An Autonomous Mobile Manipulator for Collecting Sample Containers

Sankaranarayanan Natarajan*, Sebastian Kasperski*, Markus Eich*

*DFKI GmbH, Robotics Innovation Center, Bremen, Germany
e-mail: firstname.lastname@dfki.de

Abstract

In this work we present an autonomous mobile manipulator that is used to collect sample containers in an unknown environment. The manipulator is part of a team of heterogeneous mobile robots that are to search and identify sample containers in an unknown environment. A map of the environment along with possible positions of sample containers are shared between the robots in the team by using a cloud-based communication interface. To grasp a container with its manipulator arm the robot has to place itself in a position suitable for the manipulation task. This optimal base placement pose is selected by querying a precomputed inverse reachability database.

1 Introduction

In recent years, mobile manipulation is getting more attention in the field of space exploration. Future space robots play a critical role in collecting, distributing and maintaining components in extraterrestrial environments. The advantage of a mobile manipulator is not only the increased workspace of the robot but also the capability to place itself in a position that provides a collision free environment for the manipulator. The complexity of the manipulation task is increased due to the additional degrees of freedom and uncertainty in sensors and actuators. These challenges attract more researchers in the mobile manipulation field. The state of the art of mobile manipulation has been advanced in recent years. Some of the most advanced state of the art mobile manipulators like PR2, Justin, HRP2, HERB and ARMAR are able to perform complex manipulation, grasping and navigation tasks.

The work presented here has been done during the space project IMPERA (Integrated Mission Planning using heterogeneous robots)[1]. The focus of the project IMPERA is to develop a strategy for distributed mission and task planning. In this project a heterogeneous team consisting of the mobile manipulator AMPARO (Autonomous Manipulation Robot) and two SCOUT robots is used. The task planning for the mobile manipulator is divided into two major subtasks: moving the mobile base near a target pose and then manipulating the object. This approach is easy to implement because all these subtasks are independent of each other. The disadvantage is that the entire task may fail occasionally. Sometimes the mobile base moves near the target object but the manipulator cannot reach the target object because no inverse kinematic solution exists. Thus the mobile base placement plays an important role in the overall motion planning problem. This problem can be solved by combining the mobile base placement task and manipulation task together. Bereson, et al. [1] proposed an optimization-based method by coupling subtasks for finding optimal grasp positions and base placements. Vahrenkamp, et al. [15] developed a reachability distribution to identify oriented base poses for a target grasping pose. In our approach we used the inverse reachability module from OpenRAVE [3] which computes a distribution of possible robot base placements for a given grasp pose. From this distribution of possible base placements we choose an optimal one which is explained in the following section.

The work presented in this paper is focused on methods and results of an autonomous mobile manipulator robot capable of collecting and distributing sample containers with the help of a two SCOUT robots. These SCOUT robots are used since they can navigate faster and with less energy than the AMPARO robot through the unknown territory. System description and system architecture are explained in Section 2. In Section 3 the exploration and navigation part is explained. The mobile manipulation software modules are described in Section 4. Experiments and conclusion are given in Section 5 and 6.

2 System Description

A mobile manipulator is a robotic system in which a robotic manipulator is placed on a mobile platform. Figure 1a shows the autonomous mobile manipulator robot AMPARO. As mentioned in the introduction AMPARO will work together with two similar SCOUT robots. Figure 1b shows one of the scouts. All of the robots are based on the Pioneer3-AT platform. Table 1 shows an overview of the sensors available in the team of robots.

The manipulator mounted on the mobile base is a Jaco manipulator from Kinova[2] with six degrees of freedom.

[1]http://robotik.dfki-bremen.de/en/forschung/projekte/impera.html

[2]Jaco research edition robotic arm, http://kinovarobotics.com/

(a) AMPARO (b) Scout

Figure 1. : Heterogeneous Robot Team

Sensors	Amparo	Scout 1	Scout 2
Manipulator	yes	no	no
Camera	no	yes	yes
2D Laser Scanner	yes	yes	yes
3D Sensor	yes	no	no

Table 1. : Sensors available in the robot team

The Jaco manipulator has a gripper with three fingers, each having one degree of freedom, attached to it. For 3D Perception Amparo is equipped with a 3D LIDAR system consisting of a 2D laser range finder (Hokuyo UTM-30LX with 30m range) mounted on a Direct Perception DP-46 pan-tilt unit. An additional *Sick* LMS-111 laser system is mounted in a low, horizontal position to provide 2D navigation. The Scout robots use the same 2D LIDAR system (Hokuyo UTM-30LX) for the exploration task and a color camera (Guppy C36 with a resolution of 752 x 480 pixels) for locating possible sample containers.

Figure 2. : Sample Container

The sample container described in this paper is a cylindrical structure as shown in Figure 2 with a diameter of 0.06 meters and height of 0.30 meters.

2.1 System Architecture

The system architecture shown in Figure 3 describes the software components running on the different robots. All the robot use the Robot Operating System (ROS) as framework. The Scout robots have an additional distributed SLAM (simultaneous localization and mapping) module to cooperatively build a map of the explored area [8].

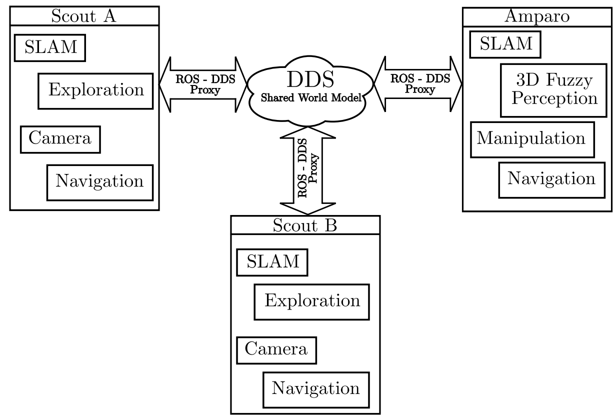


Figure 3. : Amparo

Hence the communication between the team members is critical and the robots should be able to continue their work in case of a communication failure. To achieve such a reliable link, the cloud-based communication system "Data Distribution Service" (DDS) [11] is used. A detailed description of the application of DDS in the communication between robots is described in [6].

3 Exploration and Navigation

3.1 Motivation

For the sample return scenario we assume that neither the topology of the surrounding environment nor the position of the targets is known in advance. Therefore the sample containers have to be located individually before they can be collected by the Amparo robot. We further assume that the search for the targets is performed by a number of smaller robots with only sensing capabilities. Once the Amparo is within sensing range of a target, it can use its 3D laser scanner to position itself in a way suitable for manipulation. Reaching the target from its current position in an efficient and secure way on the other hand requires a map and the target's position within this map. This map is created with a distributed, graph based mapping approach that uses the reliable communication of the used DDS framework.

3.2 Distributed mapping

Mapping in general is the process of integrating incoming sensor information, in our scenario laser scans and odometry information, into a consistent world model. When done with only one robot, the collected data can be processed locally and the map that is continually advanced can be used to find new exploration targets [16] and navigate towards them. Most offline mapping algorithms can also be used to integrate data from different robots, although the computational expenses might grow exponentially. If the robots are to share information defined on the map, for example their current location or target positions, all robots need to share the same map. This can be achieved in a straight forward way if only one robot performs the mapping and sends the created map to all other team members. Such a centralized approach has two major disadvantages: it requires a lot of communication bandwidth to send the map to all robots and the mapping requires a constant connection of all robots to the master, because no map updates can be processed otherwise.

The solution for this is a distributed approach, where all robots perform the mapping locally and collected information is send to all robots instead of the master. To realize such an online mapping with multiple robots graph based mapping approaches [5] are inherently well suited. The world model is continually built up by adding new measurements as nodes to the internal pose graph. Incoming data is registered only with the last few scans (sequential scan matching) while registration with the rest of the world model (global optimization) is delayed. This allow to send every added measurement to all other robots regardless of the current localization uncertainty to have the same map on all robots. Global optimization is done on every robot individually using Sparse Pose Adjustment [9], a SLAM centered variant of the Sparse Bundle Adjustment.

3.3 Navigation on partial maps

Common navigation approaches usually consist of two separate levels, a local and a global path planner. The global planner uses the available environment model to generate a path from the current position to the selected target. This plan is optimized with regards to an arbitrary cost function, which is in most cases the travelled distance but may include other aspects like terrain difficulty or safety of the generated plan. During the navigation process the generated plan is refined by a local planner, which includes recent sensor readings and a usually finer grained map of the robot's vicinity.

During map generation there is a very common situation when a robot is driving directly into uncharted terrain in order to expand the world model. In this case the global planner is of no use at all, because the area in front of the robot might not have been added to the world model yet and the robot has to rely solely on the local planner. As a result an exploration planner should not be designed to work on top of a regular navigation setup by sending goals to global planner. Instead it should be able to send commands to the local planner directly when extending the explored area at one of its frontiers. Only when an area is completely explored and the robot needs to navigate towards another frontier, the regular navigation approach should be used.

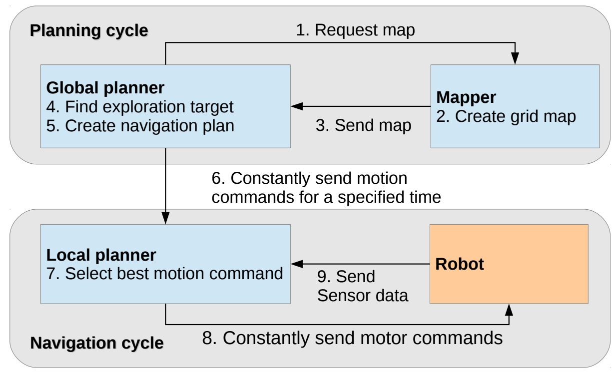


Figure 4. : **Navigation architecture:** A grid map is only generated after a request from the planner. Planning cycle and navigation cycle are loosely coupled and can run at different frequencies.

3.4 Request driven map generation

A common issue with graph based mapping algorithms is that generating the map from the underlying pose graph can become computational expensive for large maps. This is a problem for map exploration because the last added information to the map might be most important to plan the next exploration step. Especially creating an occupancy grid map with every newly added scan to the pose graph has proven completely impractical. Instead we propose a request driven approach where the planner (path or exploration planner) actively requests a new map when it starts a new planning cycle. This allow to reduce the generation of grid maps from the pose graph while still being able to use the most up-to-date map for exploration planning. Figure 4 shows the sequence of actions for a navigation cycle during the exploration phase.

3.5 Navigation during exploration

By using the graph based SLAM to realize a distributed mapping for the team of robots we are able to cooperatively create a map that is shared via DDS. This way the map is also available to the AMPARO where it can be used within a standard navigation approach. Theoretically the AMPARO could also be used as a scout to join in the exploration process if desired. However within our scenario we consider the energy of the AMPARO to be a limiting factor and will therefore only move it to reach the targets with minimal effort.

To share map updates between robots we already use the reliable communication via the DDS infrastructure.

This link can also be used to coordinate the actions of the robots during the exploration phase by sharing their current positions and exploration goals. By including these informations in the exploration planner, different exploration strategies can be applied to coordinate the multi-robot exploration. Details on the cooperative exploration and evaluation results of different strategies can be found in [8].

During the exploration phase the global planner should be restarted regularly for several reasons. When expanding a frontier the robot can only plan a few meters ahead and therefore has to plan again in order to include new information from the mapper. But even when moving towards a more distant frontier it is useful to recheck the current exploration target regularly because another robot might have already explored the area [7]. This way robots avoid driving long distances to far away frontiers that have already expanded or even completely explored by other members of the team.

4 Mobile Manipulation

Amparo's manipulation framework integrates several software modules to achieve the complex manipulation task. Figure 5 shows the schematic structure of this framework.

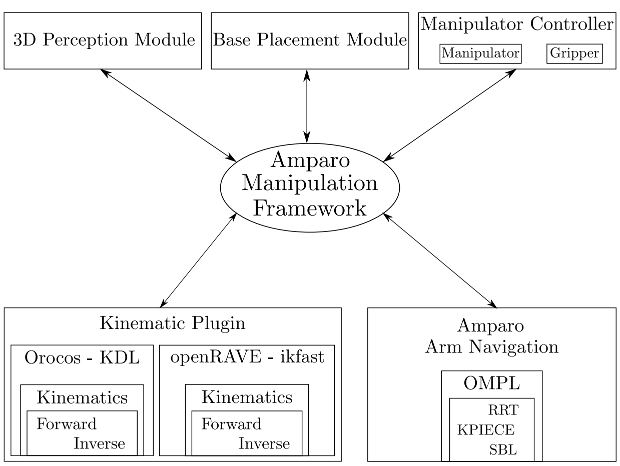


Figure 5. : Schematic structure of the Amparo manipulation framework

The manipulation controller is a module which wraps the Jaco manipulator's API in order to communicate with the manipulator. The API is used to get the current joint angle of the manipulator and its gripper and also to move the manipulator in joint space or in Cartesian space. The manipulator controller module accepts Cartesian poses or joint angles or trajectory as input. This module is also used to control the three fingers of the gripper in joint space. The joint current and position of the gripper is used to confirm the grasp process as AMPARO lacks external sensors for verifying the grasp task. When the sum of the finger's joint current exceeds a predefined threshold value and the finger's joint position is less than the target position it is assumed that the gripper has grasped the sample container. The threshold value for the finger joint current for the sample container has been identified by several experiments.

Kinematics describes the motion of rigid bodies without regarding the forces or torques which cause that motion. The kinematics of a robot manipulator describes the relationship between the motion of the joints of the manipulator and the resulting motion of the rigid bodies which form the robot [10]. The two problems in the kinematic of the manipulator are forward or direct kinematic and inverse or indirect kinematic. There are several open source kinematics solver such as [12], Orocos KDL module ([13]), robotics toolbox ([2]), openRAVE ([3]) are available. The kinematic plugin module give access to two different kinematic solver namely, openRAVE *ikfast* and Orocos KDL. The Jaco manipulator on AMPARO is non-redundant, so the inverse kinematic problem can be solved analytically. The computational cost for solving the inverse kinematic analytically is comparatively less compared to solving the inverse kinematic numerically. On AMPARO, we use openRAVE *ikfast* for solving the inverse problem analytically and the Orocos KDL module for solving the forward kinematic problem. The Orocos KDL module also provides a generic numerical inverse solver.

The motion planner provides a path for the manipulator to move from start position to goal position. The computed path should be collision-free with itself and its environment. The most commonly used planner in manipulation are sampling-based motion planners. The main advantages in using sampling-based motion planner are the computational cost is less and it can solve high-dimensional problem relatively fast compared to other types of motion planners. On AMPARO, a sampling-based motion planner OMPL [14] is used. The Amparo arm navigation module is developed using the arm navigation package from ROS. This package provides an interface to use the OMPL. The Amparo arm navigation keeps the representation of the current robot state by subscribing to joint angles of the manipulator and gripper and position of the pan-tilt unit. This state representation is essential in motion planning to avoid self-collision and constraint checking. The collision free trajectory generated by the motion planner is not necessary smooth and therefore the generated trajectory is passed through a cubic spline shortcutter filter. This filter removes random waypoints and smooths the waypoints using cubic spline in the generated trajectory and check if the smooth trajectory is collision free. Then this smoothed trajectory is send to the manipulator controller module.

The 3D perception module makes use of the 3D point cloud generated by 3D LIDAR system mounted on the top of AMPARO. The sample container is extracted from the point cloud by combining Description Logic (DL) based spatial reasoning approach with 3D feature extraction method. A detailed description of the 3D perception module methods and results are described in [4].

The base placement module is responsible for finding an optimal AMPARO base placement for detecting and grasping the sample container. The following two subsections will explain how optimal AMPARO base placement is calculated.

4.1 Base Placement for 3D Perception

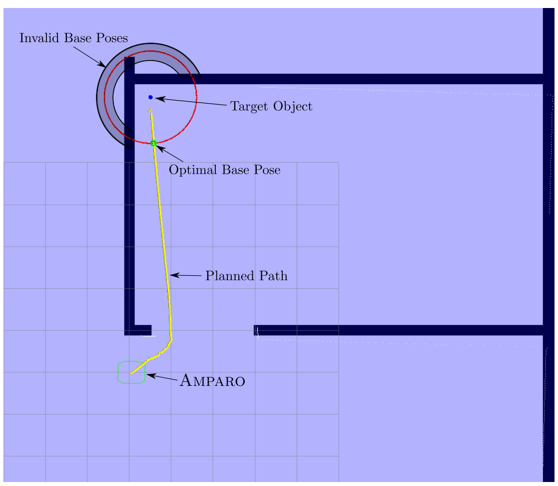
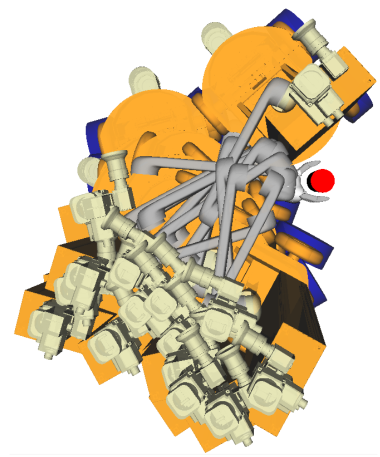


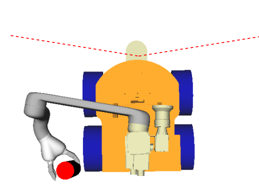
Figure 6. : Optimal Base Placement for 3D Perception

In order to detect the sample container, AMPARO uses its 3D LIDAR system located on the top of its black tower [see Figure 1a] to generate a point cloud. The 3D perception module uses this point cloud to detect the sample container. The optimal distance required by 3D perception module to detect the sample container is in the range of 0.5 to 1.5 meters from AMPARO's base frame. Since the map and possible candidate pose with respect to the map are known to AMPARO, a simple solution is to move AMPARO randomly to any pose which is 1 meter in front of the candidate and use it 3D Perception module to detect the sample container. The problem with this approach is that the chosen pose is not always obstacle free and the candidates is not always visible from AMPARO. Figure 6 shows an example of a simulated scenario in which the target object is marked. In Figure 6, the red circle with a radius of 1 meter represents the possible AMPARO base placement pose for 3D perception. As one can see not all the poses is valid for object detection. The area marked as "Invalid Base Pose" in Figure 6 represents, where sample container cannot be seen from AMPARO and some poses has obstacle. This problem was solved by generating a path to the target object from the global navigation planner. Then a circle with radius of 1 meter is generated with target object as centre. The intersection between the circle and global path will gives an optimal base placement which is free from the two problems described before. In Figure 6 the yellow line represents the global valid path from the AMPARO to the sample container and the green circle represents the intersection of the circle and the global path.

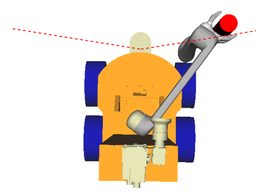
4.2 Optimal Base Placement for grasping



(a) Distribution of 20 Possible base placement poses for grasping.



(b) Ignored base placement pose



(c) Considered base placement pose

Figure 7. : AMPARO Base Placement Module

Optimal base placement for grasping modules uses openRave's Inverse Reachability module to get a distribution of all the possible base pose for a particular grasp, and pick an optimal pose from the possible base poses. The Inverse reachability takes end effector pose as input and returns a distribution on the 2D plane of where the base can be in order to achieve that particular end-effector pose [3]. Figure 7a shows possible AMPARO base placements for a particular grasping position. It shows 20 different base placements in total. After identifying the sample container using the 3D perception module, AMPARO can move to any random pose from the possible base placement poses in order to grasp the sample container. Due to odometry errors, the target base placement pose cannot be accurately reached. This error will affect th grasping pose. Therefore randomly chosen poses are not a solution. The simple solution to get rid of this error is, to get the object pose after reaching the base placement pose. AMPARO uses the front 2D laser scanner to verify the sample container pose before grasping the object. From all the generated possible base placements, only poses which are in the visibility of the laser scanner are considered. Figure 7b and 7c show two different AMPARO base poses with respect to the sample container. In the former case, the sample container is not in the laser scanner range of AMPARO and in latter case the sample container is in the AMPARO laser scanner range.

Algorithm 1: Optimal AMPARO base placement for grasping

```
Input: Pose of sample container P_s
Input: Target grasping pose P_g
Input: Current base pose P_c
Result: Optimal mobile base pose P_ob
foreach Task: Sample container picking do
    P_b ← Calculate all possible base placement
    based on the grasping pose
    foreach Possible base placement P_bi do
        R_bi ← Calculate rotation angle between P_s
        and P_bi
        if rotation angle R_bi is in between laser
        scanner range then
            The pose P_bi is stored in P_ni
        end
    end
    P_ob ← Find the minimum distance in P_ni with
    respect to current base pose
end
```

An optimal AMPARO base placement for grasping the sample container from the possible base placement is chosen based on Algorithm 1. The algorithm takes Pose of sample container P_s, grasp pose P_g and current base pose P_c as inputs and will give an optimal base pose P_{ob} as output. The algorithm will first calculate all possible base placement poses P_b for a particular grasp pose. Then a rotation angle R_{b_i} is calculated between each base placement pose P_{b_i} and sample container pose P_s. This rotation angle R_{b_i} is used to check whether the sample container is in AMPARO 2D laser scanner range, if it is in the range then the base placement pose P_{b_i} is stored. An optimal base placement pose is chosen from the stored pose based on the minimum distance between stored pose P_{n_i} and the current base pose P_c.

5 Experimental Results

The goal of the experiment is to identify and collect the sample containers in an unknown environment. In this experiment two SCOUT robots and AMPARO are used. All the robots start from a common starting area. At first one of the SCOUT robots starts exploring and generates a map of that area. Then the other SCOUT robot generates a map from shared laser scans it receives through DDS. As soon as the second SCOUT robot has localized itself in the generated map it joins with the first SCOUT robot in exploring and finding the possible sample containers. Figure 8 shows the generated map of the experiment area and the positions of the sample containers.

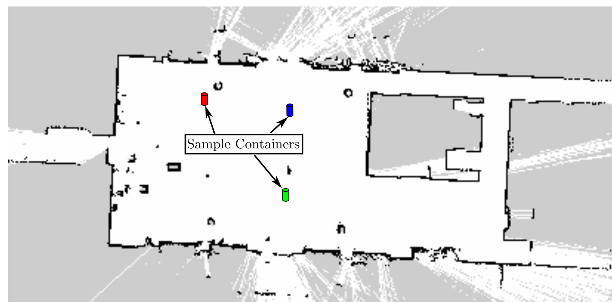


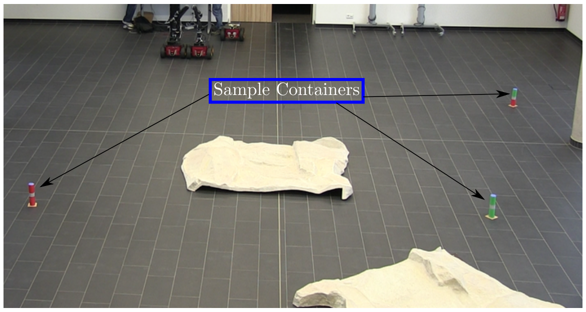
Figure 8. : The map of the experiment field (size: 41.4m x 20.4m)

The SCOUT robots use their cameras to find possible sample containers. Once they found a possible sample container they register the pose in the map and publish it via DDS.

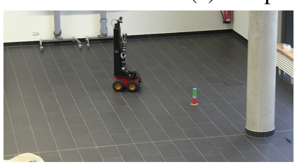
Figure 9a shows the sample containers needed to be collected. AMPARO receives the shared laser scans from the two SCOUT robots and thus is able to generate the common map. As soon as it receives the possible sample container pose it generates a base placement pose for 3D perception. In Figure 9b AMPARO reached 1 meter from the sample container and 3D perception is used to verify the object. Figure 9c shows the optimal base placement pose for grasping the sample container. The sample container is grasped (Figure 9d) and placed in the collecting area (Figure 9e). Similarly the other two sample containers are collected.

6 Conclusions

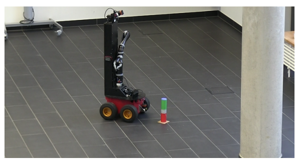
In this paper we explained the approach used by an autonomous mobile manipulator to find and pick a sample container in an unknown environment. Our approach was tested on the autonomous mobile manipulator AMPARO which is part of a heterogeneous team of mobile robots to identify and to grasp sample containers. In future, we will extend the grasping task by using a grasp planner to grasp more complex objects which are not consisting purely on basic geometric shapes. We intend to equip the gripper with additional sensors to get more information on the object. This additional information about



(a) Sample Containers

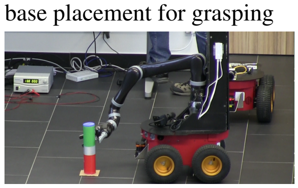


(b) AMPARO in place for 3D perception

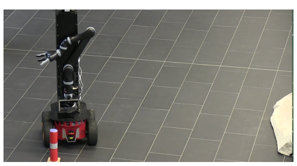


(c) AMPARO reached optimal base placement for grasping

(d) Grasping a sample container



(e) Placing the sample container in collecting area



(f) Picking the second sample container

(g) Placing the sample container in collecting area

Figure 9. : AMPARO collecting the sample containers

the object will be used in motion planning and optimal grasping pose.

7 Acknowledgement

This work is partially supported by the IMPERA project and the INCASS project (MOVE-FP7-605200-INCASS). The project IMPERA is funded by the German Space Agency (DLR, Grant number: 50RA1111) with federal funds of the Federal Ministry of Economics and Technology (BMWi) in accordance with the parliamentary resolution of the German Parliament.

References

[1] D. Berenson, H. Choset, and J. Kuffner. An optimization approach to planning for mobile manipulation. In *IEEE International Conference on Robotics and Automation (ICRA) 2008*, pages 1187–1192, May 2008.

[2] P. I. Corke. *Robotics, Vision & Control: Fundamental Algorithms in Matlab*. Springer, 2011.

[3] R. Diankov. *Automated Construction of Robotic Manipulation Programs*. PhD thesis, Carnegie Mellon University, Robotics Institute, August 2010.

[4] M. Eich, R. Hartanto, S. Kasperski, S. Natarajan, and J. Wollenberg. Towards Coordinated Multi-robot Missions for Lunar Sample Collection in an Unknown Environment. *Journal of Field Robotics*, 31(1):35–74, 2014.

[5] J. S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. *International Symposium on Computational Intelligence in Robotics and Automation*, pages 318–325, 1999.

[6] R. Hartanto and M. Eich. Reliable, Cloud-based Communication for Multi-Robot Systems. In *Technologies for Practical Robot Applications (TePRA), 2014 IEEE International Conference on*, Apr. 2014.

[7] D. Holz, N. Basilico, F. Amigoni, and S. Behnke. Evaluating the Efficiency of Frontier-Based Exploration Strategies. In *Proceedings of the joint conference of the 41st International Symposium on Robotics (ISR 2010) and the 6th German Conference on Robotics (ROBOTIK 2010)*, pages 36–43, Munich, Germany, June 2010.

[8] S. Kasperski, J. Wollenberg, and M. Eich. Evaluation of cooperative exploration strategies using full system simulation. In *Advanced Robotics (ICAR), 2013 16th International Conference on*, pages 1–6, Nov 2013.

[9] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent. Efficient Sparse Pose Adjustment for 2D Mapping. *International Conference on Intelligent Robots and Systems*, pages 22–29, 2010.

[10] R. M. Murray, Z. Li, and S. S. Sastry. A mathematical introduction to robotic manipulation, 1994.

[11] O. M. G. OMG. Data Distribution Service for Real-time Systems. Technical Report OMG Available Specification formal/07-01-01, OMG, 2006.

[12] ROBOOP. A robotics object oriented package in c++. `http://www.cours.polymtl.ca/roboop/`, 2012. visited 28.11.2012.

[13] R. Smits. KDL: Kinematics and Dynamics Library. `http://www.orocos.org/kdl/`, 2012. visited 28.11.2012.

[14] I. A. Şucan, M. Moll, and L. E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012. `http://ompl.kavrakilab.org`.

[15] N. Vahrenkamp, T. Asfour, and R. Dillmann. Robot placement based on reachability inversion. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1970–1975, May 2013.

[16] B. Yamauchi. A frontier-based approach for autonomous exploration. In *In Proceedings of the IEEE International Symposium on Computational Intelligence, Robotics and Automation*, pages 146–151, 1997.