Image:

B	B	B	B	B	B	O	O
B	B	B	B	B	B	B	B
B	B	B	U	U	U	U	B
B	B	U	U	U	B	B	B
B	U	E	U	B	B	O	O
U	U	U	B	B	B	O	O
U	U	U	B	B	B	B	B
U	U	U	B	B	B	B	B

Another Obst. Img.:

B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B
B	B	U	U	U	B	B	B
B	B	U	U	U	B	B	B
B	B	U	U	U	U	B	B
B	B	B	B	U	U	B	B
B	O	B	B	B	U	B	O
O	O	B	B	B	B	B	O

Revised Obst. Img.:

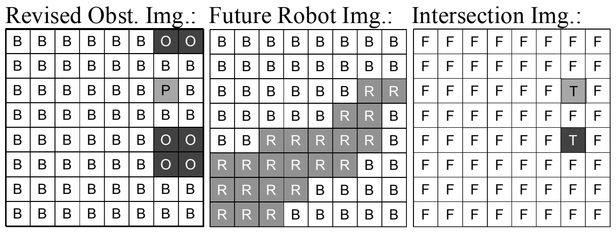
B	B	B	B	B	B	O	O
B	B	B	B	B	B	B	B
B	B	B	B	B	B	P	B
B	B	B	B	B	B	B	B
B	B	B	B	B	B	O	O
B	B	B	B	B	B	O	O
B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B

Fig. 5: The generated images for an example scene. Unknown and error pixels in the Obstacle Image are resolved by epipolar line information in Another Obstacle Image resulting in the Revised Obstacle Image according to Table III. One example pixel and the corresponding epipolar line are hatched. (Assuming $\theta = 1$)

will now be explained for a pixel-based example in Fig. 3. The binary pixels located at the same position in difference and robot image are used to determine one of the four values of the pixel in the obstacle image. If the difference and robot image pixels are set to background, the corresponding pixel in the obstacle image is also background. If the difference image pixel is foreground and the robot image pixel is background, the object seen in this pixel is an obstacle. If the

TABLE II:
PIXEL MAPPING FUNCTION TO CONSTRUCT THE INTERSECTION IMAGE

Revised Obst. Image	Future Robot Image	Intersection Image
Background	Robot	False
Obstacle	Robot	True
Possible obstacle	Robot	True
Background	Background	False
Obstacle	Background	False
Possible obstacle	Background	False


Revised Obst. Img.:

B	B	B	B	B	B	O	O
B	B	B	B	B	B	B	B
B	B	B	B	B	B	P	B
B	B	B	B	B	B	B	B
B	B	B	B	B	B	O	O
B	B	B	B	B	B	O	O
B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B

Future Robot Img.:

B	B	B	B	B	B	B	B
B	B	B	B	B	B	B	B
B	B	B	B	B	B	R	R
B	B	B	B	B	R	R	B
B	B	R	R	R	R	R	B
R	R	R	R	R	R	B	B
R	R	R	R	B	B	B	B
R	R	R	B	B	B	B	B

Intersection Img.:

F	F	F	F	F	F	F	F
F	F	F	F	F	F	F	F
F	F	F	F	F	F	T	F
F	F	F	F	F	F	F	F
F	F	F	F	F	F	T	F
F	F	F	F	F	F	F	F
F	F	F	F	F	F	F	F
F	F	F	F	F	F	F	F

Fig. 4: The generated images for an example scene. The Revised Obstacle Image and Future Robot Image are used to calculate the Intersection Image according to Table II.

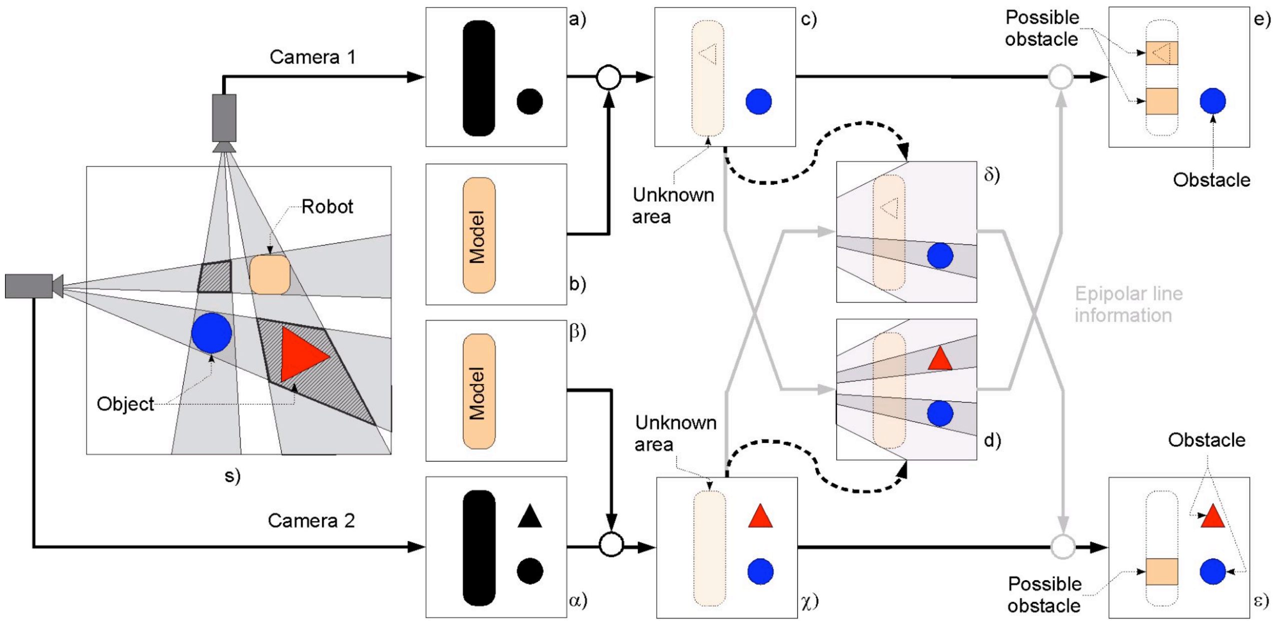


Fig. 6: Data flow of the revised obstacle reconstruction with two cameras illustrated by schematic images. The top-view of the scene is provided in (s) followed by the sequence of images corresponding to Camera 1 (a – e) and Camera 2 (α – ε). Difference images (a resp. α) show the scene from perspective of Camera 1 resp. 2. The model information (b resp. β) identifies the unknown areas covered by the robot in the obstacle images (c, χ). All data flow based on epipolar line information is illustrated in gray (d, δ). In the revised obstacle images (e, ε), the unknown areas are resolved resulting in *possible obstacles* and *obstacles*. (Assuming $\theta = 1$)

difference image pixel is foreground and the robot covers this pixel, the obstacle image pixel is set to unknown, as there might be an obstacle in front of or behind the robot. If the difference image pixel is background but the robot should cover this pixel, the obstacle image pixel is set to indicate an error because we see the background in a location where the robot should be. This error occurs either if the robot image is false due to an imprecisely determined robot position or a conservative robot model, or if an object in front of the robot or the robot itself has the same tile features as the background.

With the obstacle image and the robot model, we are able to check an arbitrary future robot configuration (Fig. 2l) for collisions with the current obstacle situation. The robot model (Fig. 2h) is transformed into the future robot geometry (Fig. 2m) and projection of this geometry into the camera view yields the *future robot image* (Fig. 2n). Then, the future robot image is intersected with the obstacles reconstructed from the scene image (Fig. 2k), resulting in the *intersection image* (Fig. 2o).

In a single-camera system, the future robot configuration can be considered as collision-free only if the intersection is completely empty, i.e. all robot pixels in the future robot image are background pixels in the obstacle image. Unfortunately, in applications with a continuously moving robot, the intersection is rarely empty since the projections of the current and future robot configurations typically overlap to some extent, i.e. some robot pixels in the future robot image are unknown pixels in the obstacle image.

A multi-camera system offers multiple perspectives of the (future) robot configuration, resulting in an improved ability to prove that a future robot configuration is collision-free. This proof depends on the number of cameras with no robot/obstacle intersections, which is determined by the maximum number θ of cameras in which a dynamic obstacle can simultaneously be occluded by the robot [4]. For example, if $\theta = 2$ then any dynamic obstacle is occluded by the robot in two cameras at most and a future robot configuration is collision-free if it causes no intersection with obstacles in more than two cameras. This number is called *occlusion threshold* θ in the following and is application-specific since it is influenced by the camera perspectives and the geometry of the obstacles and robot.

3.4. Revised obstacle image generation

The basic collision test has a major drawback: Even if there is an intersection of the future robot configuration with an obstacle in a camera, this future configuration will not necessarily lead to a collision. This may be the case when the robot only (partially) occludes the obstacle without contacting it.

To overcome this drawback, we need to further exploit the information from the other cameras. Most importantly, the portion of the robot that occludes the obstacle can probably be recognized as collision-free by a camera with another perspective of the scene if the epipolar line method is applied [12]. (A detailed overview of multiple-view geometry is provided in [25]). For this purpose, it is only necessary to revise the computation step depicted in Fig. 2f, j, and k. The algorithm is now presented for two cameras in

Fig. 6 and later extended to the multi-camera case.

Recall, that the current scene (Fig. 6s) and the resulting difference images for each camera are given (Fig. 6a and α, Fig. 2f). Furthermore, the current robot images are generated using the current robot configuration (Fig. 6b and β, Fig. 2j). Finally, the robot in the difference image is masked by the robot images, resulting in the obstacle images (Fig. 6c and χ, Fig. 2k). Thus, the procedure is similar to the basic collision detection algorithm.

The construction of the revised obstacle image is based on the obstacle images which contain areas occluded by the robot. For Camera 1, these occluded areas are resolved by projecting them in the obstacle image of Camera 2 using epipolar geometry (Fig. 6d). The same is done for Camera 2 (Fig. 6δ). If these projections intersect with obstacles, then the areas occluded by the robot may cover possible obstacles, otherwise they cover background. This information is stored in the revised obstacle images (Fig. 6e and ε). In the scene top-view, the areas of possible obstacles are hatched.

At the pixel level, the revised obstacle image generation is done according to Table III and illustrated further for our example in Fig. 5. Recall that the area covered by the robot in the obstacle image results in unknown and error pixels which need to be resolved. To do so, each unknown or error pixel is projected into the obstacle image of the other camera as an epipolar line. (In Fig. 5 one of these epipolar lines is represented by the hatched pixels and originates from the hatched robot pixel in the obstacle image.) Then, the epipolar lines are checked for intersection with obstacle pixels, i.e. with any pixel that is neither a background nor robot pixel. If there is an intersection, then the corresponding pixel is an area where the robot may cover an obstacle. Thus, this pixel is labeled as a *possible obstacle pixel*, otherwise it is marked as a background pixel in the resolved obstacle image. All the other pixels remain unchanged.

In a multi-camera system with C cameras, all the cameras are used to resolve pixels. For each unknown or error pixel, the corresponding epipolar lines are projected in all other camera images and tested for intersections with obstacle pixels. If the number of cameras signaling an intersection is greater than or equal to $C - \theta$, then this pixel is labeled as a possible obstacle and otherwise as background.

With the revised obstacle image, the collision test (Fig. 2k, n, and o) is done according to Table II and illustrated in Fig. 4. The future robot configuration is projected in the camera and checked for intersection. In this example, the future robot configuration intersects with both obstacle and possible obstacle pixels; thus, the camera signals a collision. In general, the multi-camera system will consider a future robot configuration as collision-free if there is no collision in at least one camera, i.e. no intersections of the future robot configuration with either obstacle or possible obstacle pixels.

3.5. Revised collision test condition

The collision test mentioned above treats intersections with obstacles and possible obstacles equally. Unfortunately, this may lead to robot immobility in cases where the robot contacts possible obstacles (Fig. 6e and ε). In our example, this originates in the fact that possible obstacles are reconstructed for both cameras. Therefore, any robot configuration close to the current position results in intersections with possible obstacles and the robot remains immobile.

This problem can be solved with a more elaborate collision test considering pixel sets instead of single pixels, e.g. all robot pixels in an image form the robot pixel set. Additionally, it is based on the assumption that the robot is collision-free within its very own volume. In Table IV this assumption is applied to all cases involving testing a future robot configuration for collisions with (possible) obstacles for two cameras with $\theta = 1$. Additionally, examples of such cases are provided in Table IV. For an arbitrary number of cameras C, the 16 possible cases that can occur can be subdivided into the following four classes:

1. Class A: Covers all cases in which at least one camera signals no intersection with either pixel type. For all cases in this class, there can be no collision with an obstacle at all, because if any volume intersection exists between a (possible) obstacle and the future robot configuration, each camera contains a pixel set intersection of the respective projected volumes. (This is analogue to Section 3.4.)

TABLE IV:
CASES OF INTERSECTION BETWEEN FUTURE ROBOT PIXEL SETS AND OBSTACLE AND POSSIBLE OBSTACLE PIXEL SETS IN A TWO-CAMERA SYSTEM

Robot pixel set intersects in Camera 1 with		Robot pixel set intersects in Camera 2 with		Class
Obstacle pixel set	Possible obst. pixel set	Obstacle pixel set	Possible obst. pixel set	
false	false	false	false	A
false	false	false	true	A
false	false	true	false	A
false	false	true	true	A
false	true	false	false	A
false	true	false	true	B
false	true	true	false	C
false	true	true	true	C
true	false	false	false	A
true	false	false	true	C
true	false	true	false	D
true	false	true	true	D
true	true	false	false	A
true	true	false	true	C
true	true	true	false	D
true	true	true	true	D

2. Class B: Comprises all cases that are not in Class A and for which the following condition holds true: Less than $C - \theta$ cameras signal collisions only with obstacles. Cases in this class are collision-free because if they collide, at least $C - \theta$ cameras would show a collision with an obstacle, as any obstacle is visible in at least $C - \theta$ cameras.
3. Class C: Comprises all cases that are not in class A or B and for which the following condition holds true: At least $C - \theta$ but less than C cameras signal a collision with an obstacle. Cases in this class are not collision-free, because we must assume that the robot intersects with a volume that contains an obstacle, as this volume is visible in at least $C - \theta$ cameras as an object, which assumed to be true for any real dynamic obstacle.
4. Class D: Comprises all other cases (C cameras indicate a collision with an object and an arbitrary number less than C indicates collision with a pseudo-object). These cases indicate a definite collision with the obstacle volume. It is nevertheless possible that in reality even these robot configurations do not actually collide because the visual hull of an object is a conservative approximation and exceeds the true object volume.

With this insight, the collision test condition of a (future) robot configuration can now be stated more formally. Let in the obstacle image S_i of camera i the set O_i be the foreground pixels labeled as obstacle pixels and the set P_i be the foreground pixels labeled as possible obstacle pixels. Further, let the set R_i be the robot pixels of an arbitrary robot configuration. We can then introduce the predicates $Coll_{RO}$ and $Coll_{RP}$ providing the Boolean information whether camera i signals a collision with real obstacles and with possible obstacles, respectively:

$$Coll_{RO}(R_i, S_i) := R_i \cap O_i \neq \varnothing$$

$$Coll_{RP}(R_i, S_i) := R_i \cap P_i \neq \varnothing$$

In general, collisions can be detected for the current or any future robot configuration. Let the set T_i be pixels of the robot in a certain configuration. Additionally, let the shared workspace be observed by C cameras and allow any obstacle to be occluded by the robot in at most θ cameras. Then, a collision is detected if at least $C - \theta$ cameras detect an obstacle collision and no camera is completely free of either obstacle or possible obstacle collisions. Thus, a collision is detected if and only if the following condition holds true:

$$\left(\left\| \left\{ i \mid Coll_{RO}(T_i, S_i) \right\} \right\| \geq C - \theta \right) \wedge \left(\left\| \left\{ j \mid Coll_{RO}(T_j, S_j) \vee Coll_{RP}(T_j, S_j) \right\} \right\| = C \right)$$

3.6. Collision detection for complete motions

The vision-based collision test of single robot configurations can be extended to test complete robot motions of arbitrary shape. To do so, it is sufficient to regard the volume swept by the robot during the motion and test it against the obstacle volumes. In the collision test algorithm described in Section 3.3, this is done by exchanging the future robot image for each camera with the projection of the swept volume, the *swept volume image*.

To calculate the swept volume image, the motion is interpolated by several intermediate robot configurations. For each configuration, the corresponding robot geometry is transformed and projected into the camera. To avoid possible gaps between the projections of two successive configurations, we can either reduce the Cartesian distance $x_{\max}$ between the successive configurations or enlarge the robot geometry by radius r. There will be no gaps if the condition $x_{\max} < 2r$ holds true. (In our experiments, we set $x_{\max}$ to the smallest link cross-section of the robot.) Altogether, the union of the individual projected configurations yields the swept volume image.

To determine the intermediate robot configurations for a serial robot with only rotatory joints, the given maximum Cartesian movement $x_{\max}$ can be transformed into a maximum joint movement [15]. Let l_i be the distance between the rotational axis of joint i and the farthest point the end effector can reach. Then, the maximum joint movement Δq_i, which does not exceed the Cartesian $x_{\max}$ in the worst case, is calculated as

$$\Delta q_i = 2 \arcsin\left(\frac{x_{\max}}{2 \cdot l_i}\right)$$

The movement of joint i during the complete robot motion is then subdivided into portions of size Δq_i. If there are n_i portions of joint i and d joints total, then there are $n = n_1 + \ldots + n_d$ single portions, each of which may result in a robot movement of $x_{\max}$ in the worst case scenario. Thus, we need to subdivide the complete robot motion into n intermediate configurations in order to guarantee a robot movement shorter than $x_{\max}$ between successive configurations. To reduce redundant computation each robot link is rendered only if necessary, that is if its actual movement exceeds the distance $x_{\max}$.

4. COLLISION TEST FOR PICK-AND-PLACE OPERATIONS

The classical difference image method involves the underlying assumption that a static environment is present, which is impractical for many applications [11] [13]. For example, typical tasks for industrial robots involve moving work pieces from one location to another (*pick-and-place operations*), which changes the robot's environment. These changes usually result in additional foreground pixels in the difference image at the previous and present locations of the work piece. The more locations are involved, the more cluttered with virtual obstacles the difference image will become, forcing the robot into immobility. Nevertheless, these environmental changes are part of the application process and need to be considered. Since the difference image method stores an environment representation in the reference image, an update of the reference image is necessary (Section 4.1). A survey of reference image update algorithms is given in [26]. As with the difference image areas occluded by the robot, here, the reference image update areas occluded by obstacles require special attention.

4.1. Reference image update algorithm

The central data structure of the reference image update algorithm is the pixel set U of difference image pixels that define the area of changed pixel values. Consequently, a difference image pixel belonging to U defines the tile of the reference image that needs to be replaced with the corresponding tile in the current scene image, resulting in an updated reference image.

The update process starts when the robot either picks or places a work piece. This start time t_0 can be signaled by the robot application program by providing an additional identifier for the picked or placed work piece. The corresponding geometric model of the work piece is used for two purposes: First, the model determines the position and shape of the initial update area U. Second, the work piece model extends the robot model for save transfer movements with the respective work piece gripped (Section 3).

Additionally, the update process must prevent a foreground object (obstacle) from becoming integrated into the reference image. A problem is that no obstacle can be directly observed based on image differences in the update area, as image differences already occur in this area because of the pick/place process. Based on the epipolar line technique described in Section 3.4, we can detect the set of possible obstacle pixels O within the update area of one camera based on obstacle information from other cameras views (Fig. 8c). Treating these possible obstacles as real and removing their pixels from the update area pixel set, we can safely update the remaining pixels of the update area, resulting in the refined update algorithm described above.

With these preliminary considerations, the update algorithm can be formulated in pseudo-code:

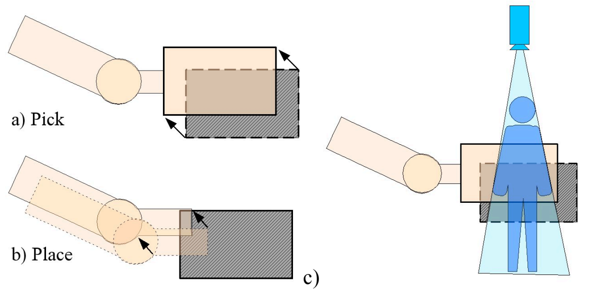


Fig. 8: Illustration of the update procedure during a pick operation (a), a place operation (b) and an object occlusion (c). The remaining set E of pixels to be updated is hatched. Obstacles in front of the update area in the camera view are detected with the help of obstacle information along epipolar lines in the other cameras.

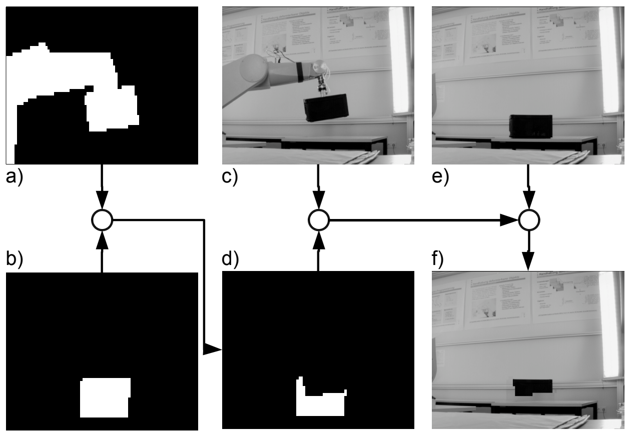


Fig. 7: Data flow of a step in the reference image update algorithm with images of a sample scene where the robot performs a pick operation.

```
$U$ := computeWorkPiecePixels($t_i$)
repeat
    $R$ : = computeRobotModelPixels($t_0$)
    $E$ := $U \setminus R$
    $O$ := computePossibleObstaclePixels($t_i$)
    $E$ := $E \setminus O$
    updateReferenceImageTiles($E$)
    $U$ := $U \setminus E$
until $U = \varnothing$
```

First, an image of the work piece at the robot position at time t_0 is calculated, which determines the pixel set U. In each iteration step i at time t_i of the system, the robot model provides the current pixel set R defining the robot's shape. This set determines the currently occluded update area, and thus by subtraction, the set of pixels to be updated E is produced. Pixels of the set O that might be an obstacle in front of the update area need to be subtracted too.

The data flow for one update step during a pick operation is illustrated in Fig. 7. The sequence is from a trial run of our prototype system with the robot cyclically picking and placing a black box. Fig. 7a shows the image of the robot model including the object currently being picked. Fig. 7b indicates the update area derived from the object model image at the pickup position. If the robot model image is subtracted from the update area, the result is the update area in Fig. 7d. The update area indicates the pixel of the current scene image in Fig. 7c, which can be used to update the current reference image in Fig. 7e. The update result is the new reference image in Fig. 7f that only shows a portion of the black box remaining to be updated.

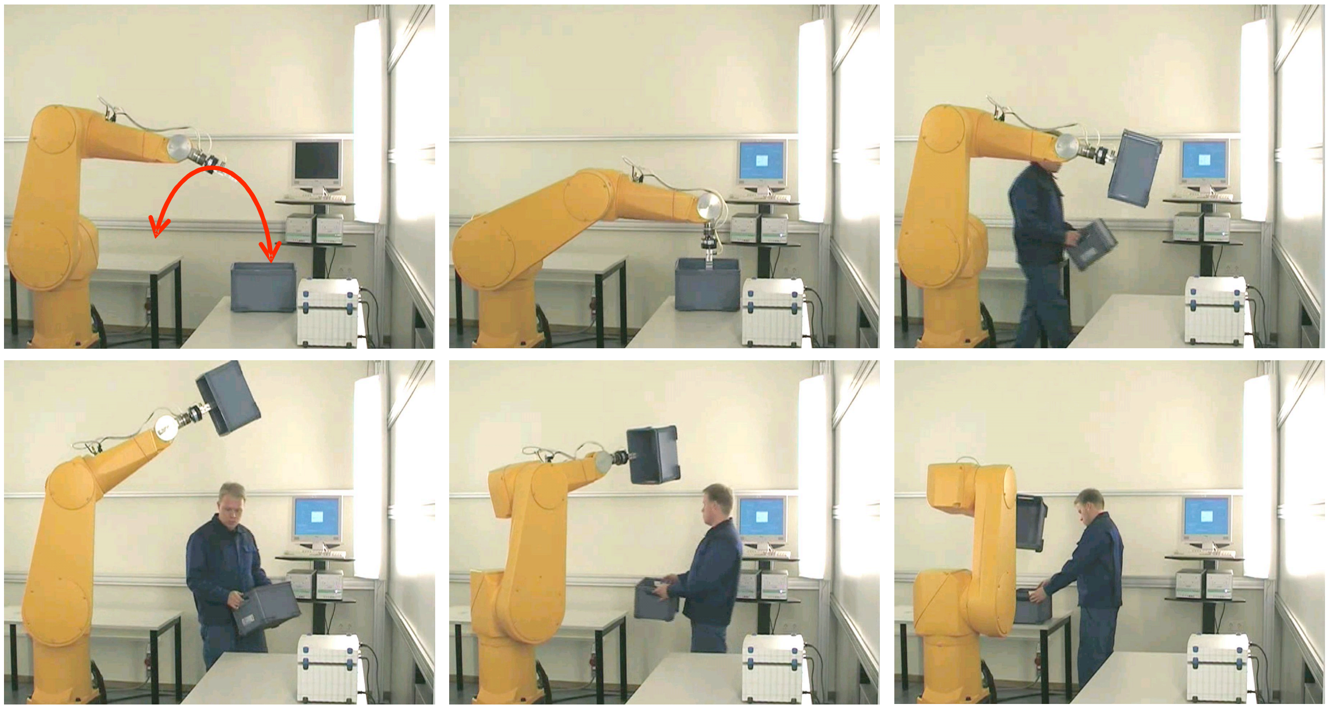


Fig. 9: Photo sequence of a collision-free pick-and-place motion avoiding dynamic obstacles as humans and work pieces. The undisturbed robot paths are indicated in the top-left photo.

5. EXPERIMENTAL RESULTS

The algorithms were implemented and evaluated in the prototype system. (Previous versions have been presented in [6] [7].) As sensors, four grayscale DMK 73/C CCD cameras were connected to two DFG/BW1 Frame grabbers. These frame grabbers were installed in one AMD-Athlon 2200+ PC, with two cameras attached to each frame grabber. The images from all four cameras were acquired simultaneously. The resolution of the cameras was 704×576 and that of the difference images was set to non-overlapping 64×64 tile pixels. An industrial robot Stäubli RX130 was controlled by an Adept CS7 robot controller, which was connected via 10 Mbit Ethernet to the server PC. The software was implemented in Visual C++ 6.0 on the Windows NT 4.0 SP 6 operating system.

The workspace shared by human as an unknown object and robot had a size of ca. 1.5×2×2 m^3. The maximum robot speed was limited to 70 cm/s to ensure that a human still feels comfortable in the vicinity of the moving robot. With an update rate of ca. 7 Hz, the robot configuration space was subdivided into 8192 robot positions.

The vision-based collision test presented can be applied to collision-free path planning, for example. As a simple global planning method, we chose the wave-propagation algorithm in [18]. Each joint was individually discretized, limiting the maximum robot movement to a predefined Cartesian distance [15]. The resulting accuracy allows even close (30 to 50 cm) approaches to *a priori* unknown obstacles.

A prototypical realization of an application example is shown in Fig. 9. The system's task is to transport the black work piece from one workplace to another. In automatic mode, the undisturbed robot path is indicated by the arrow (Fig. 9, top-left). The figure shows the reaction of the system to the obstruction of the programmed robot path. First, a collision with the operator is avoided by raising the work piece (Fig. 9, bottom-left). Second, the system stops the motion since the goal position is occupied by a similar work piece placed there by the operator (Fig. 9, bottom-right).

6. CONCLUSION

We presented a vision-based collision test to enable robot applications in environments containing a priori unknown objects. The approach uses multiple stationary cameras to detect *a priori* unknown dynamic obstacles such as humans. The basic difference image method is enhanced by classifying foreground and background pixels, by exploiting epipolar line information, by considering pixel sets in the collision test and by automatically updating the reference image. It enables collision detection for transfer motions and pick-and-place motions in a 3D environment. Since any future robot configuration can be tested, collision-free path planning is applicable.

Future work may augment this approach in several ways: First, the images may be pre-processed by smart cameras to improve computing speed [8]. Second, instead of a collision test, the smallest distance between robot and dynamic obstacles may be calculated to improve the velocity damping and path planning capabilities [16] [17]. Third, non-static environments (background pixels) such as conveyer belts or illumination changes may be considered to enlarge the application area. Finally, the robot motions may be guided by the human to enable closer human/robot cooperation [14].

REFERENCES

[1] Ameling W. (Ed.), *Flexible Handhabungsgeräte im Maschinenbau*, Ergebnisse aus dem Sonderforschungsbereich 208, VCH Publishing, 1996

[2] Bischoff A., *Echtzeit Kollisionsvermeidung für einen Industrieroboter durch 3D-Sensorüberwachung*, Diplomarbeit Fernuniversität Hagen, 1999

[3] Ebert D., *Bildbasierte Erzeugung kollisionsfreier Transferbewegungen für Industrieroboter*, Dissertation, Schriftenreihe Fachbereich Informatik, Band 12, Verlag Universität Kaiserslautern, 2003

[4] Ebert D., Henrich D., "Safe human/robot cooperation: Image-based collision detection for industrial robots", in *IEEE International Conference on Intelligent Robots and Systems* (IROS'02), Lausanne, September 30th - October 4th, 2002

[5] Ebert D., Henrich D., "Safe human/robot cooperation: Problem analysis, system concept and fast sensor fusion", in *IEEE Conference on Multi-sensor Fusion and Integration for Intelligent Systems* (MFI'01), Baden-Baden, August 20-22, 2001

[6] Ebert D., Henrich D., „SIMERO - Sicherheitsstrategien für die Mensch-Roboter-Kooperation", in *OTS-Systeme in der Robotik – Roboter Ohne Trennende Schutzeinrichtungen*, Reihe BKM Berichte, Herbert Utz Publishing, pp. 5.1-5.17, München, June 25., 2002

[7] Ebert D., Henrich, D., "SIMERO - Sichere Mensch-Roboter-Koexistenz", in *2. Workshop für OTS-Systeme in der Robotik - Mensch und Roboter ohne trennende Schutzsysteme*, Fraunhofer IRB, S. 119-134, Stuttgart, June 24, 2003

[8] Ebert D., Komuro T., Namiki A., Ishikawa M., "Safe human/robot coexistence: Emergency-stop using a high-speed vision chip", in *IEEE International Conference on Intelligent Robots and Systems* (IROS'05), Barcelona, April 18-22, 2005

[9] Eckert G., "Automatic Shape Reconstruction of Rigid 3-D Objects from Multiple Calibrated Images", in *Eusipco 2000 Proceedings*, Tampere, Finland, 2000

[10] Eisert P., Steinbach E., Girod B., „Automatic reconstruction of stationary 3-D objects from multiple uncalibrated camera views", in *IEEE Transactions on Circuits and Systems for Video Technology: Special Issue on 3D Video Technology*, vol. 10, no. 2, pp. 261-277, March 2000

[11] Gecks T., Henrich D., „Human/robot cooperation: Safe pick-and-place operations", in *14th IEEE International Workshop on Robot and Human Interactive Communication* (ROMAN'05), Nashville, August 13-15, 2005

[12] Gecks T., Henrich D., „Multi-camera collision detection allowing for object occlusions", in *37th International Symposium on Robotics* (ISR'06), *4th German Conference on Robotics* (ROBOTIK'06), Munich, May 15-17, 2006

[13] Gecks T., Henrich D., „SIMERO: Camera supervised workspace for service robots", in *ASER 2004 2nd International Workshop on Advances in Service Robotics*, Feldafing, Germany, May 20-21, 2004

[14] Henrich D., Kuhn S., „Modelling intuitive behaviour for safe human/robot coexistence and cooperation", in *IEEE International Conference on Robotics and Automation* (ICRA'06), Orlando, May 15-19, 2006

[15] Henrich D., Wurll Ch., Wörn H., "On-line path planning with optimal C-space discretisation", in *IEEE/RSJ Int. Conference on Intelligent Robots and Systems* (IROS'98), Victoria, Canada, October 12-16, 1998

[16] Kuhn S., Gecks T., and Henrich D., "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* (MFI'06), Heidelberg, September 3-6, 2006

[17] Kuhn S., Henrich D., "Distance-based Velocity Control for Safe Human/Robot Coexistence and Cooperation", submitted to *IEEE Int. Conf. On Robotics and Automation*, Rom, 2007

[18] Latombe J.-C., *Robot motion planning*, 4th print, Kluwer Acad. Publ., Boston, 1996

[19] Leou J. J., Chang Y. L., Wu J. S., „Robot operation monitoring for collision avoidance by image sequence analysis", in: *Pattern Recognition*, Vol. 25, No. 8, pp. 855-867, 1992

[20] Liu, Hoover, Walker, "Sensor network based workcell for industrial robots", in *IEEE International Conference on Intelligent Robots and Systems* (IROS'01), 2001

[21] Meisel A., *3D-Bildverarbeitung für feste und bewegte Kameras*, Vieweg Verlag, Reihe Forschritte der Robotik Nr. 21, 1994

[22] Meisel A., Föhr R., Ameling W., "3D-Kollisionsschutzsensor auf der Basis von CCD-Kameras", in *SENSOR* 91, pp. 157-170, Nürnberg, May 1991

[23] Niem W., "Error Analysis for Silhouette-Based 3D Shape Estimation from Multiple Views", in *Proc. on Int. Workshop on Synthetic - Natural Hybrid Coding and Three Dimensional Imaging*, Rhodos, September, 1997

[24] Noborio H., Katahira Y., „A Stereo Volume Intersection Method for Reconstruction of 3D Multiple Objects", in Proceedings of the 1992 Second International Conference on Automation, Robotics and Computer Vision, pp.CV-5.3.1-CV.5.3.5, September 1992

[25] Pollefeys M., "Tutorial on 3D modeling from images", in *ECCV*, Dublin, Ireland, 2000

[26] Radke R. J., Andra S., Al-Kofahi O., Roysam B., „Image change detection algorithms: A systematic survey", in *IEEE Transactions on Image Processing*, 2004

[27] Schulz O., „Image-based 3D-surveillance in Human-Robot-Cooperation", in *Modern Trends in Manufacturing*. Second International CAMT Conference, Wroclaw, Poland: Oficyna Wydawnicza Politechniki Wroclawskiej, pp.327-34, Feb. 20-21, 2003

[28] Som F., „Sichere Steuerungstechnik für den OTS-Einsatz von Robotern", in *4. Workshop für OTS-Systeme in der Robotik* (Sichere Mensch-Roboter-Interaktion ohne trennende Schutzsysteme), Stuttgart, Nov. 2., 2005

[29] Spingler J., Thiemermann S., „Direkte Mensch-Roboter Kooperation in der flexiblen Montagezelle", in *Robotik 2002*, VDI-Berichte Nr. 1679, 2002, pp. 191-195

[30] Thiemermann S., *Direkte Mensch-Roboter-Kooperation in der Kleinteilmontage mit einem SCARA-Roboter*, Dissertation, University of Stuttgart, 2005

[31] Thiemermann, S.: „team@work – Direkte Mensch-Roboter Kooperation", in *OTS-Systeme in der Robotik*, Reihe BKM Berichte, Herbert Utz Verlag, München, 2002, pp. 4.1 – 4.5