Coupling between Perception and Manipulation: Learning to Grasp Objects in Highly Cluttered Environments

Cognitive Robotics Course
https://rugcognitiverobotics.github.io

Assignment Overview

Service robots typically use a perception system to perceive the world. In particular, perception systems provide valuable information that the robot has to consider for interacting with users and environments. To assist humans in various daily tasks, a robot needs to know *how to grasp and manipulate objects in different situations*. For instance, consider a clear table task, where a robot needs to remove all objects from a table and put them into a basket. Such tasks consist of two phases: the first phase is dedicated to the perception of the object, and the second phase is about the planning and execution of the manipulation task. In this assignment, we mainly focus on deep visual object grasping and manipulation.

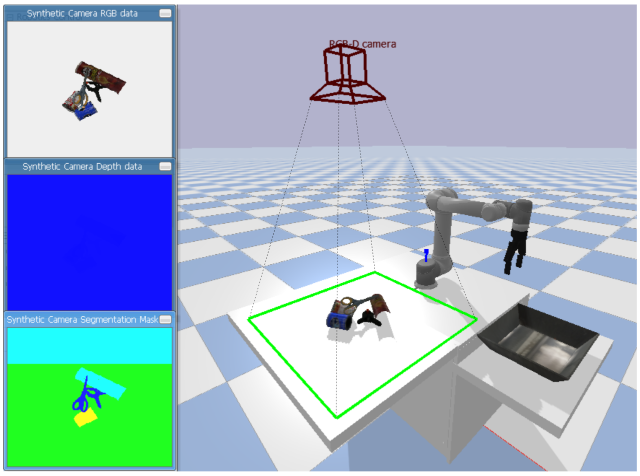

Figure 1: Our experimental setup consists of a table, a basket, a UR5e robotic arm, and objects from the YCB dataset. The green rectangle shows the robot's workspace and the camera indicates the pose of the camera in the environment. Synthesis RGB and depth images, together with a segmentation mask are shown on the left side of the figure.

The main goal of this assignment is to make a coupling between perception and manipulation using eye-to-hand camera coordination. Towards this goal, we have developed a simulation environment in PyBullet, where a Universal Robot (UR5e) with a two-fingered Robotiq 2F-140 gripper perceives the environment through an RGB-D camera. The experimental setup for this assignment is shown in Fig. 1. This setup is very useful to extensively evaluate different object-grasping approaches. After successful completion of this assignment, students will be able to implement and experiment with several methods for object grasping.

Traditional object-grasping approaches explicitly model how to grasp different objects by considering prior knowledge about object shape and pose. It has been proven that it is hard to obtain such prior information for never-seen-before objects in human-centric environments. More recent approaches try to tackle this limitation by formulating object grasping as an *object-agnostic* problem, in which grasp synthesis is detected based on learned visual features without considering prior object-specific information. In this vein, much attention has been given to object-grasping approaches based on Convolutional Neural Networks (CNN). Among deep visual grasping approaches, GR-ConvNet [1] showed state-of-the-art results. In particular, GR-ConvNet receives RGB and Depth images and generates a pixel-wise grasp configuration. As an example of how to use CNN in visual grasping experiments, we have integrated the GR-ConvNet into our setup.

In this assignment, we are pursuing three main goals: (*i*) learning about **at least two** deep visual grasping approaches, (*ii*) evaluating and comparing their performances in three scenarios including, *isolated*, *packed*, and *pile* scenarios (see Fig. 2); (*iii*) investigating the usefulness of formulating object grasping as an object-agnostic problem for general purpose tasks. You can also use this setup to develop your **final project**.

Figure 2: Illustrative examples of three evaluation scenarios: (left) isolated; (center) packed and (right) pile of objects.

In this assignment, we use simulated YCB objects dataset [2]. All objects were inspected to be sure that at least one side of the object fits within the gripper.

Your tasks

This assignment comprises two parts, each worth 50% of your grade. **For the first part**, you need to understand and describe how this system works by reading GR-ConvNet paper, checking the provided code, and examining its performance in *isolated*, *packed*, and *pile* scenarios. We explain the evaluation scenarios and metrics below.

For the second part of this assignment, you need to select another deep visual grasping approach and integrate it into the system. Similar to the previous part, you need to evaluate the model in isolated, packed, and pile scenarios. Finally, you need to analyze and compare the obtained results with the GR-ConvNet model.

It should be noted that it is possible to select a 3D based deep learning approach (e.g., [3, 4, 5, 6]), or depth only based approaches (e.g., [7]), or even data-driven approach (e.g., [8]) instead of RGB-D based approaches. To convert the RGB-D image to point cloud you can use Open3D library. In the case of deep learning approaches, we strongly recommend using pre-trained models.

To evaluate a grasping approach, you need to perform 10 rounds of experiments per scenario and analyze the obtained results. In the case of *pile* and *packed* scenarios, for each experiment, we randomly generate a new scene consisting of five objects (see Fig. 3). For the isolated object scenario, we place a randomly selected object in an arbitrary pose inside the robot's workspace. In all experiments, the robot knows in advance about the pose of the basket object as the placing area, while the robot needs to predict grasp synthesis for the given scene and select the best graspable pose to grasp the target object, pick it up, and put it in the basket. Note that, at the beginning of each experiment, we set the robot to a predefined setting, and randomly place objects on the table.

A particular grasp is recorded as a success if the object is inside the basket at the end of the experiment. You need to report the performance of an approach by measuring $\texttt{success\_rate} = \frac{\text{numberof successful grasps}}{\text{number of attempts}}$.

In the case of pile and packed scenarios, to generate a simulated scene containing five objects, we randomly spawn objects into a box placed on top of the table. We wait for a couple of seconds until all objects become stable, and then remove the box. To generate a packed scenario, we iteratively placed a set of objects next together in the workspace. These procedures are shown in Fig. 3.

For the pile and packed scenarios, in addition to the `success_rate`, you need to report the average percentage of objects removed from the workspace. An experiment is continued until either all objects get removed from the workspace, or four failures occurred consecutively. Note that, the system automatically reports a summary of the obtained results in the "`results`" folder, and the prediction of network is visualized and saved in the "`network_output`" folder.

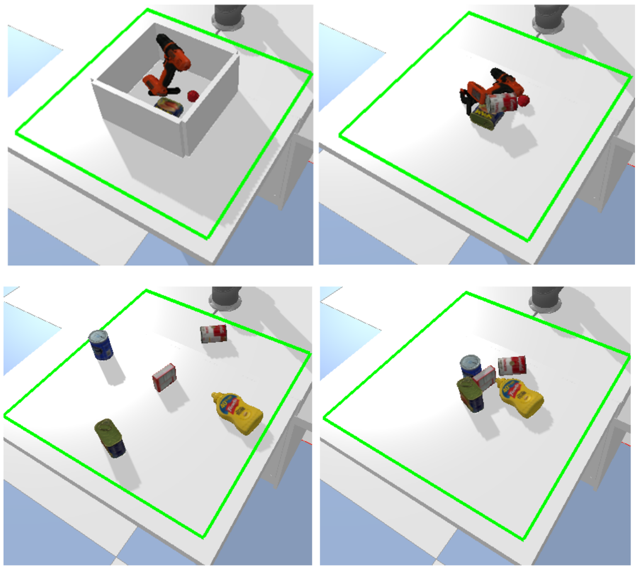

Figure 3: Generating random scenes to make: pile of objects (*top-row*); and packed objects (*bottom-row*).

What we offer for this assignment

- A detail instruction about how to install and run the experiments. **Check the GitHub repository**
- Automatically report the obtained results after each round of experiments.

Policies

- Feel free to collaborate on solving the problem but please write your code individually. **In particular, do not copy code/text from other students**.
- **You are not allowed** to use this code in other projects (even if the code is partially used). If you want to publish your results as a scientific paper or use this framework in other projects, contact Hamidreza Kasaei (`hamidreza.kasaei@rug.nl`) directly and discuss the case explicitly.

Submission

At the end of the practical assignment, a report (i.e., up to four pages + one page appendix in IEEE conference format) has to be delivered. At the end of the report, you have to include an "Authors' Contributions" section, and explain how did you divide the work among the members, and what are the contributions of each author. We expect all authors contribute equally to the assignment. This assignment prepares students to do the final course project. Submit your assignment in as a pdf file named `group_number_prj2.pdf`

⚖ Do not delete your results after submitting the report. We may ask you to send us your `cognitive_robotics_manipulation` packages and the obtained results.

References

[1] Sulabh Kumra, Shirin Joshi, and Ferat Sahin. Antipodal robotic grasping using generative residual convolutional neural network. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9626–9633, 2020.

[2] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols. *IEEE Robotics and Automation Magazine*, 2015.

[3] Michel Breyer, Jen Jen Chung, Lionel Ott, Siegwart Roland, and Nieto Juan. Volumetric grasping network: Real-time 6DOF grasp detection in clutter. In *Conference on Robot Learning*, 2020.

[4] Adithyavairavan Murali, Arsalan Mousavian, Clemens Eppner, Chris Paxton, and Dieter Fox. 6DoF grasping for target-driven object manipulation in clutter. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6232–6238. IEEE, 2020.

[5] Peiyuan Ni, Wenguang Zhang, Xiaoxiao Zhu, and Qixin Cao. PointNet++ grasping: Learning an end-to-end spatial grasp generation algorithm from sparse point clouds. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3619–3625. IEEE, 2020.

[6] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[7] Douglas Morrison, Peter Corke, and Jürgen Leitner. Closing the Loop for Robotic Grasping: A Real-time, Generative Grasp Synthesis Approach. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[8] Marcus Gualtieri, Andreas Ten Pas, Kate Saenko, and Robert Platt. High precision grasp pose detection in dense clutter. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 598–605. IEEE, 2016.