Robotic Manipulation

Exercise Practicalities

Jens Lundell Vladimir Petrik

Exercise session overview

- Exercise sessions every Wednesday 08:15-10:00 in TU3 and Thursday 10:15-12:00 in AS3.
- Voluntary presence.
- Bring your own laptops to these sessions.
- Teacher assistants (TAs) present to help you.

Requirements

- A Linux system (we recommend and support **Ubuntu 16.04**)
- ROS Kinetic (we support this version) `http://wiki.ros.org/kinetic/Installation/Ubuntu`
- catkin tools `https://catkin-tools.readthedocs.io/en/latest/installing.html`
- MoveIt! for Kinetic `https://moveit.ros.org/install/`
- The MuJoCo simulator *mujoco200 linux* `https://www.roboti.us`.
- The MuJoCo license found in MyCourses under "For Aalto users". **IMPORTANT**: The license is for **personal use only** and cannot be redistributed!
- You can download an already configured virtual machine including the above programs by following these steps `https://version.aalto.fi/gitlab/robotic_manipulation/installation`.

Communication

- The preferred means of communication is the course slack channel (separate email for registration has been sent to all course participants) and not email.
- When you sign up to the slack workspace use your **aalto username** as the **Nick name** as we will link this to your gitlab repository.
- If you have specific problems with your code do not send it over email or slack. Instead, tell us that you have some problems and push your latest commits to the gitlab repository and we will pull it from there and start investigating.
- Usually more than one student have similar problems and thus we advocate asking questions in the exercise slack channels to enable students to help each other out.
- TAs and the lecturer will, per default, not answer emails or slack messages during the weekends.

Exercises

- In total seven problems:

Exercises

- In total seven problems:
  1. Introduction to ROS.

Exercises

- In total seven problems:
  1. Introduction to ROS.
  2. Simple pick and place with MoveIt

Exercises

- In total seven problems:
  1. Introduction to ROS.
  2. Simple pick and place with MoveIt
  3. Planning algorithms benchmark in MoveIt

Exercises

- In total seven problems:
  1. Introduction to ROS.
  2. Simple pick and place with MoveIt
  3. Planning algorithms benchmark in MoveIt
  4. Visual perception

Exercises

- In total seven problems:
  1. Introduction to ROS.
  2. Simple pick and place with MoveIt
  3. Planning algorithms benchmark in MoveIt
  4. Visual perception
  5. Pushing an object

Exercises

- In total seven problems:
  1. Introduction to ROS.
  2. Simple pick and place with MoveIt
  3. Planning algorithms benchmark in MoveIt
  4. Visual perception
  5. Pushing an object
  6. Planning a stable grasp

Exercises

- In total seven problems:
  1. Introduction to ROS.
  2. Simple pick and place with MoveIt
  3. Planning algorithms benchmark in MoveIt
  4. Visual perception
  5. Pushing an object
  6. Planning a stable grasp
  7. Move object in task space with two robots

Tentative exercise schedule

The exercises are introduced on the following exercise sessions

- 9th of January intro to exercise 1
- 16th of January intro to exercise 2
- 23rd of January intro to exercise 3
- 30th of January intro to exercise 4
- 6th of February tutorial on ROS control.
- 13th of February intro to exercise 5
- 27th of February intro to exercise 6
- 13th of March intro to exercise 7

Exercise Deadlines are stated separately in each assignment PDF. However, below is a tentative schedule

- 22nd of January deadline for exercise 1.
- 29th of January deadline for exercise 2.
- 5th of January deadline for exercise 3.
- 26th of February deadline for exercise 4.
- 12th of March deadline for exercise 5.
- 26th of March deadline for exercise 6.
- 9th of April deadline for exercise 7.

Submissions

- Each solution include at least source code written in C++ that solves the problem and for most of the exercises you also need to submit a report written in English.
- The report (saved as PDF) should answer the questions posed in the assignment which can, for example, be the following
  1. Which planner is fastest and why? Plot the running times and answer the questions by comparing the algorithms.

Submissions

- Each solution include at least source code written in C++ that solves the problem and for most of the exercises you also need to submit a report written in English.
- The report (saved as PDF) should answer the questions posed in the assignment which can, for example, be the following
  1. Which planner is fastest and why? Plot the running times and answer the questions by comparing the algorithms.
- If the exercise requires you to submit both a report and code you need to submit both. Otherwise, your solution will be rejected and you will be awarded with 0 points.

Submissions

- Each solution include at least source code written in C++ that solves the problem and for most of the exercises you also need to submit a report written in English.
- The report (saved as PDF) should answer the questions posed in the assignment which can, for example, be the following
  1. Which planner is fastest and why? Plot the running times and answer the questions by comparing the algorithms.
- If the exercise requires you to submit both a report and code you need to submit both. Otherwise, your solution will be rejected and you will be awarded with 0 points.
- Each student forks the exercise into their own gitlab group, solves the exercise there, and finally upload everything to the respective repository before deadline

Grading

- If the exercise requires both code and a report the report accounts for 50% and the code for 50% of all points awarded for that exercise. Otherwise, one or the other accounts for 100% of the points.
- The code is graded based on correctness (0-100%).
- The report is graded based on:
  - Correctness (0-100%),
  - How well it is written (satisfactory, good, excellent).
  - We will grade both language and structure of the report. For example, the report should be easy to read and coherent, and you need to refer to all figures, tables etc. Think of every single report as a part of your future MSc. thesis.
- TAs will push, to your gitlab repository, the grade and feedback for the given exercise.

Exercise rules

- Exercises are handed in and done **individually**
- You are allowed to discuss the problems but not share solutions.
- No copying of exercises (neither code nor report). If we notice plagiarism it is reported and consequences follow.
- No late submissions are accepted.
- You are allowed to miss one exercise; although this will probably affect your grade negatively. Missing more than this requires a valid excuse such as a doctor's statement of you being ill.