Coursework Description COMP0213 – Object Oriented Programming for Robotics and AI

October 2025

1 Task Context

Grasping and Grasp Planning in Robotics and AI In 2024, the new World Robotics report recorded 4,281,585 units operating in factories worldwide. Although it is hard to give exact numbers to the type of robots comprised in this sum, industrial robots are generally robotic arms capable of operating autonomously for soldering, screwing, and moving through grasping.

Grasping refers to the act of a robotic end-effector (e.g., gripper or hand) securely holding that allows for further manipulation of an object. It's a fundamental capability for physical interaction with the environment—whether it's picking up a cup, assembling components, or assisting users with disabilities.

Grasp planning, on the other hand, involves computational strategies that determine how, where, and when to grasp. It integrates sensory input, object geometry, force models, and robot kinematics to evaluate grasp candidates and select the most stable, efficient one, often using algorithms such as force closure analysis, machine learning, or sampling-based methods.

Importance in Robotics and AI:

1. Physical Autonomy: Enables robots to perform tasks in unstructured environments, such as homes, hospitals, or disaster zones.

2. Human-Robot Interaction (HRI): Safe and adaptive grasping fosters trust and usability, especially in educational or assistive robotics.

3. Learning and Adaptation: Reinforcement learning and imitation techniques allow AI systems to generalize grasps across novel objects.

4. Ethical Impact: Designing inclusive grasp strategies ensures accessibility for diverse user needs, such as children, elderly individuals, or users with mobility impairments.

Therefore, grasp synthesis, namely grasp configuration generation is an important step for an autonomous robot to act in human environments. In this process, two of the main steps involve selecting candidate grasp configurations, i.e. how to position the robotic hand with respect to an object and how to quantify the quality of a grasp configuration (meaning whether the grasping with that configuration would be successful, leading to successful further manipulation). These two steps enable robots to efficiently and safely anticipate how to interact with diverse objects and users, enhancing autonomy, adaptability, and social acceptance in human-robot environments.

Course Work Objective Build a full pipeline of generating grasp candidates/configurations and binary scoring each grasp candidate using ML models and PyBullet simulation.

Intended Leaning Outcomes (ILO)

1. Having hands-on experience in robotic simulation with PyBullet using grasp planning as an application

2. Practical experience with classifiers

3. Data visualization by plotting results

4. OOP concepts being applied into a fundamental robotics problem

2 Breakdown of the project

Generate grasp dataset

In this project, you will be provided with models of robotic grippers and objects to implement a grasp planning framework. An example of a three-fingered gripper is shown in Figure1. The objective is to design and evaluate a grasp planning pipeline consisting of two main components:

1. A grasp configuration sampling strategy.
2. A classifier for evaluating grasp success.

The grasp planning procedure will operate as follows: candidate grasp poses, defined as possible placements of the gripper relative to an object, will be generated using the chosen sampling strategy. Each candidate grasp will then be executed by positioning the gripper at the sampled pose, closing the fingers around the object, and lifting the object to a specified height.

To execute the grasps, in cases where collisions with the object are frequent, it is recommended to first position the gripper at an approach pose using the planned gripper orientation. This approach pose is defined at a specified distance from the intended final grasp pose, ensuring that the gripper fingers can open without colliding with the object. The execution procedure is as follows: first, open the fingers of the gripper; then move the hand to the approach pose; finally, advance the gripper along the approach trajectory to reach the planned grasp pose and perform the grasp.

During lifting test, If the object remains securely in the gripper for several seconds without slippage or dropping, the grasp will be labeled as stable. This procedure will be repeated for each sampled grasp in order to obtain binary success labels. The feature space of the classifier will be six-dimensional, consisting of three translational and three rotational parameters describing the hand pose relative to the object. Alternatively, a seven-dimensional feature space may be employed if orientation is represented using quaternions.

To construct the dataset, you are required to employ two distinct grippers and two distinct objects. The implementation should adhere to the principles of object-oriented programming (OOP) to ensure extensibility with respect to different gripper and object models. You need to design your classes to ensure you do not repeat code to achieve the goal of using two grippers and two objects, e.g. making use of inheritance.

The specific strategy for grasp pose sampling may be chosen at your discretion. One possible approach is to position the object at the center of a notional sphere and to generate grasp candidates by directing the gripper toward the object along radii of this sphere (see Figure 1). Variations in distance and the addition of Gaussian noise to the sampled poses can be incorporated to enhance dataset realism and increase data diversity.

During execution, the gripper pose relative to the object (measured prior to lifting) should be recorded. These poses will serve as input features to the classifier, while the binary grasp outcome (success or failure) will provide the target labels.

The final dataset must be organized as a pandas DataFrame, where the input dimensions (pose features) occupy the initial columns and the final column contains the corresponding ground-truth labels. This dataset will form the basis for training and evaluating the grasp classification model. You will work with a balanced dataset, where the number of positive and negative samples is equal. The dataset should contain at least 120–140 samples.

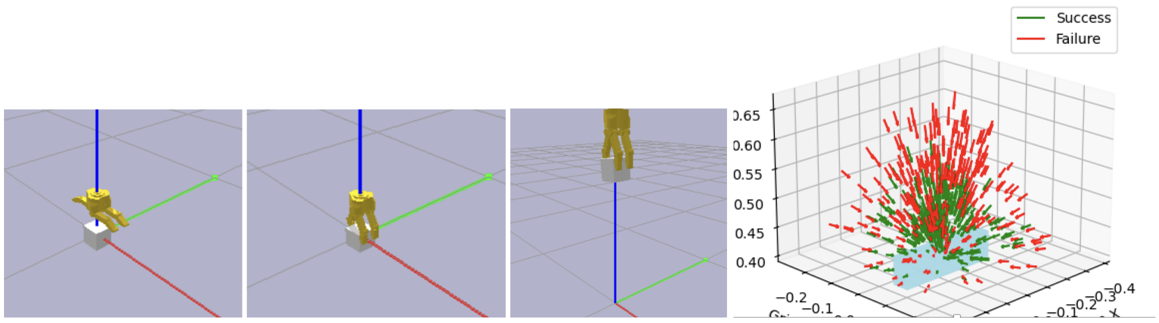

Figure 1: Left: The gripper approaching the cube to grasp (leftmost), grasping the cube (middle) at the planned pose, and lifting and holding the cube lifted (rightmost). This is an example of a successful grasp. Right: Example sampled hand positions and orientations/approach vectors around an object.

Building your classifier based planner

Training Phase

(a) Train a classification model of your choice to distinguish between successful and unsuccessful grasps using the dataset constructed in the previous stage. Employ a classifier available within the scikit-learn library, and use the recorded grasp configurations (position and orientation) as input features. The classifier should be trained to predict binary outcomes corresponding to grasp success or failure. You are encouraged to experiment with different classifiers and hyperparameter settings in order to identify the configuration that yields the highest validation accuracy, at least 60-70%.

Testing Phase

(b) Construct a separate test set of grasp poses following the same sampling strategy used during training. Using the classifier trained in the previous phase, predict the success label of each test grasp and then evaluate the correctness of the prediction by executing the grasp: placing the gripper at the grasp pose, closing the fingers, and lifting the object. Test the classifier on at least ten grasp attempts and report the overall success rate of the predictions.

Term definitions

1. Grasping: The process by which the gripper closes its fingers around the target object to establish a secure hold.

2. Grasp Planning: The determination of an appropriate gripper pose in space—defined by position (x, y, z) and orientation (yaw, pitch, roll) (alternatively quaternions) at which the gripper can approach and successfully grasp the target object.

3. Lifting: The action of raising the grasped object to a specified height while maintaining the same grasp configuration.

4. Successful Grasp: A grasp is considered successful if the object remains securely held by the gripper for more than three seconds after lifting.

Requirements

You will receive the code and report submission guidelines. Your task is to implement the project using Object-Oriented Programming (OOP) principles, such as inheritance, abstract classes, and UML class diagrams. Additionally, you should incorporate other libraries introduced in this course, such as NumPy, Pandas, and others, where applicable.

References:

1. PyBullet website: `https://pybullet.org/`

2. PyBullet Quick start pdf file included in Moodle.