Accepted for Robotics and Autonomous Systems, to appear 2012.

Efficient 3D Object Perception and Grasp Planning for Mobile Manipulation in Domestic Environments

Jörg Stückler, Ricarda Steffens, Dirk Holz, and Sven Behnke

Autonomous Intelligent Systems Group, Department of Computer Science VI, University of Bonn, Friedrich-Ebert-Allee 144, 53113 Bonn, Germany

Abstract

In this article, we describe efficient methods for tackling everyday mobile manipulation tasks that require object pick-up. In order to achieve real-time performance in complex environments, we focus our approach on fast yet robust solutions. For 3D perception of objects on planar surfaces, we develop scene segmentation methods that process depth images in real-time at high frame rates. We efficiently plan feasible, collision-free grasps for the segmented objects directly from the perceived point clouds to achieve fast execution times. We evaluate our approaches quantitatively in lab experiments and also report on the successful integration of our methods in public demonstrations at RoboCup@Home competitions in 2011 and 2012.

Keywords: Scene Segmentation, Grasp Planning, Mobile Manipulation

1. Introduction

Mobile manipulation tasks in domestic environments require a vast set of perception and action capabilities. A service robot not only requires localization, mapping, path planning, and obstacle avoidance abilities to safely navigate through the environment. It also needs to integrate object detection, recognition, and manipulation. In addition, a service robot is not just to achieve a task, but to perform it in reasonable time. While much research has been invested into the general solution of complex perception and motion planning problems, little work has been focused on methods that solve such tasks efficiently in order to allow for continuous task execution without interruptions.

In this article, we present fast methods to flexibly grasp objects from planar surfaces. To achieve fast performance, we integrate real-time object perception with efficient grasp planning and motion control. For real-time perception, we combine rapid normal estimation using integral images with efficient segmentation techniques. We segment the scene into the support plane of interest and the objects thereon. Our perception algorithm processes depth images in real-time at a frame rate of approx. 20 Hz. From the

Email address: stueckler@ais.uni-bonn.de, steffens@cs.uni-bonn.de, holz@ais.uni-bonn.de, behnke@cs.uni-bonn.de (Jörg Stückler, Ricarda Steffens, Dirk Holz, and Sven Behnke)

raw object point clouds, our grasp planning method derives feasible, collision-free grasps within about 100 ms. We consider grasps on objects either from the side or from above. The planned grasps are then executed using motion primitives. We integrate our approaches into a system that we evaluate for robustness and efficiency in lab experiments. Finally, we report on the public demonstration of our approaches at RoboCup@Home competitions in 2011 and 2012.

2. Related Work

Many research groups currently develop systems for mobile manipulation in everyday environments. A very prominent example is the Personal Robot 2 (PR2), developed by Willow Garage [3]. It is equipped with two 7 DoF compliant arms and a parallel gripper with touch sensor matrices on the gripper tips. Leeper et al. [11] use the system in a teleoperated setting. Besides directly controlling the robot's end effector, the user can follow different strategies for grasping objects. In one of the strategies, the user selects a grasp from a set of feasible poses suggested by a planner [8]. Beetz et al. [1] let a PR2 make pancakes together with a second robot. This task involves fetch and delivery actions for a variety of objects which are perceived either based on the raw 3D measurements, object-specific visual appearance models, or 3D CAD models.

Further systems that perform object manipulation in cluttered environments have been reported by Srinivasa et al. [22, 23]. In [23], the authors present a robotic busboy system in which a mobile tray delivers mugs to a statically mounted manipulator. The mobile tray navigates through visual ceiling markers to a predefined position. The manipulator grasps the mugs and loads them into a dishwasher rack. A real-time vision system that is designed for the mugs estimates the pose of the mugs on the tray. Since the objects are known, valid grasps on the mug are precomputed. The grasp planner then selects online a best feasible grasp from several criteria like reachability and collision avoidance. The authors report a total average duration of 51 sec for executing a grasp and releasing the mug in the dishrack. With the robot HERB [22], the system has been extended to more general object recognition and motion planning. While object recognition is aborted after 1 sec, the planning of motions is reported to take several seconds. Our approach is not restricted to recognizable objects.

Okada et al. [15] demonstrate dishwashing and water-pouring with a humanoid HRP2 robot. They adapt pre-trained motions to actual scene context and verify the behavior using sensory information. The perception methods are designed for the specific applications. Jain and Kemp develop EL-E [9], a mobile manipulator that shall assist the impaired. EL-E consists of a Katana manipulator on a vertical linear actuator mounted on a differential drive. The user can draw the robot's attention to objects on tables and the floor by pointing on the objects with a laser pointer. The robot then picks the object up and delivers it to the user. While we extract object information in real-time from a depth image sensor, they segment measurements of a 3D laser using connected components labelling to find object clusters above table height. Similar to our approach, they perform top grasps along the object's principal axis. However, side grasps are not considered. If an object is too high or too wide to fit into the gripper, they also consider overhead grasps on top-most points of the object. To ensure that the grasping motion is not in collision, a cuboid volume from the manipulator base to the object is checked for obstacles.

Morales et al. [13] propose a system that selects feasible, collision-free grasps on objects from a database. They determine the set of feasible grasps for the object from its CAD model in an offline phase. After the object has been recognized and localized with a stereo vision system, a grasp simulation framework (GraspIt! [12]) is used to select a collision-free grasp among the potential grasps for the object. The authors report 5 ms computation time for the recognition of objects in a database of five objects. The time for planning of collision-free, feasible grasps in GraspIt is reported to range from seconds to several minutes [12].

2.1. Object Perception

In typical household environments, objects are usually constrained to well-defined locations like, for instance, tabletops, shelves and other horizontal support planes. This natural restriction of space is exploited in the majority of approaches to object perception and search. A common processing scheme [17, 7] and perception pipeline for detecting and recognizing objects in depth images and 3D point clouds is to

1. detect the horizontal support planes,
2. extract and cluster the measurements on top of these planes, and
3. perform further processing, e.g., recognizing, classifying or tracking of the found clusters.

Differing in related works are, most notably, the used methods for the individual processing steps that determine – amongst others – the robustness, speed, and runtime requirements of the overall system.

Rusu et al. [17] propose to segment point clouds into objects on planar surfaces. They suggest to use RANSAC to detect planes and to extract shape primitives on the objects. Remaining points are described by meshes. Schnabel et al. [21] decompose noisy point cloud data into geometric shape primitives with an efficient multi-resolution approach. In previous work [7], we combined planar pre-segmentation as in [17] with efficient object modelling using shape primitives [21], and made the approaches applicable to the measurements of time-of-flight (ToF) cameras. We presented techniques to cope with the specific error sources of the cameras, to speed up processing by exploiting the image-like data organization, and for detecting geometric primitives in the found object clusters.

In [18], shape primitives are extracted and are used as obstacles for a motion planner. Şucan et al. [27] extend this approach to identify areas of a scan that are occluded by the robot. They maintain these areas from a sequence of scans while the robot is moving. In this way, the robot can still avoid obstacles occluded by itself.

2.2. Grasp Planning

Grasp planning approaches can be divided into empirical methods and approaches that analyse the mechanical stability of grasps. A recent survey on grasp planning approaches can be found in [19].

In the latter category, Borst et al. [2] proposed the grasp wrench space to measure the stability of a grasp for articulated hands. This approach has been incorporated into the GraspIt! framework [12] which plans grasps on objects composed of shape primitives in

simulated scenes. Pelossof et al. [16] fit superquadrics to objects and train support vector machines to predict the quality of grasps on objects. They train on samples generated with the GraspIt! simulator.

These approaches, however, require the knowledge and perception of complete 3D object models. To circumvent this, several methods have been developed that operate directly on measurements from 3D sensors [8] or color images [20]. Saxena et al. [20] propose a learning approach that retrieves grasping points from the observation of the object in 2D color images. Similar to the work in [16], they train on synthetic data obtained with a grasp simulator.

Similar to our approach, Hsiao et al. [8] derive feasible, collision-free grasps from the raw object point cloud. They select the best-ranked grasp and plan a collision-free motion for the arm, taking into account obstacles that are perceived by the robot's 3D sensors. While the authors demonstrate that the approach can robustly grasp a variety of objects in a wide range of configurations, the execution speed of the system for perception and grasping is still far slower than human performance.

3. System Overview

3.1. Design of Cognitive Service Robot Cosero

Domestic environments are designed for the specific capabilities of the human body. It is therefore natural to endow the robot with an anthropomorphic upper body scheme for similar manipulation abilities. The two anthropomorphic arms of our robot Cosero resemble average human body proportions and reaching capabilities (see Fig. 1). A yaw joint in the torso enlarges the workspace of the arms. In order to compensate for the missing torso pitch joint and legs, a linear actuator in the trunk can move the upper body vertically by approx. 0.9 m. This allows the robot to manipulate on similar heights like humans — also on the ground. To maneuver in the narrow passages found in household environments, we equipped Cosero with an omnidirectional drive.

Compared to its predecessor Dynamaid [24], we increased payload and precision of the robot by stronger actuation. We also improved Cosero's gripper design in 2012. We actuate two Festo FinRay fingers on rotary joints (see Fig. 2). These fingers are made from lightweight plastics material. When the gripper is closed on an object, the bionic fin ray structure of the fingers adapts its shape to the object surface. By this, the contact surface between fingers and object is extended significantly, compared to a rigid mechanical structure. We attached anti-skidding material onto the finger surface in order to improve their grip.

Cosero perceives its environment with a variety of complementary sensors. The robot senses the environment in 3D with a Microsoft Kinect RGB-D camera in its head that is attached to the torso with a pan-tilt unit in the neck. We also attached infrared distance sensors to the palm in each gripper to measure the distance to objects directly from the grippers.

3.2. Mobile Manipulation in Everyday Environments

We develop Cosero to perform a variety of mobile manipulation tasks in everyday environments. For mobile manipulation, we combine safe navigation of the robot through the environment with motion control methods for the upper body.

Figure 1: Cognitive service robot *Cosero* moves a chair and waters a plant at RoboCup German Open 2012.

3.2.1. Motion Control

We implemented omnidirectional driving for Cosero's eight-wheeled mobile base [24]. The linear and angular velocity of the drive can be set independently and can be changed continuously. We determine the steering direction and the individual wheel velocities of the four differential drives, which are located at the corners of the rectangular base, from an analytical solution to the drive's inverse kinematics.

For the anthropomorphic arms, we implemented differential inverse kinematics with redundancy resolution [24]. We also developed compliance control for the arms [26]. For our method, we exploit that the servo actuators are back-drivable and that the torque which the servo applies for position-control can be limited. Compliance can be set for each direction in task- or joint-space separately. For example, the end-effector can be kept loose in both lateral directions while it keeps the other directions at their targets. With these methods, Cosero can perform a variety of parameterizable motions like opening doors, handing objects over, and carrying large objects.

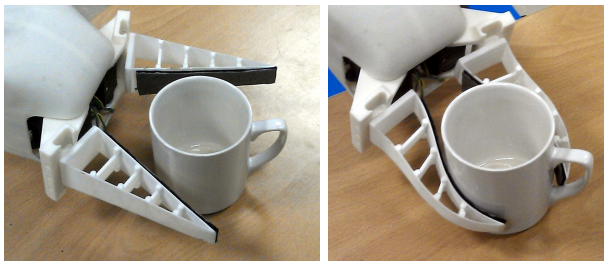


Figure 2: Design of Cosero's FinRay grippers.

3.2.2. Mobile Manipulation

We propose a coarse-to-fine strategy for aligning the robot to the objects involved in mobile manipulation. For example, when the robot grasps an object from a table, it first approaches the table roughly within the reference frame of a static map. Then, it adjusts in height and distance to the table. Finally, it aligns itself to bring the object into the workspace of its arms.

Cosero grasps objects on horizontal surfaces like tables and shelves in a height range from ca. 0.3 m to 1 m [24]. It also carries the object, and hands it to human users. We also developed solutions to pour-out containers, to place objects on horizontal surfaces, to dispose objects in containers, to grasp objects from the floor, and to receive objects from users using parametrized motion primitives. Semantic knowledge about the location of objects and persons is either specified in advance or perceived while the robot works on a task. For instance, in order to deliver objects to specific persons, the robot searches for the persons and recognizes them by their face. To find task-specific locations for objects such as shelves and tables, possible approach poses can be annotated in a map.

4. Real-time 3D Perception

Our approach to object detection is focused on processing images of depth cameras such as the Microsoft Kinect at high frame rates. At a resolution of 160×120, we can process depth images with up to 20 Hz. This enables our system to extract information about the objects in a scene with a very low latency for further decision-making and planning stages.

4.1. Overview on the Processing Pipeline

1. *Compute normals:* For all points, we compute local surface normals. For efficiency, we exploit the organized data structure of the underlying data. We approximate the local point neighborhoods using neighboring image pixels and the normals by using the approximate tangents to the surrounding surface (see Sec. 4.2).

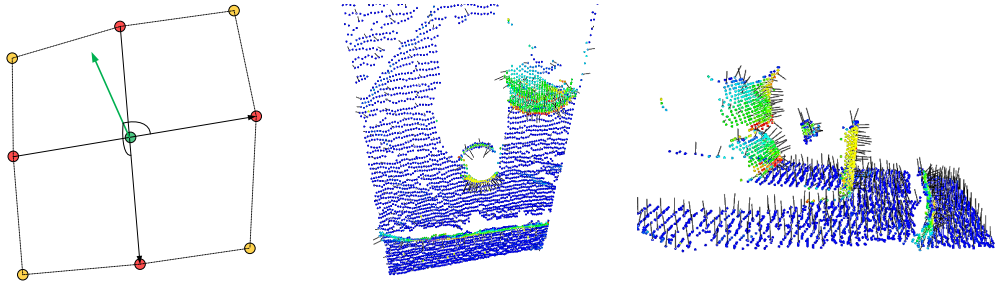
2. *Extract horizontal points*: We extract all points with vertical normals (normals pointing upwards along the z-axis). It can be assumed that the resulting points are all lying on horizontal surfaces. The pre-computed surface normals do not only allow for focusing subsequent processing steps on points on horizontal surfaces, but also for efficiently segmenting the complete depth image into planes [6].

3. *Detect support plane:* We apply RANSAC [4] to find the most dominant horizontal plane and the points supporting it. By only considering points on horizontal surfaces, we save computations and find the support plane with considerably less iterations. We further decrease the number of processed points, by limiting the search space in both height and distance to the robot. Due to the manipulation constraints of our robot, we neglect points (and support planes) higher than 1 m. In addition, we do not consider points farther away from the robot than 3 m.

4. *Detect object candidates:* For all points that do not belong to the most dominant horizontal support plane, we extract those that are above the plane and whose projections lie within the plane's convex hull. The resulting points are likely to

(a) Example table scene (b) Colored point cloud (c) Detected objects

Figure 3: (a) Example tabletop setting. (b) Raw point cloud from the Kinect with RGB information. (c) Each detected object is marked with a random color.



(a) Basic principle (b) Example (left: top view, right: side view)

Figure 4: Principle of fast normal computation using integral images. (a) Two vectors tangential to the surface at the desired position are computed using the red points. The local surface normal is computed by applying the cross product to them. (b) Typical result of an acquired point cloud with surface normals.

have been measured on the surface of an object on top of the support plane. We then apply a simple Euclidean clustering to obtain individual sets of points and object candidates, respectively. In case of grasping objects in shelves, we slightly shrink the convex hull in order to neglect points at a side or back wall of the shelf.

A typical segmentation result of the processing described so far, is shown in Fig. 3.

5. *Track objects:* Our fast segmentation approach gives us the possibility of detecting objects in depth images at high frame rates. In particular, it allows for tracking detected object clusters over several frames in order to obtain good estimates of their position on the support plane. We also compute the principal axes and oriented bounding boxes for all object candidates for subsequent processing steps.

4.2. Fast Computation of Local Surface Normals

A common way for determining the normal to a point on a surface is to approximate the problem by fitting a plane to the point's local neighborhood. Less accurate, but considerably faster is to consider pixel neighborhoods instead of spatial neighborhoods [7]. That is, the organized structure of the point cloud as acquired by time-of-flight or RGB-D

cameras is used instead of searching through the 3D space spanned by the points in the cloud. By using a fixed pixel neighborhood and, in addition, neglecting pre-computed neighbors outside of some maximum range as in [7], one can avoid the computationally expensive neighbor search, but still needs to compute and analyze the local covariance matrix. Here, we use an approach that directly computes the normal vector over the neighboring pixels in x and y image space.

The basic idea of the normal estimation method [6] is to determine local surface normals from the cross product of two tangents to the surface. For each pixel in the depth image, the tangents are estimated from local pixel neighbors. In the simplest case, both tangents could be calculated from just the horizontal and vertical neighbors, respectively. However, this approach would be highly prone to measurement noise. The tangent estimates should therefore be averaged in an image neighborhood. By using integral images, such averaging operations can be computed in constant time independent of the neighborhood size.

We first create two maps of tangential vectors, one for the rows and one for the columns in the depth image. For each map, we compute the difference vectors between the corresponding 3D points. That is, we have a total of 2×3 channels holding the Cartesian x, y, and z coordinates for the difference vectors. For each of the channels, we then compute an integral image, which leads to a total number of six integral images. By using integral images, we can compute the average tangential vectors with only $2 \times 4 \times 3$ memory accesses, independent of the size of the smoothing area. The overall runtime complexity of this approach is linear in the number of points for which normals are computed.

4.3. Tracking Detected Objects

We make the perceptions of 3D object segments in the individual frames persistent in a multi-hypotheses object tracker. For each hypothesis, we estimate 3D position and velocity in the reference frame of the mobile base through Kalman Filters (KFs). In the KF prediction step, we use odometry information to compensate for the motion of the robot. The tracks are corrected with the observations of their 3D position and extents. We use the Hungarian method [10] to associate the detections in an image uniquely with existing hypotheses.

5. Efficient Grasp Planning

Objects in everyday manipulation scenarios are highly variable in shape and appearance. Furthermore, the configuration of objects and obstacles in a scene is strongly unstructured. It is therefore challenging to develop a grasp planning method that can cope with any encountered situation. Our approach is specifically suited for rigid objects whose shape is symmetric along the vertical axes of the object, or for objects that provide ridge-like shapes in horizontal directions. We also assume that the center of gravity roughly coincides with the center of the object. While many objects meet these assumptions, our approach can also yield robust grasps for objects that violate the constraints.

We developed flexible grasping motions to grasp objects from the side or from above. When the robot encounters a new situation, it plans and executes a feasible collision-free grasp for the object of interest. The robot perceives the scene with its depth camera.

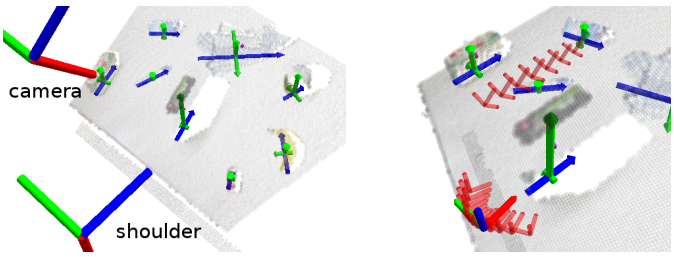


Figure 5: Left: We extract object pose and shape properties from the object points. The arrows mark the bounding box of the objects by the principal axes. Right: We rank feasible, collision-free grasps (red, size prop. to scores) and select the most appropriate one (larger RGB-coded coordinate frame).

It interprets the raw point representation of the objects on the grasp surface which is provided by our real-time 3D perception method (see Sec. 4).

5.1. Grasp Motion Primitives

We distinguish two kinds of grasps for which we apply parametrizable motion primitives. Side-grasps are designed to approach the object along its vertical axis by keeping the parallel grippers aligned horizontally. To grasp objects from the top, we pitch the end-effector by 45° downwards to grasp objects with the finger tips.

Both kinds of grasps are flexible in the orientation around the vertical upward direction. However, we limit the yaw orientation to a range between 0° and 90° (for the right arm) due to kinematic constraints of the robot arm and torso. Orientations beyond this range are grasped in the closest limit angle. Alternatively, the robot can simply choose its left arm to grasp within the reachable range.

The motion primitives approach the pre-grasp poses on a direct line with open gripper. We establish the yaw orientation at the pre-grasp pose by smooth interpolation along the reaching trajectory. Once the pre-grasp pose is reached, the side-grasp motion primitive simply approaches the object and closes the gripper. For the top-grasp motion, we do not establish the pitch orientation of the pre-grasp pose until the pre-grasp position has been reached. We assume that the pre-grasp positions are placed at a fixed distance behind the grasp position along the grasp direction (0.1 m in our case). We use the IR distance sensors in the gripper to determine premature contact with the object or the support plane. In such a case, the approach of the object is stopped. After the object has been grasped, the end-effector moves back to its initial pose.

5.2. Planning of Collision-Free Grasps

The grasp planner selects a feasible collision-free grasp for the object of interest. It samples grasp candidates, removes infeasible and colliding grasps, and ranks the remaining grasps to find the most promising one.

The planner outputs a pre-grasp pose to parametrize the grasping motion. A grasp pose directly corresponds to the pose of the end-effector which we define as follows: We place the grasp at the center of the gripper. The x-axis and y-axis of the grasp pose align with the direction from wrist to finger tips and the direction from the right to the left finger, respectively.

5.2.1. Sampling of Grasp Candidates

We sample grasp candidates depending on pose and shape properties of the object. In order to determine these properties, we project the raw points of the object into the horizontal plane and measure the principal axes of the point distribution. In addition, we determine height, center, and bounding box (aligned with the principal axes) of the object.

Once the shape and pose of the object are known, we determine feasible grasps for the object. For the side-grasps, we sample pre-grasp poses on an ellipse in the horizontal plane in equally-sized angular intervals. The center and axes of the ellipse directly correspond to the properties of the object's bounding box. The diameters of the ellipse add the distance towards the grasp point to the diameters of the bounding box (0.1 m in our implementation). We grasp the objects as low as possible above the surface at a specific height. This makes the grasping more robust for measurement and control inaccuracies. Higher grasps could easily topple the object over, when the robot touches it while moving in grasping direction. We set the grasp height to half the height of the gripper plus a safety padding of 0.03 m.

We sample the top-grasps on all points in a specific height range (set to 2 cm) below the highest point of the object. In order to merge similar grasps, we downsample the points in a voxel grid (at 1 cm resolution). We then determine at each point in the remaining set the local 3D neighborhood in the object segment using efficient kd-tree queries. We set the radius of this query to the maximum graspable width. From the distribution of these local neighbors, we determine the principal direction in the horizontal plane. We align the gripper in x-direction along the principal axis. In this way, we can plan grasps on object shapes with ridges such as bowls, or curved objects. To account for kinematic constraints of Cosero's anthropomorphic arms, we constrain the pitch of the grasp to 45° in downward direction. We place the pre-grasp pose 0.1 m above the height of the considered sample point, but at least 0.1 m above the support plane.

5.2.2. Filtering for Feasible and Collision-Free Grasps

Since the sampling stage does not consider any feasibility constraints or collisions, we filter the grasp candidates in a post-processing step. We take the following criteria into account:

- *Grasp width.* We reject grasps if the object's width orthogonal to the grasp direction does not fit into the gripper.
- *Object height.* Side-grasps are rejected if the object is too low.
- *Reachability.* We do not consider grasps that are outside of the arm's workspace.
- *Collisions.* We check for collisions during the reaching and grasping motion.

Fig. 5 shows an example for grasps that satisfy our criteria.

One possible solution for collision checking would be to search for collisions of all robot limbs during the complete trajectory of the grasping motion. However, we propose to use simple geometric constraints to find all possible collisions (see Fig. 6). While our method is more conservative, we can find collisions with only little computational effort.

We first project all points on obstacles into the horizontal plane. In order to avoid collisions of the upper arm, we search for collisions within a circle around the shoulder

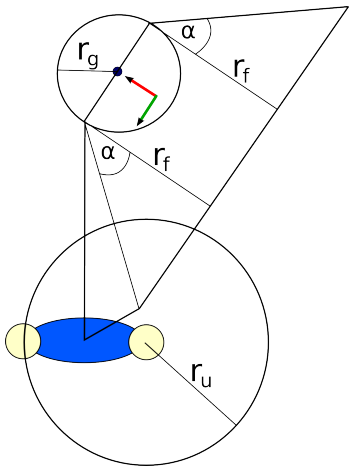


Figure 6: For each sampled grasp (final position: black dot, pre-grasp at frame origin, x-direction: red arrow, y-direction: green arrow), we check for collisions that may occur during the execution of the grasp motion primitive. All points on obstacles are projected into the horizontal plane. We require the region around the shoulder (right yellow circle) within upperarm length distance r_u to contain no obstacles. We further require that the gripper and the forearm can move towards the object by checking a cone with opening angle α and forearm length r_f behind the grasping pose. We extend the cone towards the robot's center position to cover the area swept during the reaching motion. At the final grasp position (black circle), the gripper is not in collision if there is no obstacle within a distance of r_g.

with a radius r_u equal to the upperarm length. We further require that the gripper and the forearm can move towards the object by checking a cone with opening angle α and forearm length r_f behind the grasping pose. We extend the cone towards the robot's center position to cover the area swept during the reaching motion. Finally, we search for collisions within a small circle at the final grasp position. The radius r_g of this circle is set to the maximum diameter of the open gripper.

5.2.3. Ranking of Grasps

We rank the feasible and collision-free grasps for several criteria such as

- *Distance to object center.* We favor grasps with a smaller distance to the object center.
- *Grasp width.* We reward grasp widths closer to a preferred width (0.08 m).
- *Grasp orientation.* Preference is given to grasps with a smaller angle between the line towards the shoulder and the grasping direction.
- *Distance from robot.* We support grasps with a smaller distance to the shoulder.

Fig. 5 illustrates this process with example rankings.

From the ranked grasps, we find the best top- and side-grasps and select the most appropriate one. This decision depends on the relation of the object height to the largest extent of the object in the horizontal plane. We implement a small bias towards the faster side grasps by scaling up the score of side grasps with a constant factor.

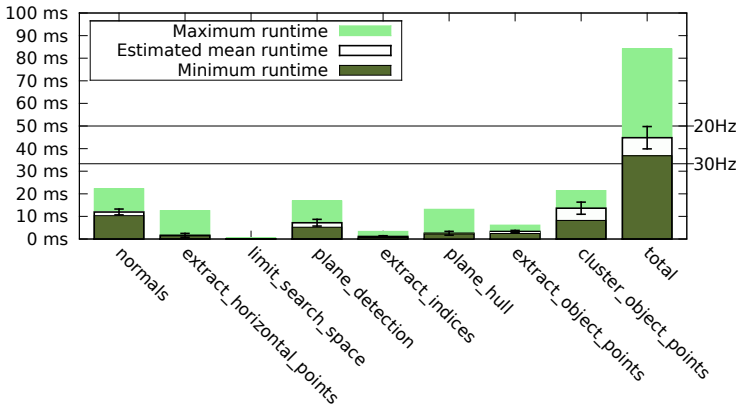


Figure 7: Measured runtime in a tabletop scene for the individual processing steps in the proposed perception pipeline. For the scene, 10.000 range images (160×120) have been acquired and segmented. Both minimum and average total runtime are slighly below the optimal 30 Hz.

6. Experiments

6.1. Quantitative Results

6.1.1. Runtime Efficiency

We summarize average runtimes of several stages of our perception and grasp planning pipeline in Fig. 7 for a tabletop scene. For a depth image resolution of 160×120, our segmentation approach achieves an average frame rate of approx. 20 Hz. The experiments have been carried out on a Lenovo X200s notebook with an Intel Core 2 Duo P8400 processor at 2.26 GHz. Using the integral image approach, normals can be estimated rapidly for the 19,200 image pixels within approx. 11 msec in average. The extraction of horizontal points takes in average 2 msec. Limiting the search space does not consume significant runtime. A more costly step is the application of RANSAC to find the support plane. It amounts to about 8 msec in average. Extracting the points in the support plane, constructing the convex hull of the plane, and extracting the points on objects above the support polygon again require low runtimes of about 2 to 3 msec. The clustering of the points into objects takes about 12 msec (avg.). The computation time in this step depends on the number of objects in the scene. Our approach to grasp planning requires computation time in the same magnitude as the segmentation, i.e., 98 msec (avg.). The timings demonstrate that our approaches are very performant and yield results in short computation times. Fig. 8 shows the average runtime on different scenes. It can be seen that our method yields low runtimes irrespective of the scene content.

We also measured the time for the complete object pick-up process. The robot has already approached the table. It perceives the objects on the table and plans a grasp on the closest object in front. It executes the grasp and moves the gripper back to its starting pose. The overall process takes approx. 15 sec for a side-grasp and 25 sec for a top-grasp.

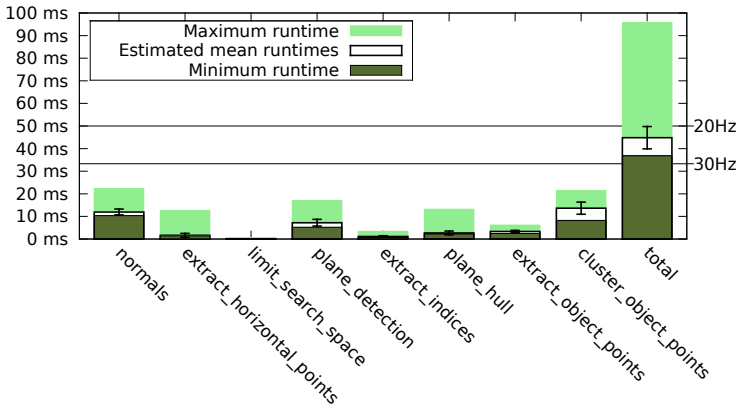


Figure 8: Measured runtimes in ten different scenes for the individual processing steps in the proposed perception pipeline. The scenes range from an empty room, over a close empty support plane to a table being farther apart (2.5 m away from the robot) and a cluttered table scene with ten objects. For each scene, 10 000 range images have been acquired and segmented. Shown here, are the average runtimes with minimum and maximum over all ten settings and all acquired range images (160×120).

6.1.2. Robustness

We evaluate the robustness of our perception and grasp planning pipeline in a series of experiments. In a tabletop setting, we chose eight typical household objects and executed 12 grasps with the right arm at 30° orientation intervals. Fig. 9 shows an example grasp for each object. Fig. 10 and Table 1 summarize the grasping success obtained in the experiment. The robot could grasp the objects reliably within its kinematic constraints. For the tea box and the banana, it would have to perform top-grasps for orientations that are kinematically infeasible with the right arm. Instead, the robot tries the closest kinematically feasible grasp possible. In some of these orientations, it fails to grasp the object. Note that the robot could better grasp the object with the left arm in these cases, since the range of achievable end-effector orientations is symmetric around the end-effector's yaw axis. Despite the fact that the clothes strongly violate our assumptions on the rigidity of objects, our method succeeds in 11 out of 12 cases in grasping this object.

We further evaluated the robustness of our approach in a shelf (see Fig. 11 and Table 2). We placed a tea box, a cup, a bowl, and a pen in 30° orientation intervals in the left and right shelf corners. By this, the grasp planner needs to consider collisions with the walls of the shelf. The object centers had a distance of 20 cm to the left or right wall and were placed 15 cm in front of the backside wall. In the left corner, the planner for the right arm yields collision-free grasps for all object orientations and succeeds in all but one trial. For the tea box, the execution of one grasp failed due to kinematic limitations of the right arm in this orientation of the object, similar to the tabletop setting. On the right side of the shelf, the situation is more challenging. Some orientations of the tea box and the pen were not feasible for our planner. In these orientations, no side-grasps are possible, and the right arm would collide with the outer wall during the execution of the top-grasp motion primitive. Please note that the situation would be easier with the left

Figure 9: Example grasps and grasp planning results for each of the eight household objects used in the experiments. For each object, we visualize the best grasp that is executed (larger RGB axes) and sampled grasps (smaller red axes) by their pre-grasp pose.

arm. For one orientation of the pen, the grasp planner yielded a top-grasp on the left tip of the pen, but failed to grasp it due to small kinematic inaccuracies. Our approach successfully executed 37 out of 38 planned grasps.

6.2. Public Demonstration

In recent years, competitions such as the DARPA Grand and Urban Challenges and RoboCup play an important role in assessing the performance of robot systems. While one can assess the quality of individual system components in the laboratory, it is often difficult to compare between different robot systems. In many competitions, details on the setup are not known in advance, such that the participants have to develop robust methods that perform well under many conditions.

The international RoboCup competitions include the @Home league for domestic service robots. In this competition, the robots have to perform tasks defined by the

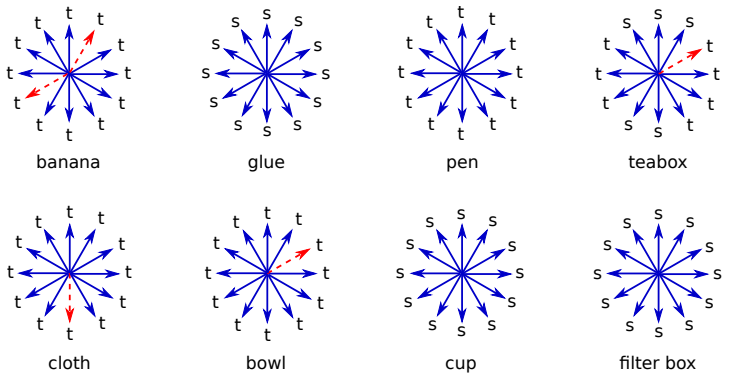


Figure 10: Grasping success (solid blue arrow) and failure (dashed red arrow) with the right arm from a tabletop (s: side-grasp, t: top-grasp). We placed the objects in 30° orientation intervals. The object orientation is indicated by arrows. The upward direction corresponds to the forward facing direction of the robot.

object	side-grasp	top-grasp
banana	0 / 0	10 / 12
filter box	12 / 12	0 / 0
tea box	5 / 5	6 / 7
cup	12 / 12	0 / 0
glue	12 / 12	0 / 0
bowl	0 / 0	11 / 12
cloth	0 / 0	11 / 12
pen	0 / 0	12 / 12

Table 1: Success rates (success / trials) when grasping objects 12 times in various orientations with the right arm from a tabletop.

rules of the competition, in a given environment at a predetermined time. In addition, there are open challenges and the final demonstration, where the teams can highlight the capabilities of their robots in self-defined tasks. The simultaneous presence of multiple teams allows for a direct comparison of the systems by measuring objective performance criteria, and by subjective judgment of the scientific and technical merit by a jury.

With our robots Cosero and Dynamaid, we won the RoboCup@Home competitions at GermanOpen 2011 and 2012, and at RoboCup 2011 in Istanbul, Turkey. In the finals of the 2011 RoboCup@Home competition at GermanOpen, Cosero and Dynamaid prepared breakfast within the 10 min demonstration slot. Dynamaid fetched orange juice out of the refrigerator, which it opened and closed successfully, and brought it to the breakfast table. In the meantime, Cosero grasped a bottle of milk, opened the bottle, and poured the milk into a cereal bowl. Cosero disposed the empty bottle into the trash bin. It then moved to another table and successfully grasped a spoon with a top-grasp. A jury member placed the spoon in an arbitrary orientation. Cosero put the spoon next to the cereal bowl and finally waited for an instruction to leave the room. Another

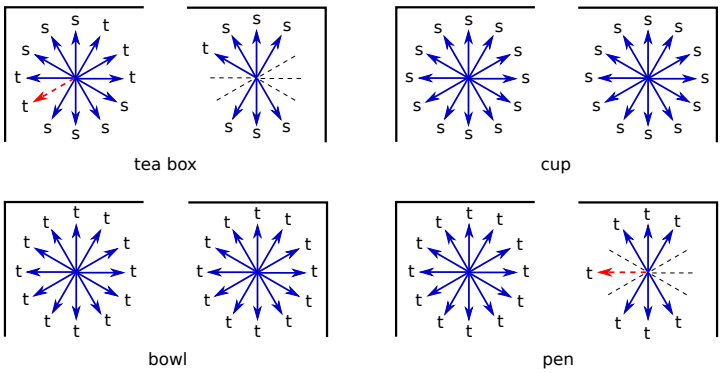


Figure 11: Grasping success (solid blue arrow) and failure (dashed red arrow) with the right arm from the left and right corners of a shelf (s: side-grasp, t: top-grasp). We placed the objects in 30° orientation intervals. The object orientation is indicated by arrows. The upward direction corresponds to the forward facing direction of the robot. Dashed black lines indicate in which orientations no feasible grasps could be found by the planner due to possible collisions.

	left corner			right corner		
object	side-grasp	top-grasp	no grasp	side-grasp	top-grasp	no grasp
tea box	7 / 7	4 / 5	0	6 / 6	1 / 1	5
cup	12 / 12	0 / 0	0	12 / 12	0 / 0	0
bowl	0 / 0	12 / 12	0	0 / 0	12 / 12	0
pen	0 / 0	12 / 12	0	0 / 0	6 / 7	5

Table 2: Success rates (success / trials) when grasping objects 12 times in various orientations with the right arm from the left and right corners of a shelf. In some situations, no valid grasp could be found by the planner due to possible collisions.

jury member pointed towards one of two exit doors using a pointing gesture. Cosero successfully recognized the pointing gesture and left the room through the correct door. The jury awarded us the highest score for the finals.

In the Open Challenge of the @Home competition at RoboCup 2011 in Istanbul, Cosero demonstrated to prepare cereal in front of a jury of team leaders. In the Demo Challenge, Cosero cleaned up the appartement by picking up laundry from the ground and putting it into the correct laundry basket. A human user could before show in which baskets to put colored and white laundry using gestures. Afterwards, Cosero picked up three objects from a table using the perception and grasping pipeline proposed in this paper. In the first attempt to pick up a carrot, it had to choose a grasp perpendicular to the carrot's principal axis and failed to keep grip of the object. However, in the second attempt, it picked up the carrot successfully along its principal axis. Finally, it grasped a tea-box with a top-grasp. The objects have been placed randomly. We could convince the jury with this demonstration and achieved the highest score.

At RoboCup GermanOpen 2012, our robot Cosero demonstrated the described grasp planning with our new gripper design in the Demo Challenge. Cosero received the highest

score in this test. Overall, we won the RoboCup@Home competition at GermanOpen 2012. [1]

7. Conclusion

In this article, we proposed highly efficient means to perceive objects on planar surfaces and to plan feasible, collision-free grasps on the object of interest. We integrate our methods into a mobile manipulation system that robustly executes object pick-up in reasonable time without longer processing delays, i.e. interruptions in the seconds, as they often occur with state-of-the-art motion planning approaches.

For object perception, we segment depth images in real-time at a frame rate of up to 20 Hz. We demonstrated that our perception and planning modules yield their results in a very short time. In the integrated system, this allows for short and steady execution of the task. In experiments, we demonstrate that our method is fast yet robust. We found that our perception and grasping pipeline in combination with a compliant gripper design is suitable to pick-up a large variety of typical household objects.

We experienced from our integration efforts that our method is easy to set up. Most parameters have an intuitive meaning and can be set empirically. Automatic refinement of the parameters from success and failure could further improve the usability of our approach. We already detect rare failures to grasp the object from IR sensor readings in the gripper. In such a case the robot tries to grasp the object again.

Our approach is well suited for situations in which the object of interest is spacially well distinguishable from others and the support plane is visible. It also provides grasps in constrained spaces such as shelves. Our collision-checking method would reject grasps that are not directly reachable with the implemented motion primitives. For such cases, we integrate more time-expensive motion planning [14]. To grasp objects in highly cluttered scenes, like piles of objects on tables or in bins, further segmentation cues such as top-down object knowledge [25] or curvature and color-contrast could be used [5].

In future work, we will study how to transfer concepts from our approach to the grasping and manipulation with strong task constraints, such as tool-use or placing objects in a specific way. This will afford the incorporation of information about the object and the task into perception and planning.

Acknowledgements

This research has been partially funded by the FP7 ICT-2007.2.2 project ECHORD (grant agreement 231143) experiment ActReMa.

References

[1] Beetz, M., Klank, U., Kresse, I., Maldonado, A., Mösenlechner, L., Pangercic, D., Rühr, T., Tenorth, M., 2011. Robotic roommates making pancakes. In: Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots.

[1] Videos including these demonstrations can be found at http://www.youtube.com/watch?v=zR_6IrJswU4, http://www.youtube.com/watch?v=nG0mJiODrYw, and http://www.youtube.com/watch?v=q041IvZ_FVU.

[2] Borst, C., Fischer, M., Hirzinger, G., 2005. Efficient and Precise Grasp Planning for Real World Objects. Multi-point Interaction with Real and Virtual Objects, Springer Tracts in Advanced Robotics.
[3] Chitta, S., Jones, E., Ciocarlie, M., Hsiao, K., June 2012. Mobile manipulation in unstructured environments: Perception, planning, and execution. IEEE Robotics & Automation Magazine 19 (2), 58–71.
[4] Fischler, M. A., Bolles, R. C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24 (6), 381–395.
[5] Holz, D., Behnke, S., 2012. Fast range image segmentation and smoothing using approximate surface reconstruction and region growing. In: Proceedings of the 12th International Conference on Intelligent Autonomous Systems (IAS).
[6] Holz, D., Holzer, S., Rusu, R. B., Behnke, S., 2012. Real-time plane segmentation using RGB-D cameras. In: RoboCup 2011: Robot Soccer World Cup XV. Vol. 7416 of Lecture Notes in Computer Science. Springer, pp. 307–317.
[7] Holz, D., Schnabel, R., Droeschel, D., Stückler, J., Behnke, S., 2011. Towards semantic scene analysis with time-of-flight cameras. In: RoboCup 2010: Robot Soccer World Cup XIV. Vol. 6556 of Lecture Notes in Computer Science. Springer, pp. 121–132.
[8] Hsiao, K., Chitta, S., Ciocarlie, M., Jones, E. G., 2010. Contact-reactive grasping of objects with partial shape information. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Taipei, Taiwan, pp. 1228–1235.
[9] Jain, A., Kemp, C. C., January 2010. EL-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. Autonomous Robots 28, 45–64.
[10] Kuhn, H. W., 1955. The hungarian method for the assignment problem. Naval Research Logistics Quarterly 2 (1), 83–97.
[11] Leeper, A., Hsiao, K., Ciocarlie, M., Takayama, L., Gossow, D., 2012. Strategies for human-in-the-loop robotic grasping. In: Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI). Boston, MA, USA.
[12] Miller, A., Allen, P., December 2004. GraspIt! A versatile simulator for robotic grasping. IEEE Robotics Automation Magazine 11 (4), 110–122.
[13] Morales, A., Asfour, T., Azad, P., Knoop, S., Dillmann, R., 2006. Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Beijing, China, pp. 5663–5668.
[14] Nieuwenhuisen, M., Stückler, J., Berner, A., Klein, R., Behnke, S., 2012. Shape-primitive based object recognition and grasping. In: Proceedings of the German Conference on Robotics (ROBOTIK).
[15] Okada, K., Kojima, M., Sagawa, Y., Ichino, T., Sato, K., Inaba, M., 2006. Vision based behavior verification system of humanoid robot for daily environment tasks. In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots.
[16] Pelossof, R., Miller, A., Allen, P., Jebara, T., 2004. An SVM learning approach to robotic grasping. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). pp. 3215–3218.
[17] Rusu, R. B., Blodow, N., Marton, Z. C., Beetz, M., 2009. Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in human environments. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). St. Louis, MO, USA, pp. 1–6.
[18] Rusu, R. B., Sucan, I. A., Gerkey, B., Chitta, S., Beetz, M., Kavraki, L. E., 2009. Real-time perception-guided motion planning for a personal robot. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4245–4252.
[19] Sahbani, A., El-Khoury, S., Bidaud, P., 2012. An overview of 3D object grasp synthesis algorithms. Robotics and Autonomous Systems 60 (3), 326–336.
[20] Saxena, A., Driemeyer, J., Ng, A., 2008. Robotic grasping of novel objects using vision. The International Journal of Robotics Research 27 (2), 157.
[21] Schnabel, R., Wahl, R., Klein, R., 2007. Efficient RANSAC for Point-Cloud Shape Detection. Computer Graphics Forum 26 (2), 214–226.
[22] Srinivasa, S., Ferguson, D., Helfrich, C., Berenson, D., Romea, A. C., Diankov, R., Gallagher, G., Hollinger, G., Kuffner, J., Vandeweghe, J. M., January 2010. Herb: a home exploring robotic butler. Autonomous Robots 28 (1), 5–20.
[23] Srinivasa, S., Ferguson, D., Vandeweghe, J. M., Diankov, R., Berenson, D., Helfrich, C., Strasdat, H., 2008. The robotic busboy: Steps towards developing a mobile robotic home assistant. In: Pro-

ceedings of the International Conference on Intelligent Autonomous Systems (IAS). Baden-Baden, Germany.

[24] Stückler, J., Behnke, S., 2009. Integrating indoor mobility, object manipulation, and intuitive interaction for domestic service tasks. In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots (Humanoids). Paris, France, pp. 506–513.

[25] Stückler, J., Behnke, S., 2010. Combining depth and color cues for scale- and viewpoint-invariant object segmentation and recognition using random forests. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4566–4571.

[26] Stückler, J., Behnke, S., 2012. Compliant task-space control with back-drivable servo actuators. In: RoboCup 2011: Robot Soccer World Cup XV. Vol. 7416 of Lecture Notes in Computer Science. Springer, pp. 78–89.

[27] Sucan, I. A., Kalakrishnan, M., Chitta, S., 2010. Combining planning techniques for manipulation using realtime perception. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Anchorage, AK, USA, pp. 2895–2901.