CSCI 445 Lab 1 — Introduction

Prof. Ayanian, University of Southern California Spring 2017

1 Prerequisites

1. Be familiar with the terms robotics and robots.
2. Bring your laptop.

2 Lab Policies

- The lab is 40 % of the overall grade (10 % final project and 30 % other labs and programming assignments).
- There is a short quiz (5 min, closed book) at the beginning of most labs. The quiz covers material in the prerequisite section of each lab. It is pass/fail; if a quiz was failed, 20% of the possible points for that lab section will be deducted. A quiz is also considered failed if a student arrives more than 10 minutes after the regular start of the lab.
- Teams of three students each will be assigned by the TAs every lab (excluding the first lab).
- Lab attendance is mandatory – a missed lab will mean 0 points.
- Most labs have two parts: simulation and practice. You are only allowed to practice on the robot after your simulation is working. Show the TA your working simulation before experimenting with the robot.
- No makeup labs will be available except for reasons officially excused by the university (see the TA in these cases immediately).
- Please keep the lab clean. Before leaving the lab, please ask your TA to check your workspace.
- Please bring your laptop to all of your lab sessions.
- Program assignments must be solved individually. Students are allowed to discuss the assignments with each other, but they are not allowed to look at others' solutions from this or previous years.

3 Robotics Platform

3.1 iRobot Create2

You will be using iRobot Create2 robots during the lab. Those are essentially refurbished Roomba vacuum cleaners (without the cleaning unit), designed as "an affordable STEM resource for educators, students and developers." The robots firmware is closed source, however a serial port is available together with an API[1] describing on how to control the different actuators and how to read sensors.
The following actuators are supported:

- 2 wheels (pulse-width-modulation or velocity control)
- Motors (main brush, vacuum, side brush)
- LEDs (4 binary status LED, 1 ring with variable color & intensity)
- 4-digit 7-segment display
- Speaker (playing "songs")

The following sensors are available:

- Wheel encoders
- 4 Cliff sensors (essentially downward facing IR-sensors)

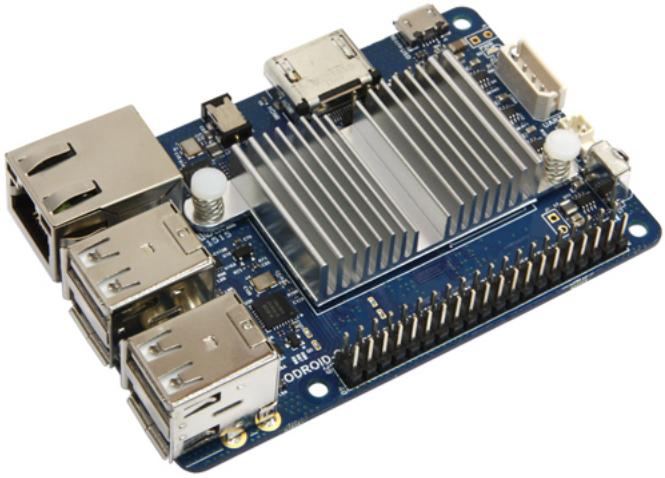
[1] http://www.irobotweb.com/~/media/MainSite/PDFs/About/STEM/Create/iRobot_Roomba_600_Open_Interface_Spec.pdf?la=en

- 3 Infrared receivers (left, right, omni-directional)
- 8 Buttons
- 6 IR distance sensors
- 2 wheel-drop sensors
- 2 bump sensors
- Temperature
- Power-related sensors (voltage, currents, battery capacity)

Furthermore, there is a dock which sends out infrared beams such that the Create2 can find it autonomously for charging.

3.2 Odroid C1+

To be able to run software without the need of a wire to your laptop, we use a small embedded computer, running Ubuntu 14.04 LTS. The Odroid C1+[2] is comparable to a smartphone without a screen. It runs a ARM Cortex A5 1.5GHz quad core CPU, has 1 GB RAM, boots from microSD, and has 4 USB ports. Additional sensors and actuators can be used using general purpose IO (GPIO). Like your phone, it can connect via WiFi to your laptop and you will be able to program the robot this way.



[2] http://odroid.com/dokuwiki/doku.php?id=en:odroid-c1

4 Simulation Platform

We will try to do everything in simulation first, before implementing it on a real robot. This has two reasons: first, it simplifies debugging and reduces the time for a working prototype. Second, it avoids safety-related issues (which is in particular relevant for larger robots).

We will be using V-REP[3], a cross-platform robotics simulator which is both used in academia and industry. It works on Windows, Linux, and Mac OS X and can be extended using several languages including C++, Python, Java, and Matlab.

While V-REP comes with many robot models included, it does not yet support the iRobot Create2. We will provide you with a model which does not support all the features of the real hardware, but will suffice for our lab purposes. In particular, we only support motors and wheel encoders at the moment.

5 Task 1: Install V-REP

1. Go to http://www.coppeliarobotics.com/downloads.html and download V-REP Pro Edu V3.3.2 (rev3) for your platform.

5.1 Linux

On a modern system you will most likely need the 64-bit version of V-REP. You can download and extract the `tar.gz` file in any folder. To launch V-REP, open a terminal, navigate to the extracted V-REP folder, and execute

```
[laptop]$ ./vrep.sh
```

5.2 Windows

Follow the installer to install V-REP. There will be a shortcut on your desktop to run it after the installation is complete.

5.3 Mac OS X

Launch the `vrep` executable after downloading the archive. You might need to change your security settings to allow applications from an untrusted source.

6 Task 2: Install Python 3

6.1 Linux (Ubuntu)

On a command prompt use:

```
[laptop]$ sudo apt-get install python3 python3-numpy python3-matplotlib
```

6.2 Windows

Download the python3 Anaconda distribution 32-bit from https://www.continuum.io/downloads — We tested Anaconda 4.2.0 with python 3.5. Please make sure you install the 32-bit version for Windows as there are some issues with the 64-bit version installation for working with V-REP. Please note that you will be able to run python by using `python` rather than `python3` as on Mac OS X and Ubuntu.

[3] http://www.coppeliarobotics.com/

6.3 Mac OS X

Download Anaconda (Python 3.5) from https://www.continuum.io/downloads and follow the installer. You can verify your installation by running `python3 --version` in a Terminal.

7 Task 3: Test the Simulation

1. Download `lab1.zip` from Piazza.
2. Extract the archive.
3. In V-REP, open the included `Lab1.ttt` scene file by using the "File/Open scene..." menu. You should be able to see a virtual iRobot Create2.
4. In a terminal with the extracted zip-file as your working directory:

   ```
   [laptop]$ python3 run.py --sim lab1
   ```

   (On Windows use `python` rather than `python3`.)
5. In V-REP, you should see your robot move for 10 seconds and then stop.

8 Task 4: Test the Robot

To execute the code on your robot, you will need to copy the files to the robot and then, using a remote terminal, execute the script without the `--sim` argument.

1. Connect to the following WiFi: `CSCI445` using the password `robotsAreFun`
2. The hostname is written on your robot "create<X>" and the username and password are both `csci445`.
3. For SFTP (using FileZilla, see below), use `sftp://create<X>` as host, where `<X>` is the number of your robot.

8.1 Linux (Ubuntu)

1. To copy the files over you can use FileZilla (a better way is described below):

   ```
   [laptop]$ sudo apt-get install filezilla
   ```

   You should now see your local file system on the left side, and the robots file system on the right side. You can copy files by dragging them over.
2. To open a remote terminal, you can use `ssh`:

   ```
   [laptop]$ ssh csci445@create<X>
   ```

   When prompted, type the password as above and you should see a terminal for your robot.
3. Navigate to the folder and execute:

   ```
   [create]$ python3 run.py lab1
   ```

   Your robot should move for 10 seconds and then stop.

Tip: To simplify copying etc. you can mount the odroids file system to a folder:

```
[laptop]$ mkdir ~/odroid
[laptop]$ sshfs odroid@create<X>: ~/odroid
```

Now the files on your robot will appear in the folder `~/odroid` and you can edit them like local files.

8.2 Windows

1. To copy the files you can use FileZilla Client[4](a better way is described below): You should now see your local file system on the left side, and the robot's file system on the right side. You can copy files by dragging them over.

2. To open a remote terminal, you can use PuTTY[5]: use `create<X>` as your host name and type username and password on request.

3. Navigate to the folder and execute:

   ```
   [create]$ python3 run.py lab1
   ```

   Your robot should move for 10 seconds and then stop.

Tip: To simplify copying etc. you can mount the odroids file system to a drive: Follow the instructions at https://igikorn.com/sshfs-windows-10/. Now the files on your robot will appear in the drive you specified and you can edit them like local files.

8.3 Mac OS X

1. To copy the files you can use FileZilla Client[6](a better way is described below): You should now see your local file system on the left side, and the robots file system on the right side. You can copy files by dragging them over.

2. To open a remote terminal, you can use `ssh`:

   ```
   [laptop]$ ssh csci445@create<X>
   ```

   When prompted, type the password as above and you should see a terminal for your robot.

3. Navigate to the folder and execute:

   ```
   [create]$ python3 run.py lab1
   ```

   Your robot should move for 10 seconds and then stop.

Tip: To simplify copying etc. you can mount the odroids file system to a folder: Follow the instructions at http://www.macissues.com/2014/10/13/how-to-mount-a-remote-system-as-a-drive-using-ssh-in-os-x/. Now the files on your robot will appear in the folder you specified and you can edit them like local files.

9 Task 5: Implement Motion Primitives

Test your understanding by implementing simple primitives: move forward, turn left (in place), turn right (in place), move forward while turning, move backwards, and stop. You will only need to edit `lab1.py`, but feel free to browse the other files for documentation. You should only use the `drive_direct` function.

Test your script in simulation first before executing it on the actual robot.

[4] https://filezilla-project.org/
[5] http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
[6] https://filezilla-project.org/

10 (Optional) Task 6: Setup IDE

You can use a plain text editor throughout the class. However, if you would like to use an IDE, we recommend PyCharm[7]. The free community edition suffices fully, however you can get the professional edition for free as well, see https://www.jetbrains.com/student/ for details.

11 Bored?

The Create2 can play music and has a small display. Feel free to play around with those features. Furthermore, try the docking capability (and guess how it might work).

[7] https://www.jetbrains.com/pycharm/