Object Handling in Cluttered Indoor Environment with a Mobile Manipulator

Cristian Militaru
Technical Univeristy of Cluj-Napoca
Cluj-Napoca, Romania
Email: Cristian.Militaru@student.utcluj.ro

Ady-Daniel Mezei
Technical Univeristy of Cluj-Napoca
Cluj-Napoca, Romania
Email: adydani_mezei@yahoo.com

Levente Tamas
Technical Univeristy of Cluj-Napoca
Cluj-Napoca, Romania
Email: Levente.Tamas@aut.utcluj.ro

***Abstract*—This paper presents an indoor object handling application using a 7 degree of freedom lightweight robotic arm and a commercial depth sensor. The motion planning for such a complex arm is demanding, especially in a cluttered scenes, like the majority of human environments are. In order to cope with this task we designed and implemented the direct and inverse kinematics for the arm as well as the perception related modules using a depth sensor. Both the simulation and experimental parts are carried out in a common framework which ensures a light integration of different planning and spatial perception parts.**

I. Introduction

An emerging field within robotics is that of service robots especially the ones giving the opportunity to cooperate with humans. The human-robot interaction in the current state is mainly representative in the indoor environment, where the interaction with other objects is must. The object in this context can be defined as a physically separable union which is independent from its environment, such that it can be moved separately from the rest of its surroundings. Two main directions are characterising the human-robot interaction with respect to object handling: the object perception and object manipulation. The first part is mainly focusing on the object separation, i.e. segmentation while the second part is dealing with the grasping and planning in the 3D space.

The object segmentation from its surrounding is not always a trivial task: several research work from the computer vision domain are focusing on this problem from decades. A common approach for solving this problem is the passive segmentation, the camera parameters relative to the object are not modified. However such an approach may have several drawbacks, especially when one has to deal with complex scenes, such as the human environment in general [1]. Also a common problem for the segmentation and scene interpretation tasks is related to the computational speed and accuracy of the object boundaries. A possible solution to overcome these limitation is the *active perception* approach which allows change in the camera parameters and even the interaction with the scene in order to gain more reliable information about this [2].

More generally, the concept of active perception can be found also in the human reasoning: one might move an object or move around an object in order to gain more information about this [3]. In case that one has a movable camera, i.e. a camera mounted on a robotic arm can take advantage of this extra degree of freedom, and can use it to gain getter representation about the surrounding objects, as this might be often needed. Nor the humans nor the surrounding scene cannot be expected to perform by itself a movement which helps the perception system to gain extra information, thus this has to be done by the robot itself.

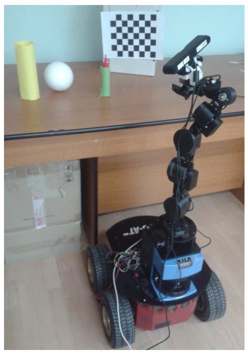


Fig. 1. The 7DoF robot arm model with the 3D depth sensor mounted on the top of it

These ideas motivate the introduction and usage of the active perception concept for the human-robot interaction [4]. Having the possibility to choose the camera external parameters for the perception system, i.e. move around the camera being mounted on a robotic arm, this leads to the ability of the robot to choose the way in which reasons about the working space.

In this paper we describe the method that we developed for an eye-in-hand mobile manipulator development containing a 7 degree-of-freedom (DoF) robotic arm used for object handling. The main contribution of this work was the development of kinematic setup for this device as well as the object detection and handling in the 3D working space.

In the first part of the paper we briefly describe the state of the art methods used for object detection and handling in indoor environment with mobile robotic manipulators. Then we describe the details of the setup that we used in order to develop the necessary real life experimental setup. Further on we show the details both for the object detection as well as the planning approach for the object handling problem. Finally, the paper is concluded with the results of these investigations.

978-1-4673-8692-0/16/$31.00 ©2016 IEEE

II. Problem description

The problem of object handling in the human-robot interaction context is far from being trivial, several aspects needs to be considered in order to have a solution to this problem [1]. Different approaches arise from either the camera mounting position with respect to the robot arm (i.e. on-board or off-board), the degree of clutter in the scene, or the planning approach (deterministic vs. probabilistic). A short overview of the related work in the main literature is presented in the next section.

A. 3D active sensing

The scene perception using active sensing approach is not a new concept, it dates back at least to the 80's [2]. Both the work in the 2D and in the 3D part has been extended especially in the mapping domain, where the map extension problem is referred as frontier based mapping [5].

In our approach we used 3D information sensing, so the in the next part we focus on the related literature. Also in the part of the research an important question is related to the best view selection from the scene, which is done for example by Potthast et al. [6], by searching each depth points back projected ray how much contribution has in terms of information gain. An extension to a single point planning approach is used in [7], where multiple position plans are computed in a single step, however the viewpoint selection and the path planning are done offline.

For dense feature matching a good overview can be found in [3]. In this paper the authors deal with a semi-complete scene map for which further information is gather in order to enhance its completeness. A similar eye-in-hand depth range sensor approach is presented in [8]: a dense map is generated with a projected light sensor and in the mean time a reasoning about the scene object is performed. This approach is similar to the one presented in the current paper too: an eye-in-hand 3D sensor for scene interpretation and object handling.

B. Planning with 7DoF arm

Object manipulation and grasping with robotic arm has a long history in the robotics community. There was a paradigm shift during the time from the deterministic way of planning such as the A^* towards the probabilistic planning methods in the main literature such as the Rapidly-Exploring Random Trees. A good overview of these techniques can be found in the book of LaValle [9].

A similar problem to the one discussed in this paper can be found in the works [10] focusing on planning issues in a cluttered scene and [11] showing demonstrations for a dual-arm planning problem. Our work is based on the concepts described in [12], which was developed within Robot Operating Systems (ROS) using the out-of-box planning algorithms such as the Open Motion Planning Library (OMPL) presented in [13]. Some other variants for planning algorithms can be found also in [14].

III. Arm control

Starting with a 7 DoF arm the development of the the forward and inverse kinematics is not trivial any more, thus approximate or sample based solutions such as the Rapidly–Exploring Random Tree (RRT) are often used [9]. This is why we adopted such a solution for our setup too.

A. Low level interface

The low level libraries for the Cyton arm is based on a action-server architecture, which communicates with the serial chained servo motors from the joints and is interfaced to the computer via the USB port.

At the motor driver level, we reused the already available ROS Dynamixel package, which could be customised in a flexible way in order to communicate with each joint individually by assigning an identifier (ID) to each motor in part. In the next level of control we used these references to address the physical joints of the arm [15].

B. Building the kinematics model

For the construction of the kinematics model of the robot arm we used two steps. The first one was building the physical structure based on the joint-link description language specific to the ROS environment. Next based on this description we configured the planning module together with the low level drivers in order to have a fully functional model for the motion planner.

For the first part of this task, i.e. the construction of the physical description we based our work on the existing models parameters from the commercial robot software package as well as the predefined universal robotics description from (URDF) for this type of arm presented in [12]. The later model contains two different module: one for the arm with rotational joint controls and one for the gripper with prismatic joint. The module controls are based on the individual joint level controllers and state feedback nodes for each node.

C. Planning algorithms

The arm has been tested with different planners from the Open Motion Planning Library(OMPL) that is available for use from MoveIt!.

a) RRT: A Rapidly–Exploring Random Tree(RRT) [16] is an algorithm and data structure successfully applied to path–planning problems that can contain obstacles, constraints of nonholonomic, kinodynamic nature. The algorithm tries to find a path between an initial starting point and a goal point by constructing a RRT that containing points from the configuration space used to reach the goal point. The RRT has its root in the specified starting point and its size grows iteratively with points selected randomly from the configuration space. Thus, the first point from the configuration space is selected and then a connection between it and the nearest point from the RRT is considered. If the connection satisfies the constraints the point is added to the RRT.

b) RRT–Connect: The RRT–Connect [17] algorithm is a variation of the RRT algorithm. The algorithm involves two RRTs, one having its root in the starting position while the other one has its root in the goal position. The algorithm consists of expanding the trees towards each other, until a connection between them appears, in such a case a path between the starting and goal points is found. The algorithms involves a greedy heuristic [17] in which a new extension towards a random point from the configuration space is done multiple times rather than a single time as in the case of the RRT.

c) PRM: In [18] a path–planning algorithm is introduced that is used to compute a path between two specified points, that avoids any collision called probabilistic road map (PRM). The algorithms works in two steps, in the first one another planner is used to build a graph data structure, called a roadmap [18] out of different randomly chosen collision free points from the configuration space. In the second step, the actual path between the starting point and the goal point is computed, by applying a shortest path algorithm, such as Dijkstra, on the previously constructed graph. The PRM is used efficiently and is suitable for multiple queries performed on the same configuration space.

d) KPIECE: The Kinodynamic Motion Planning by Interior–Exterior Cell Exploration(KPIECE) is a motion planning algorithm that is successfully applied to path–planning problems that involves high complexity constraints [19]. The planner performs a multi–level cell discretization of the configuration space, trying to explore different unexplored zones of the configuration space in the fastest time possible.

e) EST: In [20] a path planning algorithm named Expansive Space Trees(EST) is introduced. The EST is a single–query planner that is similar in its working with RRT–Connect. The planner constructs two trees rooted at the starting point and the goal point and expands them until they intersect. When this happens, a path from the initial point to the goal point has been found. Only the points from the configuration space that are related to the starting and goal points are inserted into the previously defined trees.

D. Integrating into the MoveIt!

The next step in the arm control development was the integration in the MoveIt! framework. This can be done based on the models developed in the previous step, and allows the use of different sampling based planning algorithms such as the RRT or different kinematic libraries. Beside the graphical interface based interactive planning, there is a dedicated API which can be used for planning.

IV. Depth information processing

The use of 3D perception sensors in the robotics application is widespread in the last few years mainly due to the available low price 3D cameras [21], [22]. In order to ensure the interaction with the robot arm in the Cartesian space, we used a commercial depth sensor with color. This ensures that the information in the working space of the robot is perceived in real time and with sufficient accuracy.

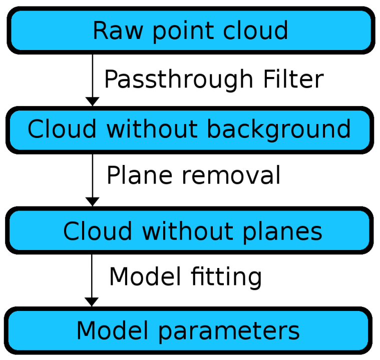


Fig. 2. The processing pipeline for the plane extraction algorithm

In this paper we focus on the already implemented processing components from PCL [23] library available with a BSD type license for RGB-D data. The main components of the processing chains included the raw data preprocessing, segmentation, and model fitting for indoor scenes with objects being sensed in $1m - 5m$ depth as this is presented in Figure 2.

Our goal was to detect a particular object in a scene provided by the sensor.

Having a point cloud from the camera, we need to search for our object in it. The searched object can be described either as a mathematical model (cylinder, sphere) or as another point cloud. In both cases, we desired to retrieve from the scene only the points that belong to the object. Depending on the description of the object and the restrictions about the scene contents, the following algorithms can be considered.

A. Plane extraction

The first scene we considered was a bottle standing on a tall box in the shape of a cuboid. The bottle had a cylindrical shape and the box can be seen as a set of planes. If we eliminate the points that belong to the box, we can obtain the points belonging to the cylinder.

The box was inside a room and the camera sensed other objects, such as walls, chairs etc. that were behind the box. The arm is supposed to grab the bottle, so we can ignore the points that are outside of the arm's working space. In order to remove the background, we used a PCL pass through filter that keeps only the points that have their z coordinate between $0.5m$ and $1m$.

After the filtering, the point cloud contained only our region of interest: the box and the bottle. From this, we could extract all the planes. Then most of the remaining points were belonging to the bottle. For this purpose, we used one of the planar segmentation algorithms provided by PCL. Providing a point cloud, the algorithm detects a group of points that belong to the same plane. By repeatedly applying the algorithm, we can remove all the planes in the scene.

The next step is finding the cylinder's parameters (the cylinder axis and radius). This requires fitting the remaining points to the mathematical model of a cylinder. We used the PCL cylinder segmentation algorithm, which is similar to the planar segmentation described above. The RANSAC was chosen for the search method, the distance threshold (maximum distance between the points and the cylinder model) was set to $2cm$. We configured the searched cylinder's radius to be between $1cm - 5cm$ (the bottle's radius was about $2.5cm$).

The algorithm works the same if there is a sphere instead of the cylinder. The only difference is the fitting step, in which we use the PCL sphere segmentation, with the following parameters: RANSAC search method, the distance threshold of $2cm$ and the radius between $5cm - 10cm$ (the sphere had a radius of about $7cm$). This algorithm has best results when the scene contains planes and, by removing them, most of remaining points belong to our object.

B. Region growing segmentation

This algorithm can be used to detect the object in a more complex scene, where there are many objects, which do not necessarily have a plane surface. Our program is designed to search for a model (cylinder or sphere) in this environment.

The Point Cloud Libray (PCL) region growing segmentation algorithm divides the point cloud into disjoint sets of points which belong to the same smooth surface, called clusters. In other words, it separates the smooth surfaces from the scene.

But what is considered a smooth surface? To answer this question, the algorithm uses the concept of surface normals. Consider that a surface represents the boundaries of a 3D object. The surface normal is defined for each point as the line perpendicular on the surface in that point. A line is perpendicular on the surface in a point if it is perpendicular on the plane that is tangent to the surface in that point. Therefore, the problem is reduced to estimating the tangent plane of the point cloud in each of its points. In order to find the tangent plane in a point, the PCL normal estimation algorithm searches for the neighboring points and finds the plane that minimizes the sum of distances from the points to that plane. The plane normal can be computed easily from the plane's coefficients.

A surface is considered to be smooth if, for each pair of two neighboring points, the tangent planes are similar, so their normals have a similar orientation. The "similarity" of normals is measured using the angle between their direction vectors.

But the angle between normals may not be enough to determine if the surface is smooth or not. Considering the following situation: a cylinder standing vertically on a plane. We would expect that the plane's points' normals to be vertical, and the cylinder's points' normals to be horizontal. But that does not happen in our case. The normals at each point are approximated using the neighbors of that point. The points of the plane that are very close to the cylinder will have their normals influenced by the points of the bottle. And the points of the cylinder that are close to the plane will have their normals influenced by the points of the plane. Therefore, the plane's points' normals will gradually become more and more horizontal as we get closer to the cylinder. Hence, if the algorithm was based only on angles between normals, it could consider the area between the two surfaces to be smooth, and therefore the cylinder and the plane would belong to the same smooth surface, which is not true.

To solve this problem, there are two options: set the angle threshold to a very low value (we had success with a value of $2.5°$), or use the curvature feature. The curvature of a surface at a point is a measure of the smoothness of the neighborhood of that point. The more the neighborhood of a point is similar to a plane, the smaller the point's curvature is. Therefore, only the points that are close to the intersection have a high curvature. We can use the feature by specifying a curvature threshold. The points that have a higher curvature than the threshold do not belong to any cluster. In our program, we achieved the best results with a curvature threshold of 0.07.

In order to divide the scene into smooth surfaces, the algorithm uses the region growing approach. Starting with a region containing only one point, called the seed, the alogithm expands the region by adding neighboring points that satisfy the smoothness constraints. When the region cannot be extended anymore (all the neighbors would violate the smoothness constraints), the cluster is complete, and the algorithm starts again with a different seed, until there are no more points in the cloud that have not been processed.

Having the groups of points that form smooth objects, we iterated through them, trying to fit each cluster to the cylindrical model. If most of the points of a cluster were fitted to the model, then we consider the object to be found.

C. Correspondence grouping detection

Another approach for object retrieval is the recent correspondence grouping one suggested in [24]. This method aims for detecting similarities between two or more objects. In this case we will use it to find a certain 3-D object (model) in a scene by grouping the correspondences in clusters, from which the pose will be extracted.

The descriptors for the keypoints are used to accurately determine the correspondences between points and hopefully find the match of the model in the scene, and must be robust to noise, resolution variation, translation and rotation. The algorithm that we used was SHOT (Signatures of Histograms of Orientations), which proved to be robust enough for the experimental part.

In the last step the correspondence grouping algorithm classifies all the correspondences that are thought to belong to the model into clusters, and rejecting the ones that are not. For this we have used the Hough algorithm which relies on a Hough Voting process, that outputs the rotation matrix and the translation vector. This aims at gathering evidence about the existence of the initial object in the plane by voting if enough feature correspondences are present.

Unfortunately this method seems to run for several tens of seconds which is far larger the the most unfavorable one from the two other methods, so we decided not using it for our real life experiments.

V. Experimental validation

In the first stage of our investigations we used also simulated robot arm, in order to test the object handling algorithms. A typical figure with our 3D simulated environment containing the robot arm, the 3D depth camera and an object of interest is visible also in Figure 4.

In the next steps we performed out real life experiment on a Cyton Gamma R2 seven degree of freedom robot arm and using an Asus Xtion Pro depth camera in a typical indoor environment. In the first part of the real life experiments we had to determine the external camera parameters relative to the gripper fingers of the robot arm.

A. Hand-eye camera calibration

The camera calibration in general has a broad interest in the computer vision and robotics community, several approaches existing for the intrinsic and extrinsic parameter estimation problems including the specific hand-eye calibration problem dating back already at the 80's [25].

In our approach to the calibration of the depth camera mounted on the arm and the gripper frame is based on a relative calibration to a fixed frame in space, as this is shown in Figrure 3. The main ideas is to use a calibration pattern as an external reference with an internally calibrated depth camera. The position of this checkboard and fixed frame with respect to the depth camera can be determined with standard methods from the computer vision field taking into account the physical size of the pattern.

Denoting with ${}_{C}\mathbf{T}^{F}$ the transformation from the camera to the fixed frame this can be used later on in computation of the ${}_{G}\mathbf{T}^{C}$ (gripper-camera transform) in a closed kinematic chain. The ${}_{B}\mathbf{T}^{G}$ transformation from the base frame to the gripper frame can be determined using the kinematic model of the robot. Further more, by moving the robot arm to at least four fixed corners of the calibration pattern, the position of the base with respect to the fixed frame ${}_{B}\mathbf{T}^{F}$ can be determined using a singular value decomposition (SVD) approach. Having all these transformations in the kinematic chain, the searched gripper camera transformation can be obtained using:

$${}_{G}\mathbf{T}^{C} = \left({}_{B}\mathbf{T}^{C}\right)^{-1} \times_{B} \mathbf{T}^{G} \quad (1)$$

By computing this transformation between the gripper and depth camera coordinate frames the camera can be integrated in the planning process easily.

B. Scene object retrieval

Further on we present the results of our investigations related to the object retrieval from the working space using the 3D perception pipeline described earlier.

The differences and special characteristics of the presented methods are shown in Figure 4 containing both the real experiment scenario as well as the object segmentation using the selected two approaches.

In the Tables I we summarized the results of our test cases for the detection rate of the suggested approaches from five different viewpoints about the same scene. The first columns contain the results from the scene with only one object of interest, while the last two columns show the results for the retrieval focusing on a specific type of object in a multi-object scenario.

Finally, the Table II summarizes the results from the standard deviation for the detected object's geometric center using different viewpoints about the same scene by moving the arm around the objects.

As one can observe the region growing method tends to be more robust, although the runtime of this is slightly larger than the plane extraction variant for the same scenes.

C. Planning benchmark

In order to test the performance of the planning algorithms on the arm, ten points from the configuration space have been chosen. The test have been carried out for the same points in two scenarios one unconstrained which contained no obstacles and one constrained in which an obstacle has been placed.

In the Table III we summarized the runtime performance analysis for our experimental setup using a standard laptop. As one could expect, the planning time with a constrained working space is greater than the one without collision objects in the scene for the sampling based algorithms. The most affected variants are in this case the RRT and its derivatives. The planning times for each algorithm are less then 5 seconds, which in this case is the minimum for the robot arm displacement in general.

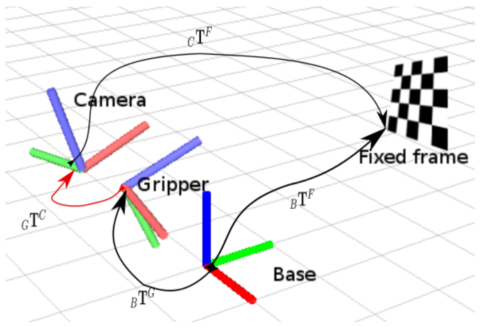


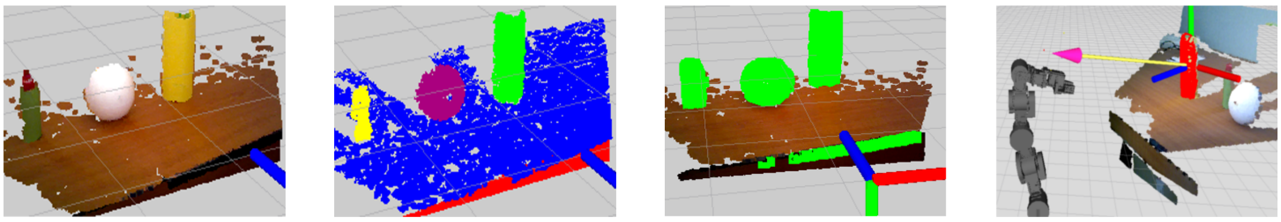
Fig. 3. The 7DoF robot arm model.

TABLE I
Detection rate for the scene objects with a single and multiple cylindrical and spherical object in the scene

	Cyl.1	Cyl.2	Sph.	Cyl.1 all	Sph. all
Plane ext.	5/5	4/5	5/5	4/5	5/5
Reg. grow	3/5	2/5	5/5	4/5	5/5

TABLE II
Standard deviation of the object centre used during the recognition from several viewpoints

	Cyl.1	Cyl.2	Sph.	Cyl.1 all	Sph. all
Plane ext.	0.03	0.43	0.01	0.48	0.55
Reg. grow	0.42	0.03	0.49	0.02	0.41



(a) Scene with objects (b) Region growing segmentation (c) Planar segmentation (d) Cylinder detection

Fig. 4. Detection scenarios for various different approaches.

TABLE III
PLANNING ALGORITHMS AVERAGE RUNTIMES IN SECONDS FOR THE 7 DOF ARM

	RRT	RRT–Connect	KPIECE	PRM	EST
Unconst.	2.18	4.13	2.48	2.24	2.13
Constr.	3.15	4.96	2.77	2.45	2.567

VI. CONCLUSION

In this paper we presented a custom solution for the object manipulation in an indoor environment using a 7 DoF commercial robot with a 3D depth camera mounted close to its gripper. We presented our solution to the scene parsing as well as planning related problems for the object manipulation using different approaches.

In the future we would like to extend our approach to a more natural human-robot interaction using advanced detection and planning algorithms being able to track objects moving in the humans hand and to interact with these objects in real life experiments.

ACKNOWLEDGEMENT

The authors would like to thank to the effort and support from their colleagues especially to Ioachim Ghisa and Elod Pall. The authors are also graceful for the support of the UTCN internal project CI1.2/2015.

REFERENCES

[1] D. Holz, M. Nieuwenhuisen, D. Droeschel, J. Stückler, A. Berner, J. Li, R. Klein, and S. Behnke, “Active recognition and manipulation for mobile robot bin picking,” in *Gearing up and accelerating cross-fertilization between academic and industrial robotics research in Europe:*, ser. Springer Tracts in Advanced Robotics, F. Röhrbein, G. Veiga, and C. Natale, Eds. Springer International Publishing, 2014, vol. 94, pp. 133–153.

[2] R. Bajcsy, “Active perception,” in *Proc IEEE, 76:996–1005*, 1988.

[3] E. Herbst and D. Fox, “Active object segmentation for mobile robots,” Tech. Rep., 2014.

[4] B. Sankaran, N. Atanasov, J. Le Ny, T. Koletschka, G. Pappas, and K. Daniilidis, “Hypothesis testing framework for active object detection,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013.

[5] E. Herbst, P. Henry, and D. Fox, “Toward online 3-d object segmentation and mapping.” in *ICRA*, 2014, pp. 3193–3200.

[6] C. Potthast and G. S. Sukhatme, “A probabilistic framework for next best view estimation in a cluttered environment,” *J. Visual Communication and Image Representation*, vol. 25, no. 1, pp. 148–164, 2014.

[7] N. Atanasov, B. Sankaran, J. Le Ny, T. Koletschka, G. J. Pappas, and K. Daniilidis, in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, May 2013.

[8] R. Monica, J. Aleotti, and S. Caselli, “A kinfu based approach for robot spatial attention and view planning,” *Robotics and Autonomous Systems*, vol. 75, pp. 627–640, 2016.

[9] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[10] Y. Hirano, K. Kitahama, and S. Yoshizawa, “Image-based object recognition and dexterous hand/arm motion planning using rrts for grasping in cluttered scene,” in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, Alberta, Canada, August 2-6, 2005*, 2005, pp. 2041–2046.

[11] R. Zollner, T. Asfour, and R. Dillmann, “Programming by demonstration: Dual-arm manipulation tasks for humanoid robots,” in *In Proceedings of the International Conference on Intelligent Robots and Systems*, 2004.

[12] G. Kanter, “Robot manipulator control using stereo visual feedback,” Tech. Rep., 2012.

[13] I. A. Sucan, M. Moll, and L. E. Kavraki, “The Open Motion Planning Library,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012.

[14] K. Hauser, “Recognition, prediction, and planning for assisted teleoperation of freeform tasks,” *Autonomous Robots*, vol. 35, no. 4, pp. 241–254, 2013.

[15] A. Fekete, “Robot grasping and manipulation of objects,” Tech. Rep., 2015.

[16] S. M. Lavalle, “Rapidly-exploring random trees: A new tool for path planning,” Tech. Rep., 1998.

[17] J. J. K. Jr. and S. M. Lavalle, “Rrt-connect: An efficient approach to single-query path planning,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 995–1001.

[18] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” in *IEEE Int. Conf. on Robotics and Automation*, 1996, pp. 566–580.

[19] I. A. Sucan and L. E. Kavraki, “Kinodynamic motion planning by interior-exterior cell exploration,” in *Algorithmic Foundation of Robotics VIII*. Springer, STAR 57, 2009, pp. 449–464.

[20] D. Hsu, J. Latombe, and R. Motwani, “Path planning in expansive configuration spaces,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1997, pp. 2719–2726.

[21] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena, “Contextually Guided Semantic Labeling and Search for 3D Point Clouds,” *International Journal of Robotics Research*, 2013.

[22] H. Ali, F. Shafait, E. Giannakidou, A. Vakali, N. Figueroa, T. Varvadoukas, and N. Mavridis, “Contextual object category recognition for RGB-D scene labeling,” *Robotics and Autonomous Systems*, vol. 62, no. 2, pp. 241–256, Feb. 2014.

[23] R. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 1 –4.

[24] A. Aldoma, F. Tombari, J. Prankl, A. Richtsfeld, L. Di Stefano, and M. Vincze, “Multimodal cue integration through hypotheses verification for rgb-d object recognition and 6dof pose estimation,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2104–2111.

[25] R. Y. Tsai and R. K. Lenz, “Real time versatile robotics hand/eye calibration using 3d machine vision,” in *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*. IEEE, 1988, pp. 554–561.