# SAPHARI

SAFE AND AUTONOMOUS PHYSICAL HUMAN-AWARE ROBOT INTERACTION

SEVENTH FRAMEWORK PROGRAMME

COGNITIVE SYSTEMS & ROBOTICS

Project funded by the European Community's 7th Framework Programme (FP7-ICT-2011-7)
Grant Agreement ICT–287513

## Deliverable D4.2.1

## *Prediction based algorithms for wide range monitoring*

| | |
|---|---|
| **Deliverable due date: 31st October 2014** | **Actual submission date: 31st October 2014** |
| **Start date of project: 1 November 2011** | **Duration: 48 months** |
| **Lead beneficiary for this deliverable: IOSB** | **Revision: 1.0** |

| **Nature: R** | **Dissemination Level: PU** |
|---|---|
| R = Report<br>P = Prototype<br>D = Demonstrator<br>O = Other | PU = Public<br>PP = Restricted to other programme participants (including the Commission Services)<br>RE = Restricted to a group specified by the consortium (including the Commission Services)<br>CO = Confidential, only for members of the consortium (including the Commission Services) |

# Executive Summary

Objective of Task 4.2 is the integration of multiple exteroceptive sensors for the monitoring of large workspaces and the development of model predictive methods for anticipating future dangerous situations.

This deliverable presents algorithms for prediction-based wide range monitoring developed at Fraunhofer-IOSB. A sensor setup has been designed for covering the workspace of a mobile manipulator robot. Obstacles are detected based on the fusion of information from heterogeneous exteroceptive sensors. Using object tracking and state estimation methods, moving obstacles are detected and their future motions are predicted. By comparing the predicted obstacle trajectories with the planned path of the robot, dangerous situations can be anticipated. Experimental results validate the real-time performance of the presented approach.

The methods and results have been published in [Fetzner13a, Frese14a].

# Table of contents

# 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 specialised products, or perform welding and riveting operations jointly with humans. A prerequisite for such human-robot interaction is the sensor-based monitoring of the workspace in order to guarantee the safety of humans. In the SAPHARI project, 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 parts to a human worker.

In WP4, a concept for monitoring the workspace of mobile manipulators has been developed which relies 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 2½D grid structure. The 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.

# 2 Design and calibration of the sensor setup for the mobile manipulator

## *2.1 Sensor setup*

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.

The KUKA omniRob platform used to demonstrate the monitoring algorithms 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 maximise their field of view and to reduce potential occlusions (see SAPHARI Milestone 13). 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, tracking and prediction methods remain applicable.

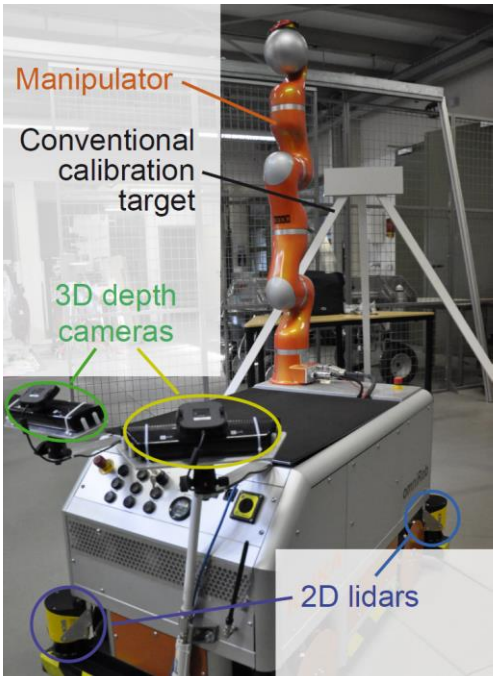

**Figure 1:** Placement of depth sensors on the  mobile manipulator.

## 2.2 Calibration of depth sensors

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.

A novel calibration procedure is proposed which avoids most of the drawbacks of conventional approaches relying on dedicated calibration targets. This method uses the robotic manipulator arm as a calibration target. The reference is provided by a 3D model of the manipulator.

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. 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 generalised iterative closest point (GICP) algorithm [Segal09a, Rusu11a]. Thereby, a coordinate transformation $\mathbf{T}_{\text{align}}$ is obtained. The transformation representing the extrinsic sensor calibration is then given by

$$\mathbf{T}_{\text{extrinsic}} = \mathbf{T}_{\text{align}}\,\mathbf{T}_{\text{initial}}\,.$$

Figure 2 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). 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.

The whole data acquisition and registration procedure is fully automated so that a frequent re-calibration is possible without human intervention.

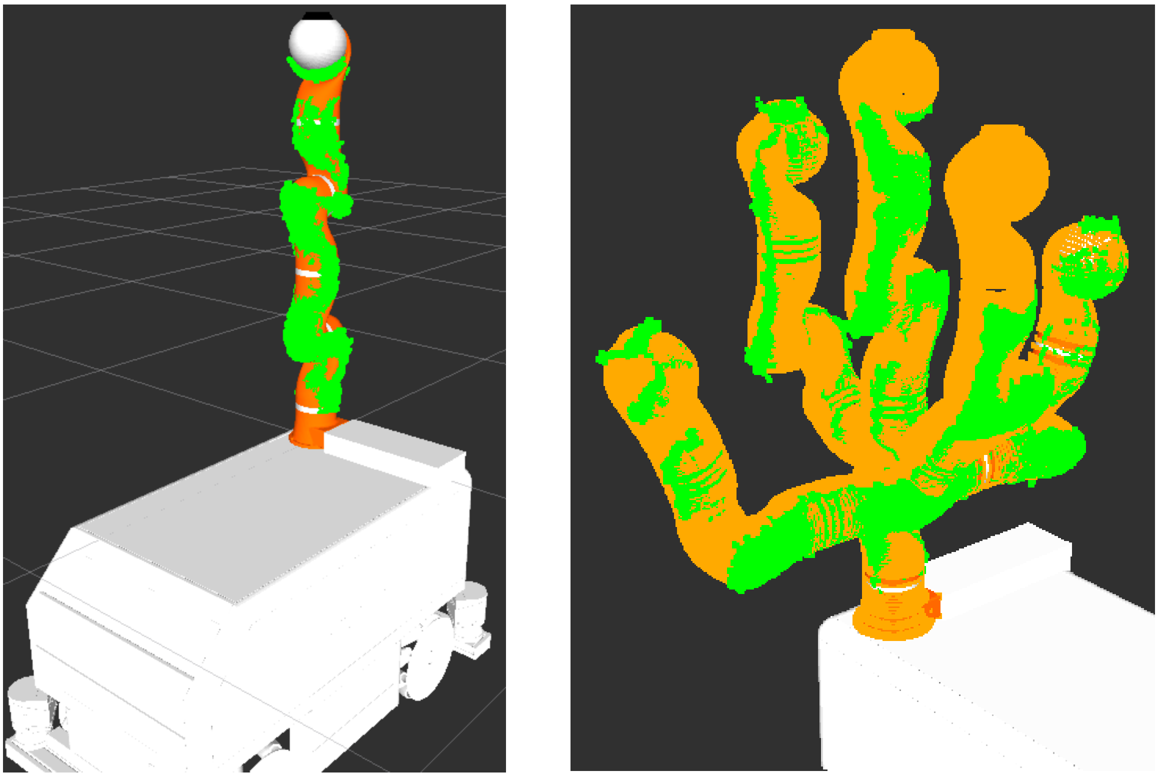

**Figure 2:** Sensor calibration using the manipulator as a calibration target. The point cloud acquired by the 3D depth camera is depicted in green, while the reference point cloud extracted from the model is shown in orange. *Left*: Calibration using a single manipulator pose. *Right*: Point clouds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

## *2.3 Results*

The performance of the extrinsic sensor calibration method has been evaluated as follows: the manipulator has been moved to several distinct poses different from the calibration poses and the misalignment between the sensor data and the model in these poses has been quantified.

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. 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, the misalignment has been only about 50 % larger compared to the best case.

By contrast, a conventional calibration method using a tripod calibration target could not achieve the required accuracy: the misalignment 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 manipulator 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.

More details on the evaluation method have been reported in [Frese14a].

# 3 Wide range obstacle detection based on multi-sensor data

## *3.1 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.

## *3.2 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 occuring 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.

## 3.3 Detection of obstacle points in 3D depth data

In the sensor setup described in Section 2.1, 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 (Figure 3). This is achieved by applying the Realtime URDF Filter from WP4.3 to the depth image (see SAPHARI Milestone 5). 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.

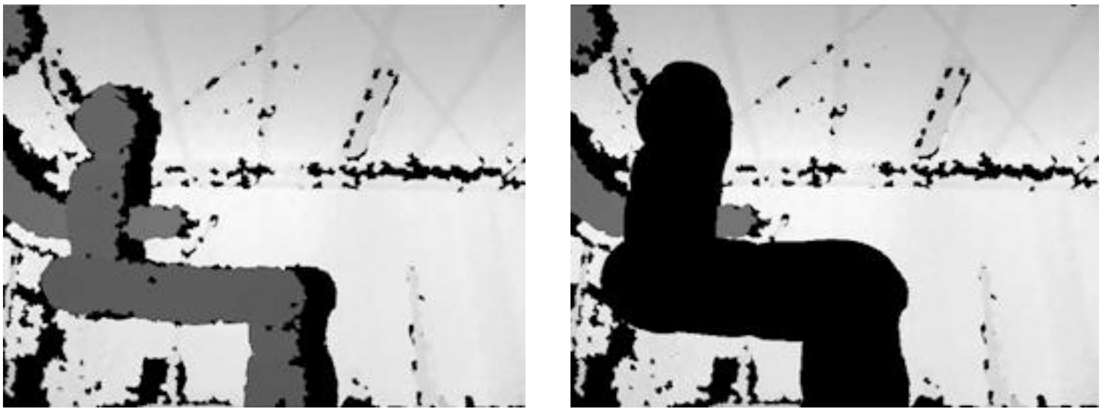

**Figure 3:** Removal of robot points from depth images using the Realtime URDF Filter. *Left*: Acquired depth image of the robot manipulator and a human reaching into the workspace. *Right*: Depth image after removal of the robot points according to the URDF model.

## 3.4 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 missed 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 2.1.

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 2½D 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 the twofold approach of close and wide range workspace monitoring has been chosen: a 3D octree representation is used for close range obstacle detection and distance computation (see Deliverable 4.1.1), 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 localisation errors do not accumulate into errors in the distance to obstacles.

# 4 Wide range 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 $\mathbf{\Sigma}_i$. Based on the estimated state and uncertainty, future object positions can be predicted for collision avoidance.

Figure 4 illustrates the data flow between the processing modules.

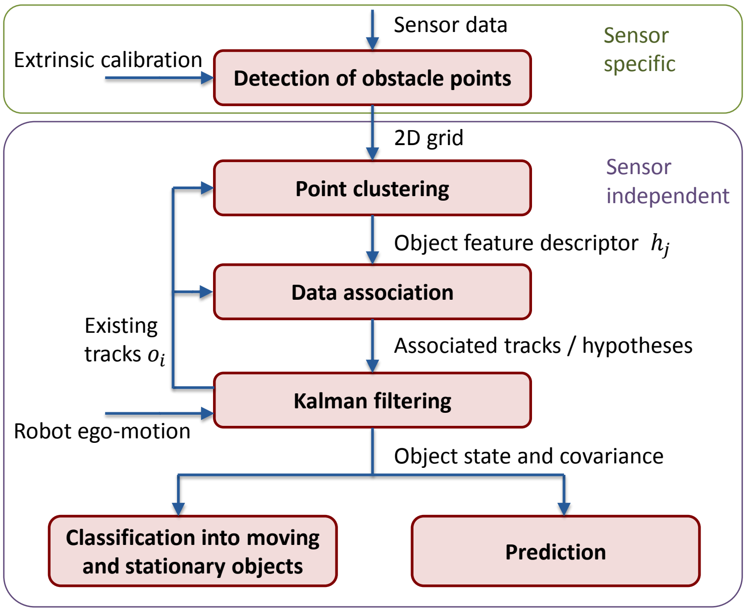

**Figure 4:** Processing pipeline for obstacle detection and tracking.

## *4.1 Clustering*

The obstacle tracking is based on the information from the annotated grid described in Section 3.4. The connected components of obstacle grid cells are computed according to the 8-adjacency. 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.

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.

The method delevoped in the SAPHARI project compares 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 the percentage of grid cells overlapping with the grid cells of the same tracked object 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 two 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 objects $o_i$ and $o_k$, respectively. This step is important for successfully tracking a person walking nearby a stationary object, e.g., a table. Figure 5 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.

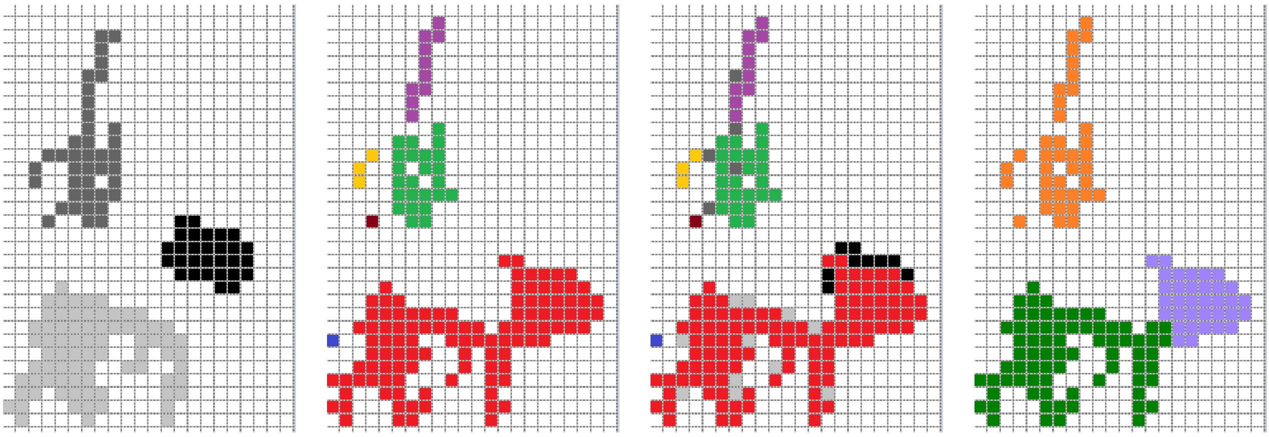

**Figure 5:** Illustration of the clustering process within the 2D grid structure.
*Leftmost image*: Clusters representing the tracked objects at time step $\boldsymbol{k-1}$.
2nd *image*: Clusters obtained from the connected components of the grid at time step $\boldsymbol{k}$.
3rd *image*: Evaluation of the overlap between the clusters in the grids of time steps $\boldsymbol{k-1}$ and $\boldsymbol{k}$.
*Rightmost image*: 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.

## 4.2 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. 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 $d(\cdot,\cdot)$ 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 borders of the sensors' fields of view.

In more detail, the distance function is the sum of a generic and a sensor specific term. 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, $(\mathbf{x}_h-\mathbf{x}_o)^{\mathrm{T}}\boldsymbol{\Sigma}_o{}^{-1}(\mathbf{x}_h-\mathbf{x}_o)$. Furthermore, a term assessing the difference in the number of occupied grid cells is added.

The sensor specific distance function rates, for each sensor, the difference in the number of detected obstacle points, and, for 3D sensors, additionally the difference in the measured height of the object. Associations in which object and hypothesis have no detecting sensor in common are penalized by adding a constant.

Tracks for which no current hypothesis matches are kept as unobserved tracks for a specified number of cycles. This accounts for missed detections, occlusions, etc. New object detections which cannot be associated to an existing track are used to initialise a new track.

## 4.3 State estimation

For estimating and filtering position and velocity of an object, a Kalman filter 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 centroid within the ground plane: $\mathbf{x} = (x, y, v_x, v_y)^{\mathrm{T}}$. The linear motion model is as follows,

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

where $\Delta t$ 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 association step. Since there is no measurement of velocity, it is solely deduced by the filter. 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. 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 maximises the overlap of hypothesis and object cells. This optimisation is performed using branch and bound search. The centroid position of the tracked object is corrected by the computed displacement.

# 5 Prediction-based wide range monitoring

Based on the velocity estimation obtained from the Kalman filter, obstacles can be classified into moving and stationary objects. This enables the robot to act more conservatively in the vicinity of moving objects. In addition to a threshold on the absolute value of the velocity, the covariance matrix is taken into account in order to improve the classification results. Once a track has been classified as moving, it keeps this attribute, as, e.g., a human can stand still for a while and then start moving again.

In typical factory environments, most tracks will represent stationary obstacles such as walls and tables, while only few objects will be moving. Nevertheless, it is necessary to track all of them to be able to distinguish moving and stationary objects, unless a sensor which provides a direct velocity measurement is employed.

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 (see Figure 7). 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 Results

The proposed approach has been implemented and validated on board the robot in the setting of the SAPHARI use cases. Its computational efficency 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 6 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.

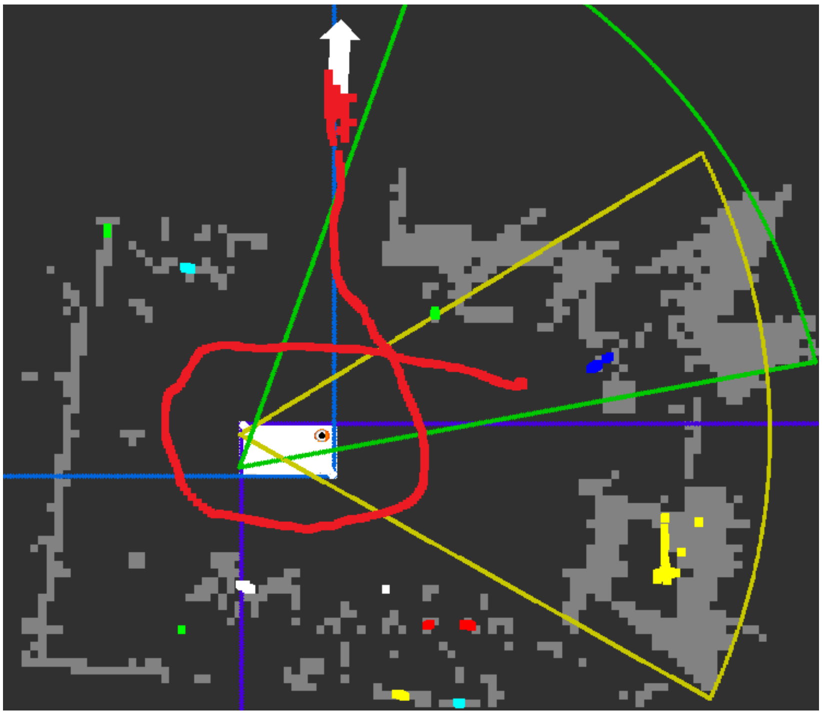

**Figure 6:** Tracking of a person walking all around the robot.

Figure 7 illustrates prediction-based collision prevention by an example recorded in the setting of the mentioned use case. 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 visualised 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.

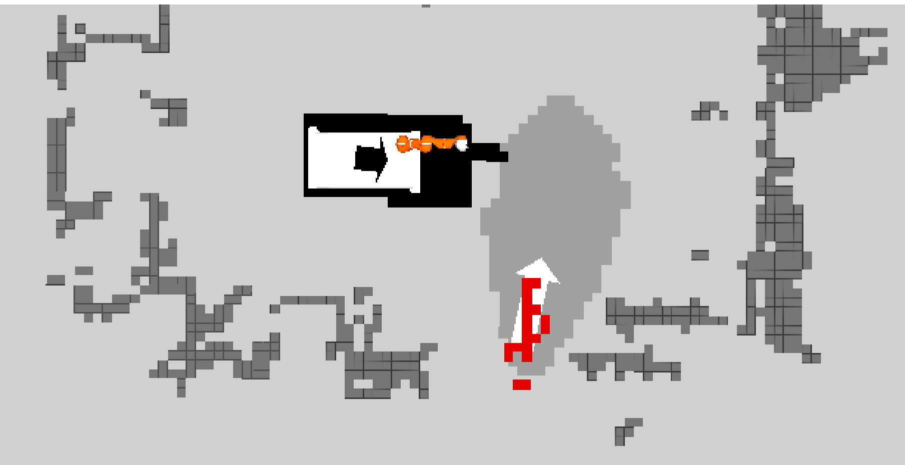

**Figure 7:** Prediction of human and robot positions for collision prevention.

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

# 7 Conclusions

This deliverable has presented a wide range workspace monitoring concept specifically designed for the requirements of a mobile manipulator. A 2½D 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. Based on the estimated object state, future motions of obstacles are predicted in order to anticipate potential collisions with the robot.

The current and predicted distances between robot and obstacles resulting from close and wide range monitoring methods (Tasks 4.1 and 4.2) are used to slow down and stop the robot in oder to prevent collisions. Future work includes integrating the workspace monitoring more tightly with motion planning and control (WPs 6 and 3), which will ultimately enable the robot to perform evasive motions avoiding moving obstacles [Zube14a].

# 8 References

[Fetzner13a] A. Fetzner, C. Frese, and C. Frey, “Obstacle Detection and Tracking for Safe Human-Robot Interaction Based on Multi-Sensory Point Clouds,” Proceedings of the 6th International Workshop on Human-Friendly Robotics, Rome, September 2013.

[Frese14a] C. Frese, A. Fetzner, and C. Frey, “Multi-Sensor Obstacle Tracking for Safe Human-Robot Interaction,” Joint 45th International Symposium on Robotics and 8th German Conference on Robotics (ISR/ROBOTIK), Munich, June 2014.

[Rusu11a] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” IEEE International Conference on Robotics and Automation (ICRA), http://pointclouds.org, May 2011.

[Segal09a] A. V. Segal, D. Haehnel, and S. Thrun, “Generalized-ICP,” Robotics: Science and Systems V, June 2009.

[Zube14a] A. Zube, A. Jung, and C. Frese, “Robot Path Adaptation for Shared Human-Robot Workspaces,” 7th International Workshop on Human-Friendly Robotics, Pontedera, Italy, Oct. 2014.