

Multi-Sensor Obstacle Tracking for Safe Human-Robot Interaction

Christian Frese, Angelika Fetzner, Christian Frey
Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB, Karlsruhe, Germany

Abstract

For human-robot interaction in industrial applications, monitoring the robot surroundings is essential in order to guarantee the safety of humans sharing the workspace with robots. This contribution presents an approach for obstacle detection and tracking based on the fusion of multiple heterogeneous depth sensors which are mounted on board a mobile manipulator. The proposed methods can track arbitrary obstacles all around the robot. Furthermore, a novel method for extrinsic calibration of the sensors is proposed, using the manipulator as a sensor calibration target. This approach enables a robust and accurate calibration without the need for a dedicated calibration object. Experimental results validate the real-time performance of the proposed obstacle tracking method.

1 Introduction

Traditionally, humans and robots operate in separate workspaces protected by fences or light curtains. Many industrial applications could however benefit from removing these barriers, enabling a closer human-robot cooperation within the same workspace. For example, mobile manipulation robots can supply parts to workbenches where human workers assemble specialized products, or perform welding and riveting operations jointly with humans. A prerequisite for such human-robot interaction is the sensor-based monitoring in order to guarantee the safety of humans. The following two scenarios are considered to demonstrate the feasibility of this approach:

1. The robot autonomously performs some supply tasks in a joint workspace with human workers and uninvolved persons. The robot must avoid collisions with humans.
2. Additionally, intended physical human-robot interaction occurs, e.g., the robot hands over some part to the human.

This contribution proposes a concept for monitoring the workspace of mobile manipulators relying solely on sensors on board the robot. Multiple heterogeneous depth sensors with partially overlapping fields of view are applied to monitor the workspace of both the manipulator and the mobile platform. The fusion of the information obtained by the different sensors is performed by mapping the detected obstacle points and the computed features into a 2D grid structure. The proposed object tracking algorithm is designed specifically for the integration of information from heterogeneous sensors. For collision prevention, future positions of tracked moving obstacles are predicted.

This paper is structured as follows: related work regarding both obstacle tracking and sensor calibration is outlined in Section 2. The sensor setup and the extrinsic calibration methods are described in Section 3. Section 4 presents the preprocessing of the sensor data to detect obstacle points and the grid structure used for information fusion. The sensor-independent object tracking method consisting of clustering, association, state estimation, and prediction is described in Section 5. Experimental results obtained by applying the proposed methods to the robot are shown in Section 6. Finally, conclusions are presented in Section 7.

2 Related Work

Workspace monitoring methods usually rely on stationary sensors mounted, e. g., at the ceiling of the room in order to ensure safe operation of a stationary industrial robot [1, 2]. It is not clear whether such methods can be adapted for monitoring the workspace of mobile robots in unrestricted environments.

A wide variety of object tracking methods has been proposed for applications in robotics [3] and intelligent vehicles [4, 5]. They can be categorized, among other criteria, in 2D and 3D approaches, and in methods for general objects [6] or specific ones, e.g., vehicles or pedestrians [7]. Tracking of vehicles observed simultaneously by a 3D lidar and a color camera is described in [8]. The object motion is estimated by aligning the point clouds of subsequent measurements. The search in the 6D space of possible alignment transformations is feasible owing to efficient pruning methods.

If multiple sensors are required to achieve the desired spatial coverage, homogeneous sensors with complementary fields of view are usually employed [9], which facilitates the association compared to the heterogeneous sensor

setup used in this contribution.

For the extrinsic calibration of multiple heterogeneous sensors, usually special-purpose calibration objects are used. In several publications, the relative calibration of a 2D lidar and a camera is obtained by matching the line observed by the lidar towards a planar calibration target seen by the camera [10, 11]. For the Kinect sensor, which provides a color camera in addition to the depth sensor, a common calibration approach is to apply standard camera calibration patterns to the color camera instead of directly calibrating the depth sensor. Such calibration targets have also been attached to a manipulator in order to automatically acquire sensor data of the target in different poses required for a unique solution of the calibration problem and to calibrate the sensor relative to the manipulator [12, 13].

3 Sensor Setup and Calibration

3.1 Selecting a Sensor Setup for a Mobile Manipulator

As the mobile robot may have a large workspace, using on-board sensors is preferable for covering the relevant environment with a reasonable number of sensors. For ensuring safety, mainly the distance between robot and obstacles is of interest. Therefore, depth sensors which directly measure distances are well suited for the considered scenario.



Figure 1: Placement of depth sensors on a mobile manipulator.

The robot platform is equipped with two 2D lidar sensors (laser scanners) which scan a horizontal plane near the ground floor. Additional 3D depth sensors are applied to capture the workspace of the manipulator. In the current sensor setup, two Kinect depth cameras are mounted at the rear of the platform (see **Figure 1**). Their position

and orientation has been chosen by means of simulation in order to maximize their field of view and to reduce potential occlusions [14]. In many applications, these sensors cover the relevant portions of the workspace and thus enable the demonstration of safe human-robot interaction. In the future, the sensor setup may be extended to a full 3D surveillance of the workspace, while the proposed fusion and tracking methods remain applicable.

The position of the 2D lidar sensors relative to the robot coordinate system is known by construction with sufficient accuracy, whereas the 3D depth cameras are mounted on adjustable pan/tilt joints. Thus, their exact position and orientation relative to the coordinate system of the mobile platform has to be estimated by an extrinsic calibration procedure.

3.2 A Conventional Extrinsic Calibration Approach

For comparison with the novel method proposed in the following subsection, a conventional calibration approach is briefly sketched [15]. It uses the large tripod shown in **Figure 1** as a calibration target for the extrinsic calibration of a 3D depth camera relative to a 2D lidar. The calibration target is placed manually in different poses where it has to be visible to both sensors. The points measured for the two front legs of the tripod are used for the registration. A plane containing the two legs is fitted to the point cloud acquired by the 3D depth camera. Finally, a transformation is estimated which minimizes the distance between the points detected by the lidar and the plane obtained from the 3D depth camera data.

While this calibration procedure works fine in principle, it is difficult to place the calibration target for the described sensor setup because the manipulator occludes many desirable poses of the tripod.

3.3 Using the Manipulator as a Sensor Calibration Target

A novel calibration procedure is proposed which avoids most of the drawbacks of the conventional approach. This method uses the robotic manipulator arm as a calibration target. The reference is provided by a 3D model of the manipulator, instead of resorting to the lidars as reference sensors. The proposed approach is motivated by the application: as the distance of objects to the arm is the main concern, the calibration between sensor and arm should be most accurate. By contrast, a conventional calibration target cannot be placed in the relevant part of the workspace because the arm will obstruct or occlude it, resulting in a lower accuracy of the calibration exactly where the highest accuracy is desired.

Our method starts with acquiring sensor data containing the manipulator arm. The robot is placed in free space so that no other objects are located in the vicinity of the arm.

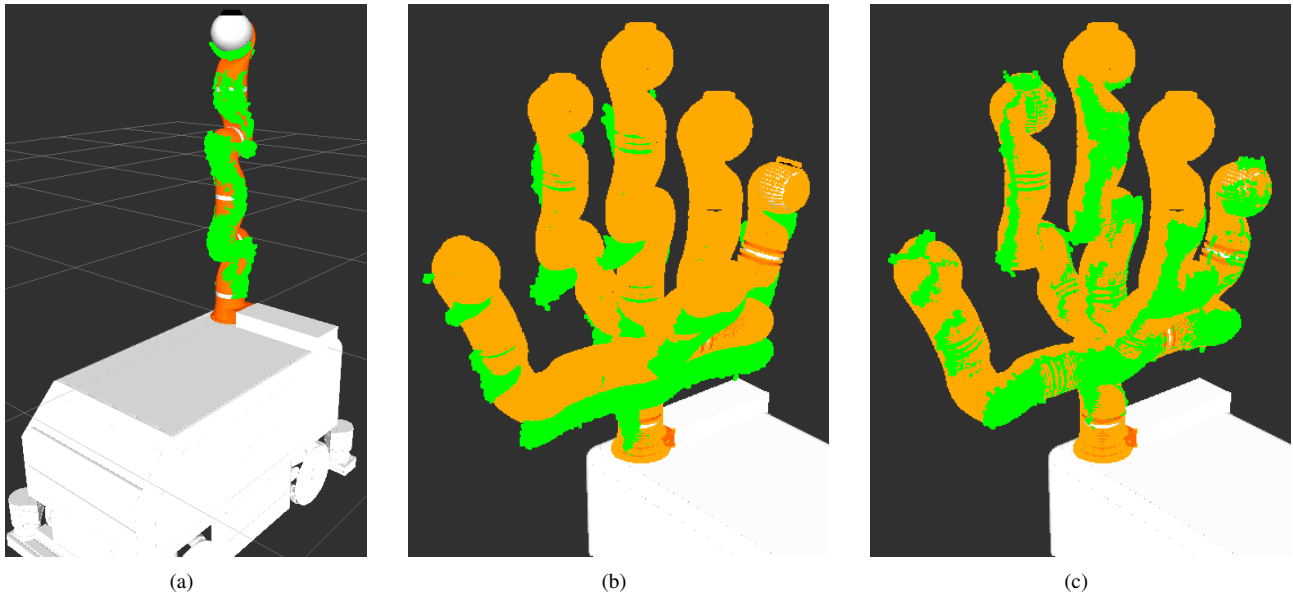


Figure 2: Sensor calibration using the manipulator as a calibration target. The point cloud acquired by the 3D depth camera is depicted in green, while the reference point cloud extracted from the model is shown in orange. (a) Calibration using a single manipulator pose, (b) data aggregated from a sequence of five poses, (c) point clouds fitted by the GICP algorithm.

The point cloud obtained from the depth camera is represented in a coordinate system $\mathbf{T}_{\text{initial}}$ given by a rough guess of the sensor position relative to the robot coordinate system. This point cloud is registered to an accurate 3D model of the arm using the generalized iterative closest point (GICP) algorithm [16, 17]. Thereby, a coordinate transformation $\mathbf{T}_{\text{align}}$ is obtained. The transformation representing the extrinsic sensor calibration is then given by

$$\mathbf{T}_{\text{extrinsic}} = \mathbf{T}_{\text{align}} \mathbf{T}_{\text{initial}}. \quad (1)$$

Figure 2(a) illustrates an example of an acquired point cloud fitted to the robot model.

In order to cover a wide field of view and to reduce ambiguities resulting from partial symmetry of the arm, point clouds of several different arm poses can be incorporated in the registration process. The arm is moved sequentially to a number of predefined positions. In each position, a sensor point cloud is acquired (**Figure 2(b)**). In order to reduce sensor noise, temporal median filtering is applied to each pixel in a series of depth images before computing the point cloud. Additionally, a reference point cloud is extracted from the model. The model encompasses the 3D geometry and the kinematics of the arm so that the extracted point cloud represents the geometry of the arm given its current joint angles. After completing the sequence of positions, the resulting aggregated sensor point cloud is fitted to the aggregated reference point cloud (**Figure 2(c)**). The whole data acquisition and registration procedure is fully automated so that a frequent re-calibration is possible without human intervention.

4 Obstacle Detection based on Multi-Sensor Data

4.1 Design Considerations

To ensure a safe operation of the robot, the design of the algorithms has to be based on conservative assumptions. In particular, we decided not to apply any human detection methods because they might induce some amount of missed detections and delay. Instead, all kinds of obstacles are handled in a generic way. This enables collision avoidance with arbitrary stationary and moving obstacles, e. g., vehicles, forklifts, and other robots. This approach can also cover cases in which human detection is difficult, e. g., when a person is carrying a large object or is partially occluded by other obstacles.

However, this design implies that all stationary and moving objects visible for the sensors have to be tracked simultaneously. Thus, the applied methods have to be computationally efficient.

For a consistent handling of obstacles in the complete surroundings of the robot, the obstacle detection and tracking methods have to integrate information from different sensors. Obstacles have to be tracked continuously while crossing the fields of view of the heterogeneous sensors. This is quite challenging as the employed 2D and 3D sensors differ considerably regarding point density, field of view, resolution, and noise. Therefore methods are proposed which are largely independent of the sensor characteristics. Only the preprocessing of the obstacle points is specific for each type of sensor.

4.2 Preprocessing of 2D Lidar Data

Two 2D lidar sensors are mounted at opposite corners of the robot platform. Each sensor has a horizontal field of view of 270° so that a plane near the floor is completely covered by the two sensors. At the margins of the field of view, some rays which detect parts of the robot platform are removed from the sensor data. Additionally, outliers occurring at depth discontinuities are filtered out. All remaining object points detected by the rays of the lidar scanner represent obstacles.

As the point data acquired by the 2D lidars is quite sparse, the obstacle points from several subsequent scans are aggregated. This means that in addition to the current scan, the data from a certain number of preceding measurements is used after an appropriate correction considering the robot's ego-motion. The data aggregation helps to alleviate fluctuations caused by sensor noise or occlusions. It is also very useful when tracking walking humans, as only the feet are visible to the sensor so that the integration over the foot motions allows for a more stable estimation of the humans' positions.

4.3 Detection of Obstacle Points in 3D Depth Data

In the sensor setup described in Section 3.1, the 3D depth cameras observe the surroundings of the manipulator. Hence, parts of the manipulator are visible in the sensor data. During workspace monitoring, it is necessary to distinguish these robot points from obstacle points. Therefore the robot points are removed from the depth camera data based on the robot model and the current joint angles. This is achieved by applying the *Realtime URDF Filter* [18] to the depth image. Afterwards, the 3D point cloud of the remaining object points in the robot coordinate system is computed using the extrinsic calibration parameters. These points represent objects different from the robot itself, but not all of them are obstacles relevant for collision avoidance. Especially, the ceiling of the room is detected by the 3D depth cameras. As it is not reachable by the robot, it is not considered to be a relevant obstacle. In principle, the same holds for points on the ground floor, but the floor is not visible in the current sensor configuration. Altogether, only points having a z coordinate (height) above the ground floor and below the ceiling are considered as relevant obstacle points.

4.4 Information Fusion in a Grid Structure

The fusion of information from different sensors has several benefits. First of all, it enables an almost complete coverage of the robot workspace, as the sensors' fields of view are largely complementary. Additionally, occluded regions are reduced significantly because the sensors are positioned strategically at the corners of the platform so that their bearings towards a given obstacle point are as

different as possible. Finally, the probability of missing detections or sensor faults is lowered by using heterogeneous sensors relying on different measurement principles, e.g., lidar and actively illuminated triangulation in the setup described in Section 3.1.

A two-dimensional grid is used as a data structure for the fusion of the obstacle points detected by different heterogeneous sensors. In the 2D grid, each cell is classified as obstacle cell or as free space. The grid basically represents the projection of the obstacle points into the ground plane. The grid cells are enhanced by features computed from the point clouds such as density of points and height above ground. In the area observed by the 3D depth cameras, the grid thus corresponds to a $2^{1/2}D$ map.

The 2D grid structure has been chosen because it provides a computationally efficient way to integrate data points from both 2D and 3D sensors. While a full 3D representation of the obstacles is desirable, it is difficult to associate the 2D data, which lacks height information, with the 3D structure. Therefore a twofold approach is proposed: a 3D octree representation is used for close range obstacle detection and distance computation [14], while the computationally more efficient 2D grid structure is used for wide range object tracking in large workspaces.

The obstacles are represented in the robot coordinate system, with the robot at the center of the grid. For safety applications, the robot-centered representation has the advantage that localization errors do not accumulate into errors in the distance to obstacles.

5 Obstacle Tracking

The proposed tracking method starts with building object hypotheses by clustering the obstacle points detected by the different heterogeneous sensors. The object hypotheses are then associated to existing tracks. Kalman filtering is applied to estimate position and velocity of each tracked object o_i , resulting in a state vector \mathbf{x}_i and a covariance matrix Σ_i . Based on the estimated state and uncertainty, future object positions can be predicted for collision avoidance. **Figure 3** illustrates the data flow between the processing modules.

5.1 Clustering

The obstacle tracking is based on the information from the annotated grid described in Section 4.4. The connected components of obstacle grid cells are computed according to the 8-adjacency [5, 19]. For each connected component, an object hypothesis h is created and annotated with a feature vector aggregating the information from the clustered grid cells. The computed features encompass the position of the object centroid, the height of the object, the number of obstacle points detected by each sensor, and the 2D shape, i. e., the list of the grid cells occupied by the cluster.

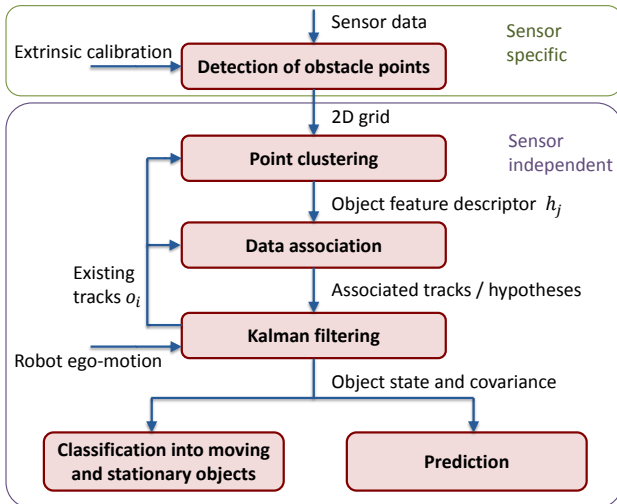


Figure 3: Processing pipeline for obstacle detection and tracking.

However, the clusters sometimes are not stable over time. This is due to fluctuations in the sensor data arising from sensor noise or occlusions. Especially in the regions only observed by the 2D lidars, the point density may be very low so that it is challenging to achieve a stable clustering.

This contribution proposes to compare the clustered object hypotheses with the grid cells occupied by already tracked objects in order to increase the stability of the clustering over time. First, the tracked objects are transformed into the coordinate system of the current grid by taking into account both the velocity estimate of the object and the ego-motion of the robot. Then, the number of overlapping grid cells between the current hypotheses and the existing tracks is evaluated. For example, if for two hypotheses h_j and h_k the percentage of grid cells overlapping with the grid cells of the same tracked object o_i exceeds a certain threshold, an additional hypothesis is created which represents the union of the two clusters. Similarly, composite hypotheses encompassing more than two clusters can be created. Thus, an oversegmentation caused by occlusions, by noise or by the sparsity of the sensor data can be avoided.

Conversely, if a cluster covers more than one tracked object, it may be useful to split the cluster. For example, if the tracked objects o_i and o_k overlap with the same hypothesis h_j to a large extent, two additional hypotheses are created which consist of the grid cells of h_j being closer to the cells occupied by o_i and o_k , respectively. This step is important for successfully tracking a person walking nearby a stationary object, e. g., a table. **Figure 4** illustrates the clustering process by means of an example.

The hypotheses created for composite or split clusters do not replace the original hypotheses obtained by the clustering algorithm, but represent additional hypotheses to be considered. The decision whether to keep the original or the additional hypotheses is made during the association step.

5.2 Association

The association of an object hypothesis h to an already tracked object o is performed by means of a distance function $d(h, o)$ which rates the differences of the two corresponding feature vectors. The distance is evaluated for all pairs (h_j, o_i) located within a reasonable spatial neighborhood. The neighboring objects for a given hypothesis can be found efficiently using a k -d tree structure [20, 17]. Then the pairs (h_j, o_i) are selected for association in the order of ascending distance. Once a hypothesis h has been chosen, all other candidate pairs (h, o_i) for the same hypothesis and also for all composite hypotheses containing h are invalidated so that each detected cluster is associated to at most one tracked object.

The distance function d has to be carefully designed to account for the heterogeneity of the sensors used in the fusion step. For example, the number of detected obstacle points will differ considerably if h consists of 2D lidar data while o is based on 3D depth data. So the distance function needs to disregard some features depending on the observing sensors, while exploiting the full feature information for association if the same sensors have detected both o and h . This approach enables a reliable association also at the border of the sensors' fields of view.

In more detail, the distance function is the sum of a generic and a sensor specific term,

$$d(h, o) := d_G(h, o) + d_S(h, o). \quad (2)$$

The generic distance function includes the squared Mahalanobis distance of the hypothesis to the object, i. e., the distance in the state space is weighted by the inverse of the covariance matrix, $(\mathbf{x}_h - \mathbf{x}_o)^T \Sigma_o^{-1} (\mathbf{x}_h - \mathbf{x}_o)$. Furthermore, a term assessing the difference in the number of occupied grid cells is added.

The sensor specific distance function is defined as follows,

$$d_S(h, o) := \begin{cases} \frac{1}{|\mathcal{S}_h \cap \mathcal{S}_o|} \sum_{k \in \mathcal{S}_h \cap \mathcal{S}_o} d_k(h, o), & \mathcal{S}_h \cap \mathcal{S}_o \neq \emptyset \\ d_P, & \mathcal{S}_h \cap \mathcal{S}_o = \emptyset, \end{cases}$$

where \mathcal{S}_h and \mathcal{S}_o denote the set of sensors that have detected h and o , respectively, and d_P is a constant penalizing cases in which hypothesis and object have no common detecting sensor. For each sensor k the function $d_k(h, o)$ rates the difference in the number of detected obstacle points, and, for 3D sensors, also the difference in the measured height of the object.

5.3 State Estimation

For estimating and filtering position and velocity of an object, a Kalman filter [21] with a generic linear motion model is employed. The generic model can handle all kinds of obstacles the robot must avoid, e. g., humans, vehicles, and other robots. The four-dimensional state vector $\mathbf{x}(o_i)$ is composed of position and velocity of the object

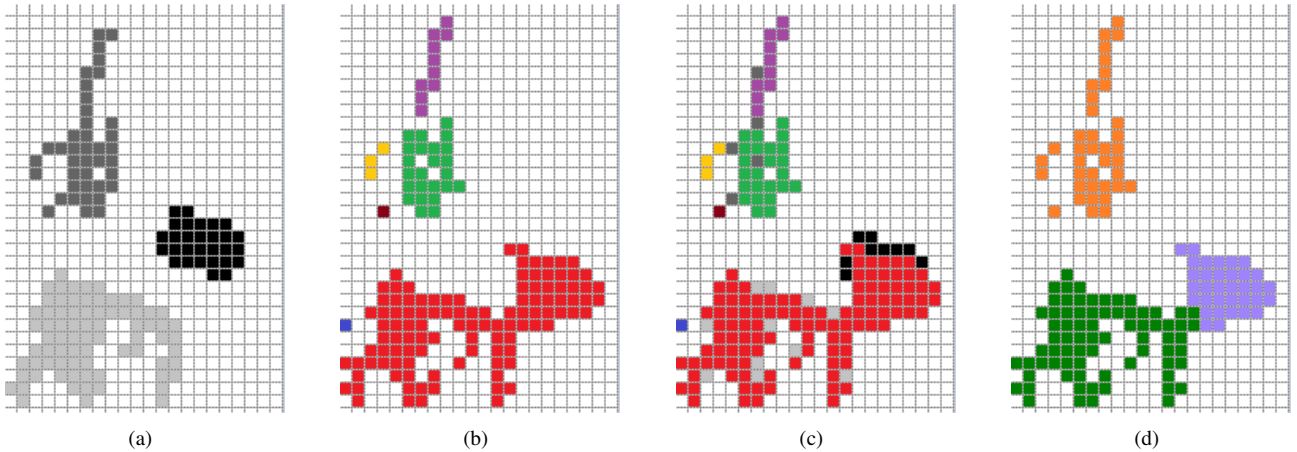


Figure 4: Illustration of the clustering process within the 2D grid structure. (a) Clusters representing the tracked objects at time step $k-1$, (b) clusters obtained from the connected components of the grid at time step k , (c) evaluation of the overlap between the clusters in the grids of (a) and (b), (d) additional hypotheses resulting from the comparison. At the top of the figure, a composite hypothesis is obtained, whereas at the bottom, a cluster is split into two hypotheses.

centroid within the ground plane: $\mathbf{x} = (x, y, v_x, v_y)^T$. The linear motion model is as follows,

$$\mathbf{x}_k = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{x}_{k-1} + \begin{pmatrix} 0 \\ 0 \\ a_x \\ a_y \end{pmatrix}, \quad (3)$$

where Δt is the time since the last measurement and the unknown accelerations a_x , a_y are considered to be normally distributed noise variables. This system model is well-suited for tracking humans who can change their direction of motion very fast and almost arbitrarily. Other obstacles such as vehicles can also be tracked using this generic model, although a more specific model considering the vehicle kinematics would be preferable if an object classification was available.

The Kalman filter algorithm computes the predicted object state and its covariance for the next hypotheses association step. As the positions are represented in a robot-centered coordinate system, the object state has to be corrected by the ego-motion of the mobile robot platform measured by its odometry sensors. Finally, the update step of the Kalman filter is performed using the position of the cluster h selected in the association step as a measurement.

Since the position is estimated for the centroid of the detected obstacle points, errors may occur if different portions of the object are visible to the sensors over time. This may be the case for an object entering or leaving the field of view as the robot moves or for an object partially occluded by another object moving in front of it. Via this effect, an apparent motion of the object centroid is induced which overlays the true motion of the object. A consequence of this effect might be, for example, that a stationary object is erroneously classified as moving.

In order to eliminate this effect, the object position estimate is corrected based on a shape alignment procedure,

inspired by the method proposed in [8]. The 2D shapes of the hypothesis and the tracked object corrected by the ego-motion of the robot are mapped into the grid. Then, the 2D displacement vector for the hypothesis shape is computed which maximizes the overlap of hypothesis and object cells. This optimization is performed using branch and bound search. The centroid position of the tracked object is corrected by the computed displacement.

5.4 Prediction

Obstacles can be classified into moving and stationary objects based on the velocity estimated by the Kalman filter. This enables the robot to act more conservatively in the vicinity of moving objects.

The Kalman filter algorithm can also compute a longer-term prediction of the object position with uncertainties represented in the covariance matrix. The 2D shape of moving objects is mapped into a grid at the predicted positions. The object shape is enlarged according to the increasing uncertainty predicted in the covariance matrix. Additionally, the robot and its planned trajectory are mapped into the grid. Then, the time to collision and the distance between the robot and a moving obstacle can be computed. Depending on the results, the robot is slowed down or stopped if it is necessary to prevent a collision.

6 Experimental Results

6.1 Calibration

The performance of the extrinsic sensor calibration method proposed in Section 3.3 has been evaluated as follows: the manipulator has been moved to several distinct poses different from the calibration poses and the Realtime URDF

Filter mentioned in Section 4.3 has been applied to the depth images acquired by the 3D depth sensors. Then the number of robot points has been counted which have not been filtered out due to misalignment between the sensor data and the model.

The calibration method using the manipulator as a calibration target has proved to be accurate and very robust regarding parameter settings. The GICP algorithm reports a root mean square (RMS) alignment error of about 7 mm. Calibration using a single manipulator pose can already yield very good results for certain poses, while the results are somewhat less accurate for some other poses. The quality of the results is more predictable when using a sequence of poses as shown in Figure 2(b). When compared to the standard ICP algorithm, the GICP algorithm which takes into account that the point registration within a plane is more uncertain than along the normal of the plane provides slightly better results and a larger region of convergence with respect to the initial guess of the transformation.

However, the differences for the various poses, sequences and algorithm variants were rather small: in the worst case, only about 50 % more robot points erroneously remained in the depth image compared to the best case. For obstacle detection, these points are removed by using a robot model with slightly enlarged links in the filtering step.

By contrast, the conventional calibration described in Section 3.2 could not achieve the required accuracy: the number of robot points which could not be filtered out was almost one order of magnitude larger. This is not caused by a general problem of the conventional calibration method, but can be attributed to the specific geometrical configuration in which the robot obstructs the placement of the tripod calibration target and thus ill-conditioned data results. For a different sensor placement without obstructing objects, the reprojection error computed by the calibration method has been about one order of magnitude lower.

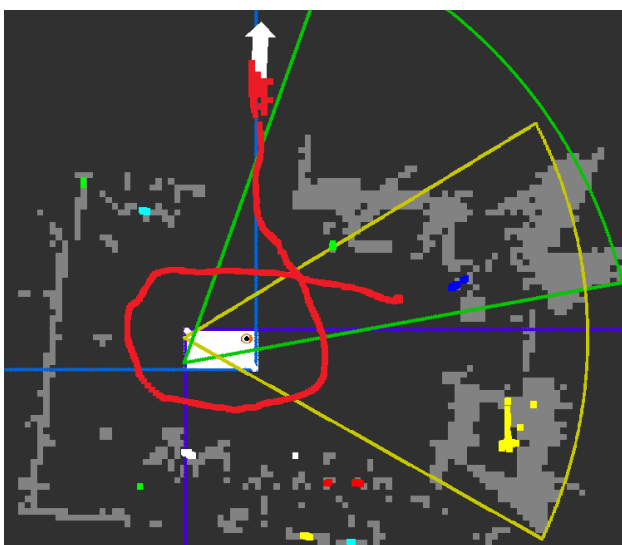


Figure 5: Tracking of a person walking all around the robot.

6.2 Obstacle Tracking

The proposed approach has been implemented and validated on board the robot in the setting of the scenario described in Section 1. Its computational efficiency allows to track in the order of hundreds of objects simultaneously in real-time at the data rate of the lidar sensors. The method enables reliable detection and tracking of moving obstacles even when they cross the fields of view of different heterogeneous sensors. The proposed extensions such as composite hypotheses and shape alignment considerably increase the tracking performance and reduce the number of false positives compared to the baseline algorithm.

Figure 5 shows an example of a person entering and leaving the fields of view of all sensors multiple times while walking all around the robot. The person is continuously tracked. The estimated positions are indicated by the red curve. The currently sensed 2D shape of the walking person is also shown in red at the top of the image, with the white arrow illustrating the estimated velocity. The sensors' fields of view are depicted as sectors in the colors corresponding to Figure 1. Stationary obstacles are shown in gray.

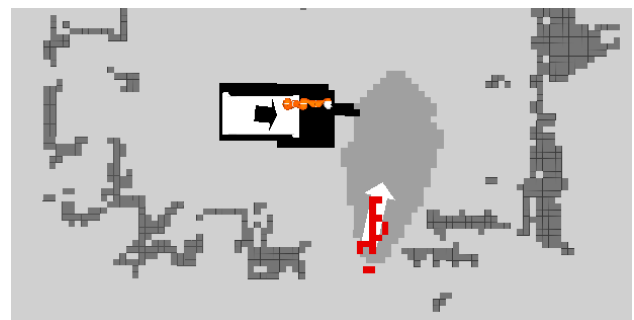


Figure 6: Prediction of obstacle and robot positions for collision prevention.

Figure 6 illustrates prediction-based collision prevention by an example recorded in the setting of the mentioned scenario. As the robot moves forward, a human crosses its way. The current position of the human is colored red. Arrows indicate the velocities of human and robot. The planned trajectory of the robot – including both mobile platform and manipulator – is visualized in black, while the predicted position of the human is depicted gray. As the two predicted areas touch in this instant of time, the robot is slowed down and stopped subsequently in order to prevent a collision with the pedestrian. Clearly, this behavior is enabled by the tracking-based prediction – it would not have been possible if all obstacles had been assumed to remain stationary at their current position.

To validate that the tracking algorithm is largely independent of the specific sensor characteristics, it has been tested on data acquired by a Velodyne 3D lidar. Pedestrians could be tracked successfully at distances of more than 40 m to the sensor.

7 Conclusions and Future Work

This contribution has presented a workspace monitoring concept specifically designed for the requirements of a mobile manipulator. A 2D grid structure is used for fusion of the information obtained by multiple heterogeneous depth sensors mounted on board the robot. The sensors are calibrated by a reliable, accurate and easy-to-use procedure using the manipulator as a calibration target. By means of a carefully designed association function, objects are continuously tracked while passing the fields of view of different heterogeneous sensors.

Future work includes integrating the obstacle tracking more tightly with motion planning and control, which will ultimately enable the robot to perform evasive motions avoiding moving obstacles.

Acknowledgement

This work has been funded by the European Commission's 7th Framework Programme as part of the project SAPHARI under grant agreement ICT-287513.

References

- [1] C. Vogel, M. Poggendorf, C. Walter, and N. Elkmann. Towards safe physical human-robot collaboration: A projection-based safety system. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3355–3360, Sept. 2011.
- [2] M. Fischer and D. Henrich. 3D collision detection for industrial robots and unknown obstacles using multiple depth images. In T. Kröger and F. M. Wahl, editors, *Advances in Robotics Research - Theory, Implementation, Application*. Springer, 2009.
- [3] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers. People tracking with mobile robots using sample-based joint probabilistic data association filters. *Int. Journal of Robotics Research*, 22, pages 99–116, 2003.
- [4] M. Montemerlo et al. Junior – the Stanford entry in the Urban Challenge. *Journal of Field Robotics*, 25(9), pages 569–597, 2008.
- [5] M. Himmelsbach, A. Müller, T. Lüttel, and H.-J. Wünsche. LIDAR-based 3D object perception. In *1st Int. Workshop on Cognition for Technical Systems*, 2008.
- [6] F. Moosmann and C. Stiller. Joint self-localization and tracking of generic objects in 3D range data. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1138–1144, 2013.
- [7] J. H. Lee, T. Tsubouchi, K. Yamamoto, and S. Egawa. People tracking using a robot in motion with laser range finder. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 2936–2942, 2006.
- [8] D. Held, J. Levinson, and S. Thrun. Precision tracking with sparse 3D and dense color 2D data. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1130–1137, 2013.
- [9] L. Chen, W. Wang, and A. Knoll. Global optimal data association for multiple people tracking. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 4713–4719, 2013.
- [10] K. Kwak, D. F. Huber, H. Badino, and T. Kanade. Extrinsic calibration of a single line scanning lidar and a camera. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3283–3289, Sept. 2011.
- [11] C. X. Guo and S. I. Roumeliotis. An analytical least-squares solution to the line scan lidar-camera extrinsic calibration problem. In *IEEE Int. Conf. on Robotics and Automation*, pages 2928–2933, 2013.
- [12] P. Rakprayoon, M. Ruchanurucks, and A. Coundoul. Kinect-based obstacle detection for manipulator. In *IEEE/SICE Int. Symposium on System Integration (SII)*, pages 68–73, Dec. 2011.
- [13] S. A. Haug, F. Weisshardt, and A. Verl. Automatic camera and kinematic calibration of a complex service robot. In *Autonomous Mobile Systems*, Informatik aktuell, pages 1–9, 2012.
- [14] A. Fetzner, C. Frese, and C. Frey. A 3D representation of obstacles in the robot's reachable area considering occlusions. In *Joint 45th Int. Symposium on Robotik and 8th German Conf. on Robotics (ISR/ROBOTIK)*, 2014.
- [15] S. Riedmüller. Multi-LIDAR-Hinderniserkennung für einen mobilen Roboter. Bachelor's thesis, 2012.
- [16] A. V. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Robotics: Science and Systems V*, June 2009.
- [17] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*. <http://pointclouds.org>, May 2011.
- [18] N. Blodow. Realtime URDF filter. https://github.com/blodow/realtime_urdf_filter.
- [19] R. M. Haralick and L. G. Shapiro. *Computer and robot vision*, volume 1. Addison-Wesley, 1992.
- [20] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9), pages 509–517, Sept. 1975.
- [21] A. Gelb, editor. *Applied optimal estimation*. MIT Press, 1974.