In: Gearing up and accelerating cross-fertilization between academic and industrial robotics research in Europe - Technology transfer experiments from the ECHORD project.
vol. 94 of Springer Tracts in Advanced Robotics (STAR), pp. 133-153, Springer Verlag, 2014.

Active Recognition and Manipulation for Mobile Robot Bin Picking

Dirk Holz, Matthias Nieuwenhuisen, David Droeschel, Jörg Stückler,
Alexander Berner, Jun Li, Reinhard Klein, and Sven Behnke
{holz,nieuwenhuisen,droeschel,stueckler,behnke}@ais.uni-bonn.de
{berner,lij,rk}@cs.uni-bonn.de

Computer Science Institute, University of Bonn, Germany

Abstract. Grasping individual objects from an unordered pile in a box has been investigated in stationary scenarios so far. In this work, we present a complete system including active object perception and grasp planning for bin picking with a mobile robot. At the core of our approach is an efficient representation of objects as compounds of simple shape and contour primitives. This representation is used for both robust object perception and efficient grasp planning. For being able to manipulate previously unknown objects, we learn object models from single scans in an offline phase. During operation, objects are detected in the scene using a particularly robust probabilistic graph matching. To cope with severe occlusions we employ active perception considering not only previously unseen volume but also outcomes of primitive and object detection. The combination of shape and contour primitives makes our object perception approach particularly robust even in the presence of noise, occlusions, and missing information. For grasp planning, we efficiently pre-compute possible grasps directly on the learned object models. During operation, grasps and arm motions are planned in an efficient local multiresolution height map. All components are integrated and evaluated in a bin picking and part delivery task.

Keywords: Mobile bin picking, contour and shape primitives, active object perception, grasp planning

1 Introduction

Removing individual parts from unordered piles in boxes—*bin picking*—is one of the classical problems of robotics research [1–3]. So far, bin picking robots have been stationary. Typical setups consist of a 3D sensor mounted above the box, an industrial robot arm that is equipped with a gripper, and a data processing unit for detecting objects and planning grasping motions. Sometimes, the sensor is mounted at the manipulator, allowing for close view of selected parts. In order to extend the workspace of the robot and to make bin picking available for environments designed for humans, we implement bin picking using an autonomous

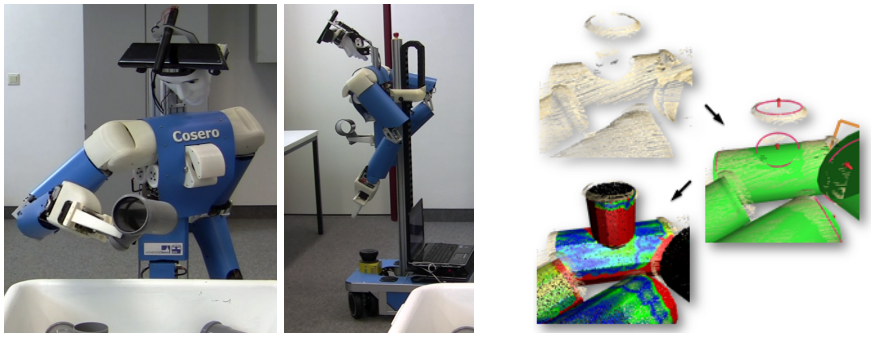


Fig. 1. Mobile bin picking scenario. Objects are grasped from a transport box (left) and placed on a processing station (middle). Both grasp planning as well as object detection and pose estimation are based on contour and shape primitives (right).

mobile manipulation robot. Mobile bin picking is made feasible by the advances in sensing, computing, and actuation technologies, but still poses considerable challenges to object perception and motion planning.

Our scenario is motivated by industrial applications. We consider the task of grasping objects of known geometry from an unordered pile of objects in a transport box as well as transporting it to a processing station and placing it. Solving this mobile manipulation task requires the integration of techniques from mobile robotics like localization and path planning, and manipulation, like object perception and grasp planning. Our robot Cosero has been designed for mobile manipulation and intuitive human-robot interaction tasks, which were tested successfully in RoboCup@Home competitions [4].

While much research investigates the recognition of generic and organic objects in images and point clouds, such methods do not exploit properties of man-made objects that are often encountered in industrial and service robotics applications. Typically, man-made objects are composed of shape primitives such as cylinders, spheres, and planes as well as certain contour primitives (see, for example, the object in Fig. 1). We exploit this property and represent objects as compounds of shape and contour primitives. This representation is used for both robust object perception and efficient grasp planning. For object perception, we follow a popular line of work by matching graph models [5]. Both the object being searched for and the scanned scene are represented as compositions of geometric primitives in graphs. Object hypotheses are generated by identifying parts of a search graph in the graph of a captured scene. Using the established correspondences, one is able to determine the pose of the object in the scene. Here, we extend our previous work [6, 7] by using a combination of contour and shape primitives to increase robustness and reliability.

For being able to manipulate previously unknown objects, we present an approach that allows the robot to learn object models from single scans or CAD models in an offline phase. To cope with severe occlusions, we employ active

perception that considers previously unseen volume and interest in particular regions in the form of primitive and object detection results. The combination of shape and contour primitives makes our object perception approach particularly robust even in the presence of noise, occlusions, erroneous measurements and missing information.

For mobility, we extend global navigation techniques by precise local alignment with the transport box and the processing station. For manipulation, we efficiently pre-compute possible grasps directly on the learned object models. During operation, grasps and arm motions are planned in an efficient local multi-resolution height map. We integrate all components to perform the complete task and evaluate the performance of the integrated system.

2 Related Work

Despite its long history, stationary bin picking is still an active research area. One recent implementation of Papazov *et al.* [8] utilizes a Microsoft Kinect sensor mounted above a table to acquire depth images of the scene. Object models are matched to the measured point cloud by means of geometric feature descriptors and RANSAC [9]. Papazov *et al.* consider tabletop scenes where multiple objects are arranged nearby, including the stacking of some objects. They select the object to be grasped based on the center of mass height (high objects are preferred). Each object is associated with a list of predetermined grasps, which are selected according to the orientation of the gripper. The grasping is performed by a compliant lightweight robot arm with parallel gripper. Bley *et al.* [10] propose another approach of grasp selection by fitting learned generic object models to point cloud data. In contrast to our approach they manipulate separated objects. Choi *et al.* [11] proposed a Hough voting-based approach that extends point-pair features [12, 13], which are based on oriented surface points, by boundary points with directions and boundary line segments. Choi *et al.* use a structured-light 3D sensor mounted on an industrial arm to acquire point clouds of small objects in a transport box, and grasp them with a high success rate. Another extension of Drost *et al.* [13] has been proposed by Kim and Medioni [14]. They consider visibility in between the paired surface elements to sort out false matches.

While the above methods for object detection work best with objects containing distinct geometric features, other approaches rely on the decomposition of point clouds into geometric primitives. The method proposed by Schnabel *et al.* [15] is based on RANSAC and efficiently detects planes, spheres, cylinders, cones, and tori in the presence of outliers and noise. Another work in this direction is Li *et al.* [16], who build a graph of primitive relations and constraints. Assuming symmetry and consistent alignments of shapes in man-made objects, the orientations and positions of detected shapes are iteratively refined. The above approaches require dense depth measurements. In contrast, Liu *et al.* [3] developed a multi-flash camera to estimate depth edges. Detected edges are matched with object templates by means of directional Chamfer matching and

objects are grasped with a three-pin gripper that is inserted into a hole at a success rate of 94 %.

Manipulation in constrained spaces like boxes and shelves requires manipulators with more than the minimal 6 DoF and leads to difficult high-dimensional motion planning problems. Cohen *et al.* [17] proposed a search-based motion planning algorithm that combines a set of adaptive motion primitives with motions generated by two analytical solvers.

Chitta *et al.* [18] proposed an approach to mobile pick-and-place tasks integrating 3D perception with grasp and motion planning. This approach has been used for applications like tabletop object manipulation and the transportation of objects. In these applications, objects stand well-separated on horizontal surfaces or are ordered in feeders. Klingbeil *et al.* [19] utilized a Willow Garage PR2 robot to grasp unknown objects from a pile on a table and read their bar-codes to demonstrate a cashier checkout application. Because the dense packing of objects in a pile poses considerable challenges for perception and grasping, Chang *et al.* [20] proposed pushing strategies for the interactive singulation of objects. Gupta and Sukhatme [21] estimate how cluttered an area is and employ motion primitives to separate piled Lego bricks.

Other systems for which mobile pick-and-place has been realized include HERB [22], developed at the Intel Research Lab Pittsburgh. HERB navigates around a kitchen, searches for mugs and brings them back to the kitchen sink. Rollin' Justin [23], developed at DLR Oberpfaffenhofen, grasped coffee pads and inserted them into the coffee machine, which involved opening and closing the pad drawer. The Armar robots [24], developed at KIT, demonstrated tasks in a kitchen scenario that require integrated grasp and motion planning. In the health care domain, Jain and Kemp [25] present EL-E, a mobile manipulator that assists motor impaired patients by performing pick and place operations. Beetz *et al.* [26] used a PR2 and the robot Rosie, developed at TU Munich, to cooperatively prepare pancakes, combining mobile manipulation and tool-use.

In most of these mobile manipulation demonstrations, the handled objects are well-separated. To the best of our knowledge, mobile bin picking has not yet been realized.

3 Representing, Detecting, and Learning Object Models

In our approach, objects are represented by compositions of contour and shape primitives. We model such compositions by a graph. Detected primitives, and the model parameters describing them, form the vertices of the graph. Spatial relations between neighboring primitives are encoded in the edges. Given the graph of a query object, we convert an input point cloud into a graph modeling the captured scene and find sub-graphs that match the graph of the query object.

3.1 Scan Acquisition and Preprocessing

Detection of Transport Box. In addition to the 3D sensor for part perception, we use three horizontally mounted laser scanners in the robot's base and

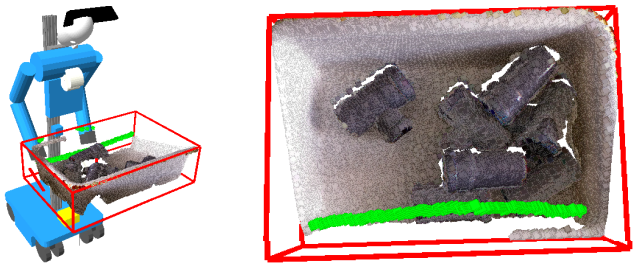


Fig. 2. Two views on a scanned transport box with the extracted line segment (green) corresponding to the box's front side and the estimated transport box model (red).

torso for localization and collision avoidance. One scanner in the robot's torso is mounted slightly above table height and used for detecting transport boxes and aligning the robot to detected boxes (see Fig. 2). To detect a transport box, we continuously extract line segments from the 2D laser scan. We check the straightness of line segments by principle component analysis and neglect those exceeding a given curvature. The closest remaining line segments that fit the dimensions of the transport box correspond to its rim.

Acquisition and Alignment of Point Clouds. For perception of the content of the transport box, we use a Microsoft Kinect camera that is mounted on the robot's pan-tilt unit. To compensate for the sensor's limited field-of-view, we acquire three overlapping point clouds from different views (left, middle, right). For obtaining a consistent scan of the transport box, we register the three point clouds with the Iterative Closest Point (ICP) algorithm [27]. For efficiency, we remove duplicate points from overlapping areas as well as outliers.

Preprocessing. For the detection of geometric primitives we rapidly calculate local surface normals for the residual points. For being able to detect contour primitives we extract all points belonging to sharp edges and occlusion boundaries. We employ the algorithm by Bendels *et al.* [28] that computes, for every point, a contour probability using various criteria: 1) the angle between the query point's normal and the normals of the neighbors, 2) the relative positions of the neighbors to the query points and 3) the shape of the underlying surface at the query point (encoded in the eigenvalues of the covariance matrix).

3.2 Primitive Detection

Shape Primitive Detection. In order to detect simple geometric shape primitives such as planes, cylinders, and spheres, we employ the algorithm by Schnabel *et al.* [29] based on random sampling. It decomposes a point cloud $P = p_1, \ldots, p_N$ into a set of geometric shape primitives ϕ_i with support points S_{ϕ_i} and a set of

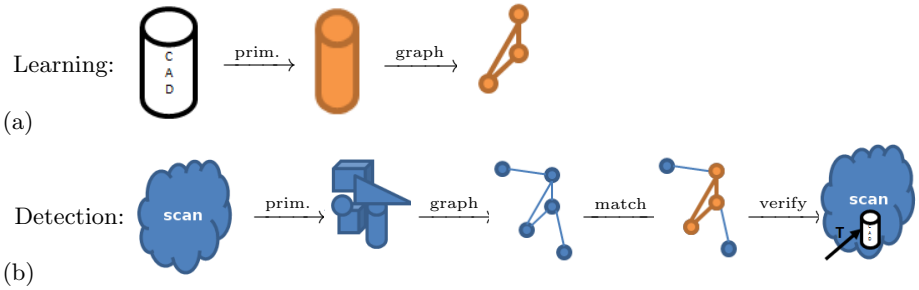


Fig. 3. Object learning and detection: (a) We detect contour and shape primitives and construct a graph encoding their composition. (b) Object hypotheses are obtained from constrained sub-graph matching and verified in the original point cloud.

remaining points R:

$$P = S_{\phi_1} \cup \cdots \cup S_{\phi_A} \cup R. \tag{1}$$

Each support set S_{ϕ_i} is a connected component of points that are 1) close to the primitive (distance $< \epsilon$) and 2) compatible w.r.t. the angle (angle $< \alpha$) between surface normals at the point n_s and on the primitive at the closest point on the primitive $n(\phi_i, s)$:

$$s \in S_{\phi_i} \Rightarrow \|s, \phi_i\| < \epsilon \wedge \angle\left(n_s, n(\phi_i, s)\right) < \alpha. \tag{2}$$

As an important extension to [29], we distinguish two different phases of detecting (shape) primitives: offline learning of new objects and online detection of known objects (see Fig. 3). For learning of new objects given CAD models, we first sample the model uniformly to obtain a 3D point cloud and aim to find an optimal decomposition into arbitrary shape primitives (open model parameters). In the online phase, we save computations by constraining the primitive detection to only find primitives existing in the query object (fixed model parameters).

Contour Primitive Detection. Given the set of points lying on sharp edges and occlusion boundaries in a point cloud, we aim to find contour primitives that help in detecting objects and resolving ambiguities. To this end, we only consider circular contours since our objects of interest either contain cylinders or cylindric holes that result in circular contours. Additionally, as circles have only one shape parameter excluding the position and orientation—the radius—we are able to detect them very robustly in noisy and occluded data. As for the shape primitive detection, we use a RANSAC-based approach and distinguish between online and offline phases. In the offline phase, we fit circle hypotheses by sampling three points, estimating the common plane, and determining the center of the circle through the points. In the online phase, the points are chosen according to the searched radii and efficiently tested using a fast octree implementation that is also used in the shape primitive detection. We only accept hypotheses with a sufficient number of supporting inliers.

3.3 Object Detection

Graph Construction. We efficiently create the annotated shape graph of the scanned point cloud and apply our sub-graph matching approach following ideas of [15]. For each detected shape primitive ϕ_i a vertex is added to the topology graph $G(\Phi, E)$, i.e., $\Phi = \phi_1, \dots \phi_a$. An edge $e_{ij} = (\phi_i, \phi_j)$ is added if the support sets of the primitives ϕ_i and ϕ_j are neighboring, i.e.:

$$\exists p \in S_{\phi_i}, q \in S_{\phi_j} \ : \ \|p - q\| < t \tag{3}$$

with t denoting a distance threshold.

Three types of constraints are encoded in the graph: *node constraints* for the similarity of primitives (model parameters such as type and size), *edge constraints* for the similarity of spatial relations (e.g., the angle between two planes), and *graph constraints* given only implicitly by the topology in the graph (e.g., parallelism of disconnected planes). Detected contour primitives are added to the graph $G(\Phi, E)$ already containing detected shape primitives. False positives in the online contour detection will be pruned in our graph matching approach, as their relative pose is inconsistent with the object model graph.

Graph Matching. The input to our graph matching method is a graph of contour and shape primitives of the query object to be searched. We start with a random edge in the query graph and find similar edges in the scene graph. We compare edges by their relative pose and nodes by their shape properties, and compute a score for each match. This score is zero if the types of primitives do not match and increases with the similarity in relative pose and shape parameters. For each matching edge, we expand the match to adjacent edges (and nodes) in the query and scene graph if they also match in relative pose and shape properties. Expansion is stopped if either the whole query graph matches or no further corresponding edges can be found in the scene graph. The process is repeated in order to find multiple objects in the scene.

Pose Estimation and Verification. We determine object poses from partial matches between the model and scene graph. Depending on the type of the shape primitive, each correspondence determines some of the six degrees of freedom of the pose. A circle-to-circle correspondence, for example, completely determines the translation between the circle centers and two rotational degrees of freedom by the alignment of the circle planes. It does not, however, determine the rotation about the axis perpendicular to the circle plane through its center. Hence, we require several correspondences between primitives until the pose of the object is fully retrieved. For symmetric objects, we can take any transformation around the self-symmetry axis and compute a valid transformation. It is possible to detect all self-symmetries when learning a new object model (cf., [30]).

Computed object poses are refined by registration of the model to the measured points using ICP [27]. As the previous steps only consider the matching

between contour and shape primitive but not their consistency with the overall scan, false-positives may be generated. In a verification step, we check the overlap of object hypotheses with the actual point cloud and remove those with insufficient overlap (15% in our experiments). In case several hypotheses overlap, we remove the hypotheses with lower overlap and only keep the best.

3.4 Learning new Object Models

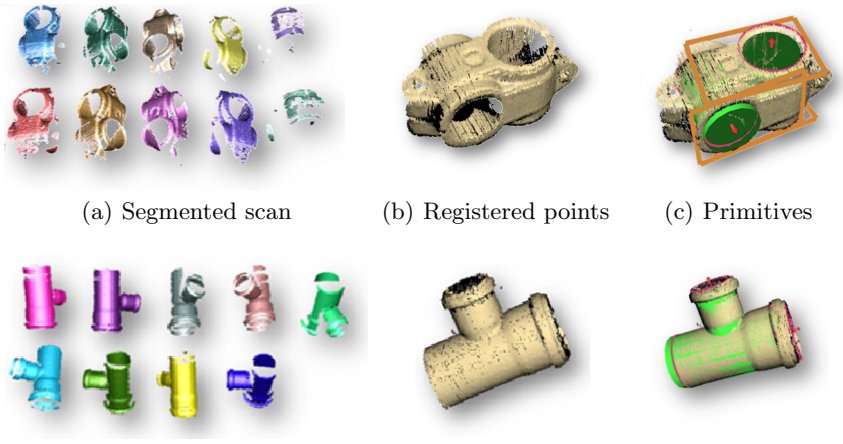
In industrial applications, CAD models of the objects are often available beforehand and can be used for creating new object models and for computing feasible grasps. For applications where no CAD models are available, we propose a simple approach for learning new object models. Multiple object exemplars are placed on a designated learning board in different, roughly known orientations, such that a single scan is sufficient to obtain all information needed.

In a preprocessing step, we compute surface normals and remove outliers. Referring to Fig. 4, we first use the shape primitive detection to find the dominant plane and to register the scanned scene to our model of the board. We then segment all points above the dominant plane and separate them into individual point clouds. The result is equivalent to the same number of scans taken from various viewpoints, with the difference that we know a rough transformation for each scan. In order to obtain a complete 3D model of the exemplar, we register segments with similar orientations in a pair-wise fashion following a coarse-to-fine approach. For an initial alignment, we compute point-pair features [31] on both point clouds to be registered, match the resulting feature descriptors and apply RANSAC to find a consistent set of matches and the corresponding transformation aligning the two clouds. We only allow transformations that are consistent with the ones given by the learning board. The found transformation is refined using ICP [27]. The segmented point clouds are then merged using the estimated inter-pair transformations. For being able to visually inspect the result, we reconstruct the surface of the scanned object using Poisson surface reconstruction [32] and present it to the user.

Regardless of whether we construct a 3D point model of the object as described above or sample the point cloud from the surface of an available CAD model, the final object model is constructed by first detecting shape and contour primitives, and then creating the graph modeling the spatial relationships between the detected primitives. Fig. 4 shows examples of learned object models.

4 Active Object Perception

With increasing geometric part complexity or depth in the pile, large surface parts are likely to be occluded and cannot be acquired from a single viewpoint. A single scan is thus prone to being incomplete and may provide only fragments of the actual part surfaces leading to considerable uncertainty in object detection and pose estimation. In order to achieve robustness of object perception with respect to occlusions, we have developed an active object perception method for



(a) Segmented scan (b) Registered points (c) Primitives

(d) Segmented scan (e) Registered points (f) Primitives

Fig. 4. For learning new object models in the offline phase, we first segment the scan (a+d), then register the residual points (b+e), and detect primitives (c+f).

actively moving the sensor to various view poses. Referring to Fig. 5, we apply the following procedure to explore the transport box and actively detect objects: we register newly acquired scans and update the so far built model of the transport box content. From the model, we can deduce previously occluded volumes. In addition, we represent the outcomes of primitive and object detection in the model as a measure of interest in the corresponding region. Both measures are considered when planning the next view from which a scan is to be acquired. For planning the next best view, we apply a sampling-based approach. We consider the box to be explored and stop the exploration if no more unknown or unseen volume exists within the transport box.

Preprocessing, Registration, and Transport Box Modeling. For fusing and aligning acquired scans, we incrementally build a volumetric model of the transport box (see Fig. 6) and register newly acquired scans. For registration, we have extended an approach to incremental registration from our previous work [33]. The aligned scan points are added to the model while avoiding to add duplicate points. For efficiency, we restrict both the processing of scans and the volume being modeled to the extents of the transport box. From the laser-based box detection and the acquired 3D scan, we deduce an oriented bounding box for the transport box and discard all points lying outside. Moreover, the laser-based measurements of the rim allow us to predict the so far unseen transport box volume. The model is represented as a multiresolution voxel grid map based on [34]. It is organized as an octree with leaves that model multiple attributes of the underlying volume, e.g., the object detection's interest in that region or the volume's occupancy. In addition to constraining the modeled volume to the transport box, we further increase efficiency by performing a lazy evaluation by

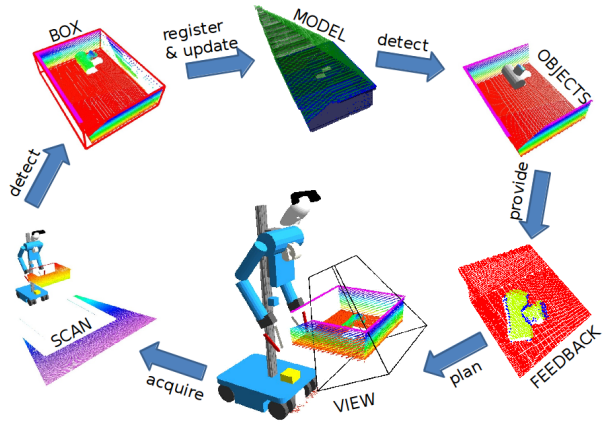


Fig. 5. Active perception pipeline: In a newly acquired scan (bottom), we first detect the transport box, then register the scan and update the so far built model. Detected objects provide feedback for focusing view planning to interesting regions. We then approach the planned view and acquire a new scan.

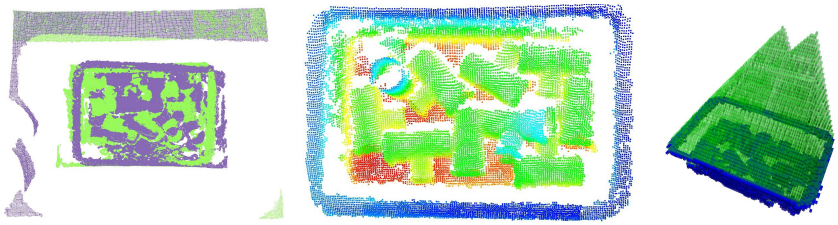
first building the tree, and then recursively updating the content of the changed cells. We employ efficient ray-casting within the bounding box boundaries that allows for fast integration of new measurements and adapting the model to scanned changes in the scene.

Sample Generation and Travel Cost. We apply a sampling-based approach for determining the next best view. We first generate a set of sample view poses $\mathbf{V}$ and then estimate, for every sample view $\mathbf{v} \in \mathbf{V}$, the involved traveling cost $L(\mathbf{v}, \mathbf{r})$ from the robot's current pose $\mathbf{r}$ and the expected information gain $I(\mathbf{v})$. The view with highest utility, i.e., with a high information gain and low traveling cost, is selected as the next best view $\mathbf{v}^\star$:

$$\mathbf{v}^\star = \arg\max_{\mathbf{v} \in \mathbf{V}} I(\mathbf{v})\ e^{-\lambda L(\mathbf{v},\mathbf{r})} \tag{4}$$

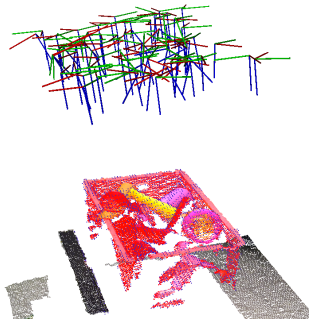
where λ is a parameter to trade off traveling cost and information gain. As approaching a new view pose close to the box only involves local navigation with the robot's omnidirectional base, we approximate the involved traveling cost $L(\mathbf{v}, \mathbf{r})$ by the Euclidean distance between the robot's current pose and the base poses of the sampled views. Typical camera view samples and a summary of the applied scheme for encoding the interest in certain regions are visualized in Fig. 7.

Identifying Regions of Interest. Our approach for detecting contour and shape primitives as well as objects composed of such primitives gives us detailed



(a) Raw input scans (b) Aligned scans (c) Updated model

Fig. 6. Raw input scans (a) are incrementally registered (b) to update a consistent model of the transport box and its content (c). The model distinguishes seen free space (green) and previously unseen volume, and encodes—for each cell—the probability of being occupied and the object detection's interest in this region.



(a) Sampled view poses

Class	Primitive detection	Object detection	Interest
1	None	None	High (0.75)
2	Positive	None	High (1.0)
3	Positive	Positive	Low (0.25)
4	Negative	–	None (0.0)

"Positive": point lies on searched primitive/object
"Negative": point does not lie on searched primitive

(b) Coding *region interest*

Fig. 7. View planning: (a) examples of sample camera view poses; (b) and scheme for encoding the object detection's interest in certain regions.

feedback on all regions in an input point cloud to guide the further acquisition of scans to regions having no or only little confidence in detected objects. Referring to the coding scheme in Fig. 7(b), when a point is detected to lie on a primitive that is not belonging to the object's shape primitive compound we are searching for (class 4), the region around the point is less interesting than regions where object detections are still possible (classes 1 and 2). We model interest as an attribute for each cell in the model in addition to the cell's occupancy.

Information Gain Estimation. Classic approaches to view planning consider previously unseen volume and previously seen surface. Given a model m, the expected information gain $I_{\text{model}}(\mathbf{v})$ from adding measurement z at view pose $\mathbf{v}$ is then approximated by the change in entropy H before and after adding z:

$$I_{\text{model}}(\mathbf{v}) = H(m) - H(m|z). \tag{5}$$

Several simplifying assumptions can be made [34] that finally allow for approximating (5) using ray-casting from the sampled view pose $\mathbf{v}$ towards the modeled

transport box and computing the information gain $I_{\text{model}}(\mathbf{v})$ in closed form by considering the number of unknown cells along the ray.

For actively guiding object detection, we focus view planning on sensing regions of interest in addition to considering previously unseen volume. For this purpose, we integrate primitive and object detection results in the model. Referring to the applied coding scheme, regions that possibly contain an object but where no object has been detected yet are more interesting and have a higher interest score, respectively. In order to draw the robot's attention to interesting regions, we compute an interest score related measure $I_{\text{interest}}(\mathbf{v})$ as the sum of interest values over all cells visible from the sensor pose, and combine both measures for the final information gain $I(\mathbf{v})$:

$$I(\mathbf{v}) = \alpha \frac{I_{\text{model}}(\mathbf{v})}{I_{\text{model}}(\mathbf{v})^\star} + \beta \frac{I_{\text{interest}}(\mathbf{v})}{I_{\text{interest}}(\mathbf{v})^\star} \tag{6}$$

where $I_{\text{model}}(\mathbf{v})^\star$ and $I_{\text{interest}}(\mathbf{v})^\star$ are, respectively, the maximum model information gain and interest score over all samples. The parameters α and β can be used to weight the two measures (in our experiments $\alpha = \beta = 1$).

Implementation Details. For efficiency, we first extract all unknown cells (previously unseen volume) from the transport box, as well as all occupied cells (previously seen surface). If regions of modeled free space are considered interesting, they are handled extra and in addition to the extracted cells. All cells outside the sensor's view frustum (at the sampled view $\mathbf{v}$) are removed. For each of the the remaining cells, we conduct a reverse ray-casting from the cell to the view pose to determine the cell's visibility and update $I_{\text{model}}(\mathbf{v})$ and $I_{\text{interest}}(\mathbf{v})$ accordingly. To further speed up this step, we limit resolution and depth in the tree. Furthermore, we only consider those segments of the ray that are contained in the oriented bounding box of the transport box model.

Overall, our approach allows focusing the acquisition of new range scans in regions where we expect to find objects. For planning these views, we can compute and evaluate 100 samples per second.

5 Mobile Manipulation

For manipulating objects in our mobile bin picking scenario, we employ efficient grasp planning, motion planning for reaching objects on collision-free paths, and a global-to-local navigation strategy for moving between the transport box and the processing station.

Grasp Planning. We plan grasps in an efficient multistage process that successively prunes infeasible grasps using tests with increasing complexity. In the first stages, we find collision-free grasps on the object, irrespective of the pose of the object and not considering its scene context (see Fig. 8(a)). These poses can be pre-calculated efficiently in an offline planning phase. We sample grasp poses on

the shape primitives. From these poses, we extract grasps that are collision-free from pre-grasp pose to grasp pose according to fast collision check heuristics.

During online planning, we examine the remaining grasp poses in the actual poses of the objects to find grasps where a collision-free solution of the inverse kinematics in the current situation exists. We filter grasps before evaluation against our height map and finally search collision-free inverse kinematics solutions for the remaining ones. We allow collisions of the fingers with other parts in the transport box in the final stage of the grasp, i.e., in the direct vicinity of the object to grasp. The shape of the fingers allows for pushing them into narrow gaps between objects. Our grasp planning module finds feasible, collision-free grasps at the object. The grasps are ranked according to a score that incorporates efficiency and stability criteria.

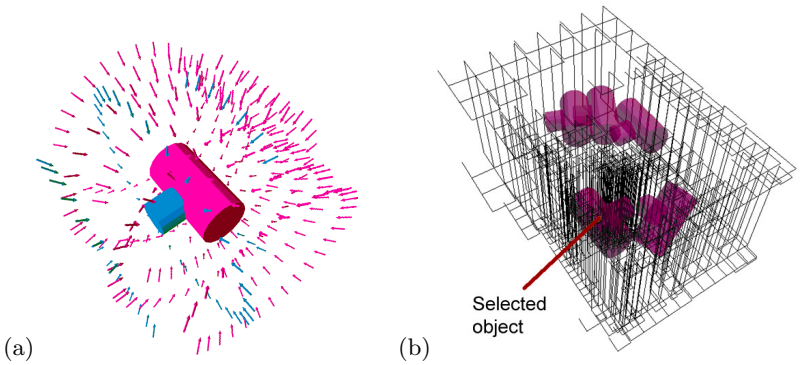
Motion Planning. We distinguish two types of motions that need to be planned: the motion for grasping the object and the motion for removing it from the transport box. For planning the grasping motion, we identify the best-ranked grasp that is reachable from the current posture of the robot arm. We solve this by successively planning reaching motions for the found grasps. We test the grasps in descending order of their score. For motion planning, we employ LBKPIECE [35]. In order to further increase performance, the grasping motion is again split into multiple segments. This allows for a quick evaluation if a valid reaching motion can be found by planning in descending order of probability that planning for a segment will fail.

After the execution of the reaching motion, we check if the grasp was successful. If the object is within the gripper, a removal motion is planned with the object model attached to the end-effector using the detected object pose. We allow minor collisions of the object and the end-effector with the collision map in a cylindrical volume above the grasp pose.

To reach the processing station, global navigation and local alignment are used in the same way as for approaching the box. Finally, the work piece is deposited at the processing station.

Multiresolution Height Map. We employ a multiresolution height map that extends our prior work on multiresolution path planning [36] for efficiently evaluating collision-free grasp postures and planning trajectories. The height map is represented by multiple grids that have different resolutions. Each grid has $M \times M$ cells containing the maximum height value observed in the covered area (Fig. 8(b)). Recursively, grids with quarter the cell area of their parent are embedded into each other, until the minimal cell size is reached. With this approach, we can cover the same area as a uniform $N \times N$ grid of the minimal cell size with only $\log_2((N/M) + 1)M^2$ cells.

Planning in the vicinity of the object needs a more exact environment representation than planning farther away from it. This is accomplished by centering the collision map at the object. This approach also leads to implicitly larger safety margins with increasing distance to the object.



(a) (b)

Fig. 8. Grasp planning: (a) We sample possible grasps and pre-grasp poses (visualized as arrows by color of primitive) for each shape primitive according to its parametric description, and discard those that are in collision within the object. (b) For grasp selection and motion planning we use a multiresolution height map.

Navigation. We use a global-to-local strategy for approaching the location where transport boxes are delivered to and the location where work pieces are processed. This removes the need of knowing the exact locations. Instead we assume an environment model in the form of a 2D map and a rough estimate of the locations. We navigate globally to the approximate locations, and accurately align the robot locally with the transport box and the processing station, respectively.

For global navigation, we employ state-of-the-art methods for localization and mapping in 2D representations of the environment. Adaptive Monte Carlo Localization [37] is used to estimate the robot's pose in a given occupancy grid map using a laser-range finder. We use A* search [38] to find the shortest obstacle-free path from the estimated pose in the map to the target location.

In order to maximize the workspace of the robot and allow for active perception, an accurate alignment between the robot and, respectively, the transport box and the processing station is necessary. We achieve this alignment by locally navigating to a predefined pose relative to the deduced box model (or model of the processing station) or to poses planned by the active perception component. This reactive alignment makes our approach robust against variations in the poses of transport boxes and processing station.

6 Experiments and Results

We tested the integrated system with our cognitive service robot Cosero [4]. For the detection experiments, we have chosen three types of objects (see Fig. 9). The objects are similar to parts found in real-world construction applications—a rectangular wooden plate containing two drilled holes (manually created CAD model), a cross clamping piece (CCP), a typical construction part to connect

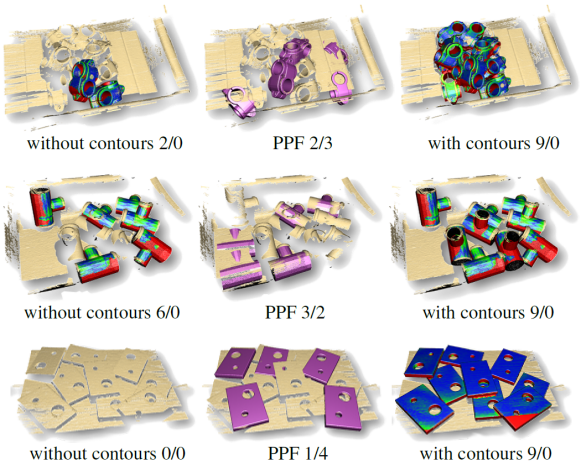
poles (CAD models are freely available), and a drain pipe connector (scanned from different perspectives and registered to obtain a 3D model).

Model Learning. For every object we have first computed a query graph using the offline variant of our approach. All reconstructions could be performed within 12 to 15 seconds. We compared the parameters of the detected primitives in the reconstruction to ground truth parameters and observed deviations of about 2% to 3%, using a precise 3D scanner. To judge the quality of our detected primitives for object detection, we compared the models learned from the scans with handcrafted models based on ground truth data. The observed recognition results were similar.

Object Detection and Pose Estimation. To assess the perception of our object detection approach, we acquired—for each object model—scans in five differently piled heaps of objects. We compared our shape and contour approach to our previous approach using shape primitives only [6, 7], and to an implementation of the state-of-the-art approach by Papazov and Burschka [31] based on point-pair features (PPF) with parameters as recommended by the authors. For qualitative analysis and visual inspection we show example scans and object detections for all approaches in Fig. 9(a). Quantitative results are summarized in the table in Fig. 9(b). As the PPF-based approach is randomized, it is run for ten times on every input scan to determine the average detection rate. For the class of objects in our problem setting our method clearly outperforms the other two methods.

Compared to the shape-only approach [6, 7], adding contour primitives improves detection results by resolving object pose ambiguities. In case of the CCP, we often find only a single cylinder that does not suffice for determining the object pose as it leaves two degrees of freedom undetermined. Adding circular contour primitives yields unique pose estimates and successful detections. The worst case for the shape-only approach is the scan of the wooden plates lying nearly flat on the table. The objects cannot be detected as the only found primitives are indefinitely extending planes with three open degrees of freedom. By combining the planes with the contour primitives, all objects with estimable pose can be detected.

Compared to the PPF-based approach (showing outstanding performance for objects of generic shape) we also achieve better detection rates. Man-made objects as addressed in our work are usually formed by compositions of few simple geometric primitives. These are easy to detect for our approach but also have the property of less varying local surface normals. This is disadvantageous for geometric features such as PPFs since feature descriptors computed in these areas do not provide unique transformations and can cause false positives. Our approach does not produce any false positives, because it requires only a minimal number of shape and contour matches. A restriction of our solution is that it is only suitable for objects that can be described by contour and shape primitives. It cannot be applied to arbitrary organic objects.



(a) Examples (true/false positives)

	Ours	Only 3D	PPF
CCP	**0.81**	0.23	0.29
Pipe	**0.89**	0.47	0.27
Wood	**0.82**	0.21	0.08
Overall	**0.84**	0.30	0.22

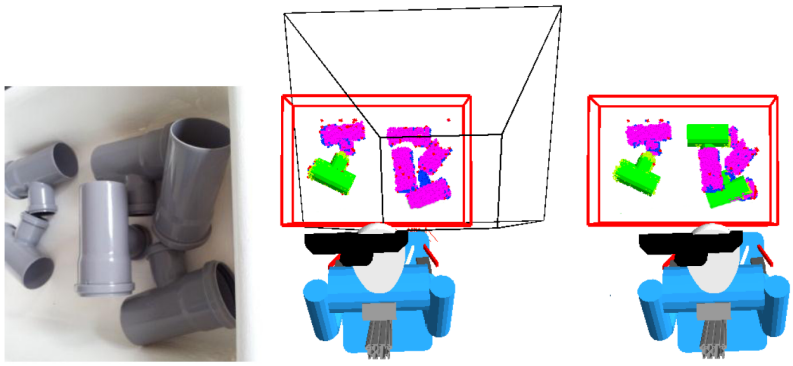
$$\text{acc.} = \frac{\text{true positives}}{\text{t. pos.} + \text{f. neg.} + \text{f. pos.}}$$

(b) Detection accuracy

Fig. 9. Results for object detection: (a) typical detections for the different objects used in our experiments, (b) overall average detection accuracies of the evaluated methods.

Active Perception. For a proof-of-concept and a qualitative analysis of the active component of our perception pipeline, we conducted a series of roughly 50 experiments where the robot had to explore different arrangements of pipe connectors in transport boxes (ranging from one to ten objects per box). After successful verification object hypotheses were accumulated and tracked using a multi-hypotheses tracker. Regions where no object was detected, but which could have contained objects drew the robot's attention when planning the next view. In all experiments, active perception resulted in a fully explored transport box and at least one graspable object. Fig. 10 shows typical examples where the active perception component leads to new object detections and a complete model of the transport box. In cases of severe occlusions, the robot has been able to find occluded objects after acquiring a second or third view. No more than three views were needed to find an object. In most cases an object was already detected in the first view.

Mobile Bin Picking. For assessing the performance of the overall system, we have recorded 32 runs, in which the robot picks up one pipe connector object from the transport box (filled with 10 pipe connectors) and delivers it to the processing station. In 28 runs, the robot could successfully grasp and deliver the object. In nine of these successful runs, the robot first failed to grasp an object, detected its failure, and performed another grasp. This was the case, for instance, when the object slipped out of the gripper after grasping. In four runs, the object was not successfully delivered to the processing station. In three out of the four failed runs the last object could not be detected. In one run, the object slipped out of the gripper after lifting. This is caused by the collisions between



(a) Scene (b) Detections + NBV (c) Second view

Fig. 10. Active perception: occlusions in the scene (a) hinder the robot in finding all objects. Considering the primitive (magenta) and object (green) detections when planning the next best view (b), allows for finding more objects in the second view (c).

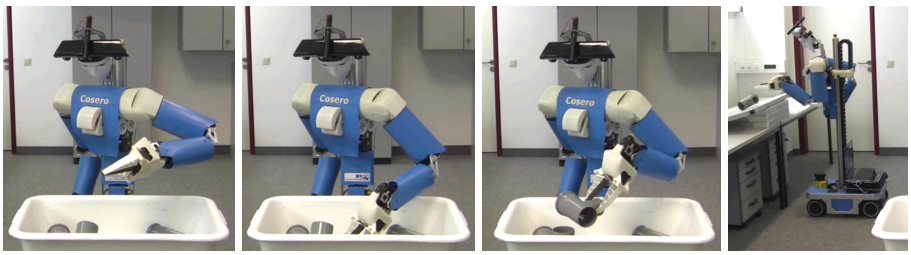


Fig. 11. Example of a mobile bin picking and delivery run. From left to right: the robot grasps an object out of the transport box and puts it on the processing station.

the gripper and other objects that we have to allow during the grasp. These minor collisions can cause changes in the object's pose that can make the chosen grasp impossible or unstable. Some images from one of the runs are shown in Fig. 11. Table 1 shows the mean and standard deviation of the measured phase durations for the 32 individual runs. Please note, that the timings for the grasp selection and motion planning within the cognition phase are averaged over the ten runs it took to clear one completely filled box. One can see that the longest phase is the cognition phase where objects are detected and the grasping motion is planned. This phase also includes the transmission of the sensor data to the object recognition module on a physically distinct computer on the robot.

7 Conclusion

In this work, we presented an integrated system for a mobile bin picking application. It combines manipulation and navigation skills, including: 3D object

Table 1. Time needed for phases of the mobile bin picking demonstration.

Phase	**Duration** (in sec.)	
	Mean	Std. dev.
Navigation (transport box)	20	8
Approaching (transport box)	16	11
Cognition phase	83	41
- Grasp selection	19.9	14.4
- Motion planning	3.8	2.6
Grasping	36	7
Navigation (processing station)	26	9
Approaching (processing station)	22	9
Putting the object on the processing station	18	2

detection and pose estimation, view planning for active perception, grasp and motion planning as well as global navigation and local precise alignment. We recognize objects using an efficient noise-resistant approach based on RANSAC and subgraph matching. In order to obtain the necessary shape and contour composition graphs, we derive models automatically from 3D point clouds or CAD files. To cope with imperfect measurements and occlusions in the unordered pile of objects in the transport box, we developed view planning techniques. Grasping objects is realized as a multistage process from coarse, i.e., global navigation in the environment, to fine, i.e., planning a collision-free end-effector trajectory within a multiresolution collision map. Intermediate steps align our robot to the transport box and the processing station using local sensing and navigation, and evaluate the graspability of objects using fast heuristics.

We showed the applicability of our approaches in a mobile bin picking and part delivery task in our lab. A video summarizing our work is available on our website[1]. Among other skills, we demonstrated mobile bin picking in the @Home final of RoboCup 2012 in Mexico, where our robots convinced the high-profile jury and won the competition.

Acknowledgments

This work was supported by the EU Project EC FP7-ICT-231143 ECHORD.

References

1. K. Ikeuchi, B. K.P. Horn, S. Nagata, T. Callahan, and O. Feirigold. Picking up an object from a pile of objects. In *Robotics Research: The First International Symposium*, pages 139–162. MIT Press, 1984.
2. K. Rahardja and A. Kosaka. Vision-based bin-picking: Recognition and localization of multiple complex objects using simple visual cues. In *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, 1996.

[1] www.ais.uni-bonn.de/ActReMa

3. M.-Y. Liu, O. Tuzel, A. Veeraraghavan, Y. Taguchi, T. K Marks, and R. Chellappa. Fast object localization and pose estimation in heavy clutter for robotic bin picking. *Int. J. of Robotics Research*, 31(8):951–973, 2012.
4. J. Stückler, D. Holz, and S. Behnke. RoboCup@Home: Demonstrating everyday manipulation skills in RoboCup@Home. *IEEE Robotics & Automation Magazine*, 19(2):34–42, 2012.
5. P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 61(1):55–79, January 2005.
6. M. Nieuwenhuisen, J. Stückler, A. Berner, R. Klein, and S. Behnke. Shape-primitive based object recognition and grasping. In *Proc. 7th German Conference on Robotics*, 2012.
7. M. Nieuwenhuisen, D. Droeschel, D. Holz, J. Stückler, A. Berner, J. Li, R. Klein, and S. Behnke. Mobile bin picking with an anthropomorphic service robot. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 2319–2326, 2013.
8. C. Papazov, S. Haddadin, S. Parusel, K. Krieger, and D. Burschka. Rigid 3D geometry matching for grasping of known objects in cluttered scenes. *Int. J. of Robotics Research*, 31(4):538–553, 2012.
9. M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.
10. F. Bley, V. Schmirgel, and K.-F. Kraiss. Mobile manipulation based on generic object knowledge. In *Proc. IEEE Int. Symp. on Robot and Human Interactive Communication*, 2006.
11. C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu, and S. Ramalingam. Voting-based pose estimation for robotic assembly using a 3D sensor. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2012.
12. E. Wahl, U. Hillenbrand, and G. Hirzinger. Surflet-pair-relation histograms: a statistical 3D-shape representation for rapid classification. In *Proc. Int. Conf. on 3-D Digital Imaging and Modeling*, 2003.
13. B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3D object recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2010.
14. E. Kim and G. Medioni. 3D object recognition in range images using visibility context. In *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, 2011.
15. R. Schnabel, R. Wessel, R. Wahl, and R. Klein. Shape recognition in 3D point-clouds. In *Proc. Int. Conf. on Computer Graphics, Visualization and Computer Vision*, 2008.
16. Y. Li, X. Wu, Y. Chrysathou, A. Sharf, D. Cohen-Or, and N. J. Mitra. Globfit: Consistently fitting primitives by discovering global relations. *ACM Trans. on Graphics*, 30:52:1–52:12, 2011.
17. B.J. Cohen, G. Subramanian, S. Chitta, and M. Likhachev. Planning for manipulation with adaptive motion primitives. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2011.
18. S. Chitta, E. G. Jones, M. Ciocarlie, and K. Hsiao. Perception, planning, and execution for mobile manipulation in unstructured environments. *IEEE Robotics & Automation Magazine*, 19(2):58–71, 2012.
19. E. Klingbeil, D. Rao, B. Carpenter, V. Ganapathi, A. Y. Ng, and O. Khatib. Grasping with application to an autonomous checkout robot. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2011.
20. L. Chang, J. R. Smith, and D. Fox. Interactive singulation of objects from a pile. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2012.

21. M. Gupta and G. S. Sukhatme. Using manipulation primitives for brick sorting in clutter. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2012.
22. S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. Van de Weghe. HERB: a home exploring robotic butler. *Autonomous Robots*, 28(1):5–20, 2010.
23. B. Bäuml, F. Schmidt, T. Wimböck, O. Birbach, A. Dietrich, M. Fuchs, W. Friedl, U. Frese, C. Borst, M. Grebenstein, O. Eiberger, and G. Hirzinger. Catching flying balls and preparing coffee: Humanoid Rollin'Justin performs dynamic and sensitive tasks. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2011.
24. N. Vahrenkamp, T. Asfour, and R. Dillmann. Simultaneous grasp and motion planning: Humanoid robot ARMAR-III. *IEEE Robotics & Automation Magazine*, 19(2):43–57, 2012.
25. A. Jain and C. C. Kemp. EL-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. *Autonomous Robots*, 28(1):45–64, 2010.
26. M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mösenlechner, D. Pangercic, T. Rühr, and M. Tenorth. Robotic roommates making pancakes. In *Proc Int. Conf. on Humanoid Robots*, 2011.
27. N. J. Mitra, N. Gelfand, H. Pottmann, and L. Guibas. Registration of point cloud data from a geometric optimization perspective. In *Symp. Geometry Processing*, 2004.
28. G. H. Bendels, R. Schnabel, and R. Klein. Detecting holes in point set surfaces. *Journal of WSCG*, 14(1–3), February 2006.
29. R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226, 2007.
30. A. Berner, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. A graph-based approach to symmetry detection. In *Proc. IEEE/EG Int. Symp. on Volume and Point-Based Graphics*, 2008.
31. C. Papazov and D. Burschka. An efficient RANSAC for 3D object recognition in noisy and occluded scenes. In *Proc. Asian Conf. on Computer Vision*, 2011.
32. M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, 2006.
33. D. Holz and S. Behnke. Sancta simplicitas – on the efficiency and achievable results of SLAM using ICP-based incremental registration. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1380–1387, 2010.
34. A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 2013.
35. I. A. Sucan and L E. Kavraki. Kinodynamic motion planning by interior-exterior cell exploration. In *Algorithmic Foundation of Robotics VIII*, volume 57, pages 449–464. Springer, 2009.
36. S. Behnke. Local multiresolution path planning. *RoboCup 2003: Robot Soccer World Cup VII*, pages 332–343, 2004.
37. D. Fox. Adapting the sample size in particle filters through KLD-sampling. *I. J. Robotic Res.*, 22(12):985–1004, 2003.
38. P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. on Systems Science and Cybernetics*, 4(2):100–107, 1968.