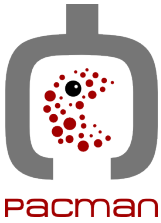

FP7-IST-60918
1 March 2013 (36months)

DR 5.1:
Grasping Objects Among Clutter

Mirela Popa, Justus Piater

Institute of Computer Science, University of Innsbruck
⟨Mirela.Popa@uibk.ac.at⟩

Due date of deliverable:	2014-02-28
Actual submission date:	2014-02-28
Lead partner:	UIBK
Revision:	final
Dissemination level:	PU

This report describes the software setup of the demonstrator Deliverable D5.1.

Executive Summary

This report describes the software setup of the demonstrator Deliverable D5.1 *Grasping objects among clutter*, defined in Task 5.1.

Role of the grasping demo in PaCMan

This demonstrator constitutes a baseline implementation of object detection, pose estimation, grasp planning, and grasp execution, upon which the actual PaCMan developments will build. Key parts of the system implemented in this demonstrator will be replaced by systems developed by the PaCMan project, allowing them to be benchmarked against this baseline.

Contribution to the PaCMan scenario

This is the first evaluation scenario and its implementing prototype defined by the PaCMan project.

1 Tasks, objectives, results

1.1 Planned work

This work addresses Task 5.1 *Grasping objects on a cluttered surface* (Months 7–12) as defined in the Description of Work:

This task will evaluate the baseline ability of our system to locate and grasp known object instances on mildly cluttered surfaces, under restricted visibility and accessibility.

1. Training of 3D models for the objects to be used in this scenario (using results from Tasks 1.4).
2. Initialization and exploratory refinement of grasp parameters and their integration into the object models.
3. Creation of a model of the experimental scene.
4. Among the grasps associated with the object models located in the scene, choice and execution of one or more grasps that are accessible under the scene constraints.

1.2 Actual work performed

1. Object models are trained using state-of-the-art methods using point cloud data.
2. Grasps are planned using a grasp planner recently developed at BHAM (Kopicki et al., 2014).
3. Scene models are aquired using point could data.
4. Grasps are selected according to their estimated success likelihood and thier kinematic feasibility, and are executed using off-the-shelf path planning and position control methods.

All intended objectives have been achieved.

1.3 Relation to the state of the art

The implemented system represents the current state of the art. It is intended to serve as a baseline against which PaCMan contributions will be measured.

2 Architecture of the Demonstrator

A high-level description of the developed software for *grasping objects among clutter* is provided in this report along with the most important architectural decisions.

The goal of this software package consists of proving a framework to support the main functionality, which can be divided into several tasks:

1. construct 3D models for the objects to be used in the kitchen scenario and extract discriminative features
2. estimate grasping parameters for each category of objects
3. plan trajectories by taking into account collisions (with the scene or self-collisions)
4. execute trajectories by the robot system (Kuka Lightweight arm and Schunk hand)

For each defined task, a software module was developed as depicted in Fig. 1 below. As input, the system receives a point cloud acquired with an RGB-D sensor. The scene is analysed, objects are recognized and their pose in the scene is estimated. The next step consists of estimating grasping points for each object. Based on several criteria, the most likely object to be grasped is picked and a trajectory is planned for it by taking into account the collisions with the other objects and with the environment. If no path can be found for a given grasp configuration, a new grasp for the same object is chosen. When a path is found, the trajectory is sent to the robot system and is executed, having the effect that the object is grasped and moved outside the scene. The same set of actions is applied for each detected object, until all the objects have been grasped and the scene is empty.

More details about each software module are provided in the following section.

The system was developed in a modular way, the functionality provided by each component is defined in an interface, for which there can exist different implementations. This decision assures a smooth communication between modules and provides the end-user with a clear understanding of the provided functionality, while enabling easy extensions of the current architecture, in terms of functionality or software components.

Furthermore, a decision had to be taken regarding the robotic middleware for providing the interaction between numerous components (software and hardware). The Robot Operating System (ROS) was chosen from the set of alternative systems, due to its attributes such as providing a flexible, robust, and distributed framework for robot software and its possibility to integrate with real-time systems. Being open-source and available for a wide

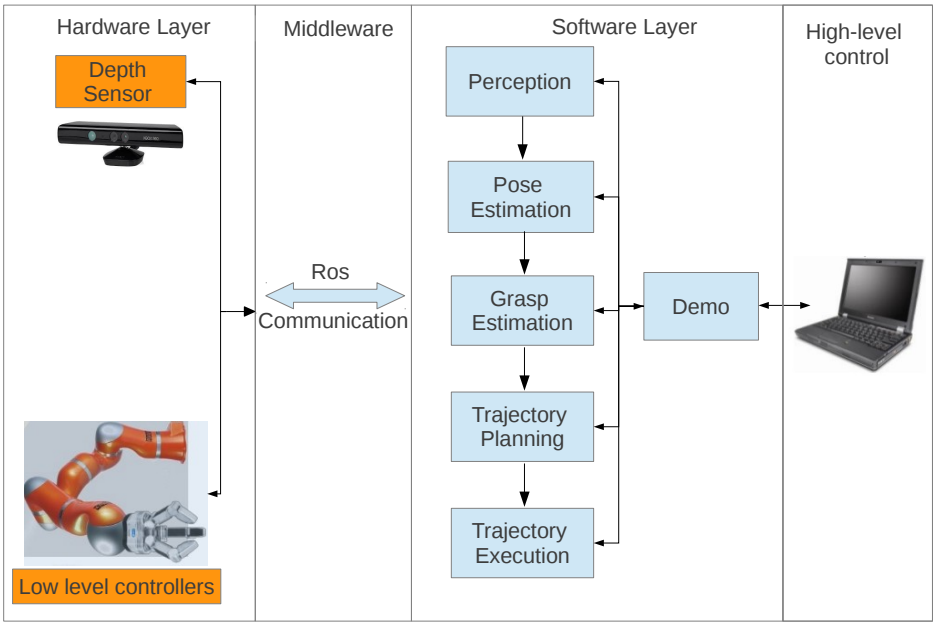

Figure 1: System architecture of the developed robot system.

variety of robotic platforms enables knowledge transfer and facilitates rapid development.

2.1 Software modules

Each software component was developed in an iterative manner, enabling software integration at an earlier stage. Moreover, a ROS wrapper was implemented for each component, facilitating the integration into the ROS framework.

The high-level controlling of the system is handled in the *Demo* component which interacts with all the modules and takes decisions according to a predefined logic. Therefore, each software component is executed only if the required conditions are met.

The *Perception* software component interacts with the RGB-D sensor and provides functions for capturing and saving the input visual data in the required format (e.g. point cloud, depth image, or RGB image).

The input visual data (scene) is processed in the *Pose Estimation* component, which recognizes the objects present into the scene and estimates their pose. The objects considered in the kitchen scenario are stored into a database, along with a set of pre-computed descriptors for each of them (e.g. FPFH, SHOT, SHOT-OMP) (Aldoma et al., 2012). New descriptors are computed for each input scene and a matching algorithm (e.g. consistency grouping) is applied for discovering the most probable hypotheses, from which only the most likely one is chosen. The object pose likelihood with respect to the scene is evaluated using the Nuklei library (Detry et al., 2009).

The *Grasping Estimation* software component has available a set of approach trajectories for each object category. When provided with a new object in a certain pose, it computes all possible grasp configurations and selects the most likely one to succeed (Kopicki et al., 2014).

Given an object and the associated grasp configuration, the *Trajectory Planning* component computes a set of possible trajectories of the robot arm by taking into account the robot kinematics constraints and the environment constraints, both static and dynamic. This component is based on the MoveIt (Mov) software, which incorporates the latest advancements in motion planning and is based on the ROS framework.

Finally, the *Trajectory Execution* software component sends the computed trajectory to the robot system (Kuka arm and Schunk hand), which executes it. In parallel with the robot system, a simulation is also available, which displays the detected objects, the collisions and the trajectory to be executed by the robot. The goal of the simulation component is to provide a fast environment for prototyping and testing different configurations.

2.2 Conclusion

The main goals have been achieved, consisting of providing a modular and flexible robot system for grasping objects among clutter, which can be re-used in subsequent work-packages. Each component can be further improved either by an alternative implementation or by providing new functionality. In particular, the sensing, recognition, pose estimation and grasping facilities will be replaced or further improved by results from the PaCMan project as they become available. The impact of each decision can be directly tested on the robot system, leading at discovering the best configuration.

Each software component has a direct impact on the end result, and solutions were needed at each level for coping with different challenges, such as noisy input data, occlusions, but also transferring grasps between objects in a category, or hardware limitations such the speed of trajectory execution by the Schunk hand.

References

MoveIt. URL `http://moveit.ros.org`.

Robot Operating System (ROS). URL `http://www.ros.org`.

Aitor Aldoma, Federico Tombari, Luigi di Stefano, and Markus Vincze. A Global Hypotheses Verification Method for 3D Object Recognition. In *ECCV (3)*, pages 511–524, 2012. doi: 10.1007/978-3-642-33712-3_37. URL `http://vision.deis.unibo.it/fede/papers/eccv12.pdf`.

Renaud Detry, Nicolas Pugeault, and Justus Piater. A Probabilistic Framework for 3D Visual Object Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1790–1803, 10 2009. doi: 10.1109/TPAMI.2009.64. URL `https://iis.uibk.ac.at/public/papers/Detry-2009-TPAMI.pdf`.

Marek Kopicki, Renaud Detry, Florian Schmidt, Christoph Borst, Rustam Stolkin, and Jeremy L. Wyatt. Learning Dexterous Grasps That Generalise To Novel Objects By Combining Hand And Contact Models. In *IEEE ICRA*, 2014.