Final Project: Pick and Place Challenge

Register your team by: 11/20/2024 @ 11:59pm

In-Class Presentations: 12/03/2024 and 12/05/2024

Testing in Auditorium: 12/10/2024

Final Competition: 12/11/2024

Report Due: 12/09/2024 *(No-Penalty Extension until 12/19/2024) @ 11:59pm*

MEAM5200 culminates in a final project that leverages all the concepts and skills you've learned throughout the semester! You will perform the practical task of picking up objects in your environment and placing them where they need to go. Your task is to develop a robust and reliable solution which can acquire blocks (either stationary or in motion) and stack them on a goal platform, in as tall a tower as possible. Your deliverables are:

1. An **In-Class Presentation** which describes the approach you took to solving this challenge, any outstanding shortcomings of your approach, and some preliminary results of your testing.
2. In the **MEAM5200 Final Demo/Competition**, your robot will go head-to-head with the other teams' robots in a shared pick-and-place environment using the real Panda arms. This demonstration and competition serves as an extended opportunity to test the performance and reliability of your solution as well as its ability to adapt to changes in the environment. The top-scoring teams will also receive a small amount of extra credit!
3. A **Final Report** (including both a group and individual component) which presents in detail the project goals, the approach you took to solving this challenge, and your evaluation and analysis of its successes and shortcomings.
4. A copy of your **code**. The code will not be graded against an autograder, but will be reviewed to check that it generally matches the strategy outlined in your report. For readability, please make sure to comment your code.

All students must work in a team of 4 students (except for a small number of teams due to class size). These team size requirements are strict. Please check out Ed Discussions to help with team formation.

You will submit all materials through Gradescope. The report and code are due 12/09/2024 due to University policy, but **a no-penalty extension will automatically be granted to all students until 12/19/2024 at 11:59pm. After this time, late submissions will be accepted per the standard late submission policy, and will be penalized by** 25% **for each partial or full day they are submitted past the time of the no-penalty extension.** After the late deadline, no further assignments may be submitted; post a private message on Ed Discussion *before the (no-penalty extension) deadline i.e 12/19/2024* to request an extension if you need one due to a special situation such as illness. There will be no extensions on the final project report deadline beyond 12/19/2024.

Most testing for this project will be done in a specially crafted Gazebo environment, but you will have some opportunities to test your solution on the real robots before the Final Competition (subject to the time and availability constraints). You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools. **If you collaborate outside of your group members or make use of outside sources, you are expected to cite them in your report.** When you get stuck, post a question on Ed Discussion or go to office hours!

1 Rules and Specifications: Pick and Place Challenge

Your task is to maximize the score your robot can reliably achieve during a round of the game!

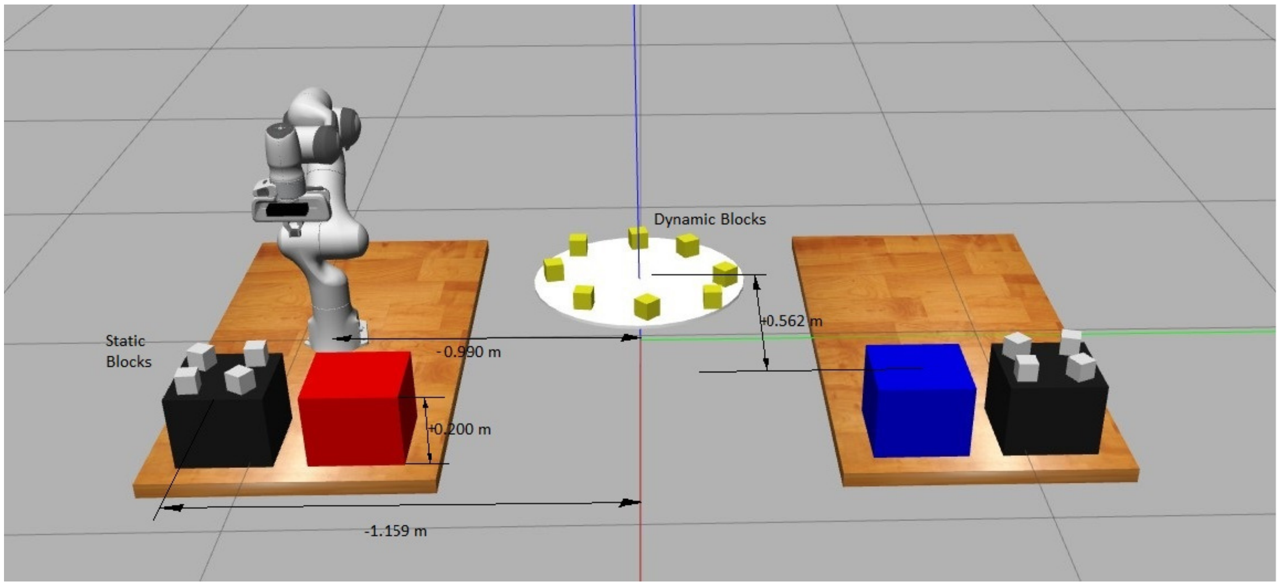


Figure 1: Final project environment

1.1 Ground Rules

1. **Students are required to work in teams of four.** You are welcome to form your own team and can also post on Ed Discussion to look for other teammates. A small number of 3-person teams will be necessary, but must obtain permission to do so in advance.

2. Each match will consist of two teams (designated Red Team and Blue Team) competing head to head in a shared environment, shown in Fig. 1.

   (a) The base of each robot is located `.990 m` from the origin of the world frame, on the world's `y` axis. The axes of the base frame of each team's robot is parallel to the world frame's axes. The robots are positioned such that their workspaces just barely do not overlap.

   (b) **No point on the robot (except the base) is ever permitted to go below an invisible horizontal planar barrier with an altitude of `.200m` in the world frame.** This will be enforced in software, in the real-world environment.

   The arrangement of scoreable objects will be randomized before the start of each match, and your opponent will also have access to the shared dynamic blocks.

3. **Matches will last for a maximum of 5 minutes.** Teams will activate their robot at the start whistle, and their robots will be halted with the Software Stop at the final whistle. All qualification rounds will be of 3 minutes duration. The knock-out matches will have a 2 minutes extra time if the score after first 3 minutes is more than 4000 points. (Refer section 1.3).

4. **Teams must only interact with the robots through a single `ArmController` object for their robot.** Teams are forbidden from directly sending or receiving data over any ROS topics or services, in order to create a level playing field for all.

5. **Teams must operate the physical robot in a safe manner *at all times*.** The judgment of whether a robot is operating safely (i.e. without erratic motions that could harm the robot, people, or the environment) is solely up to the teaching team. **If at any point during testing or demo/competition, the robot is deemed to be operating in an unsafe manner, the teaching team will not hesitate to Software Stop the robot, *without restarting the match*.** Accordingly, teams should make sure their robot operates in a way that is deliberate and can be deemed perceptibly safe to a reasonable observer.

1.2 Scoring Points

1. Scoreable objects are 50mm$\times$50mm$\times$50mm wooden blocks, which come in two varieties:

   (a) **Static Blocks (white):** These objects are randomly dispersed on a stationary platform with its surface .200m above the world's x−y plane, with its center at (.562m, $\pm$1.159m) in the world frame.

   (b) **Dynamic Blocks (yellow):** These objects are randomly dispersed on a rotating turntable with radius .305m, centered at the world frame origin with its surface .200m above the world's x−y plane.

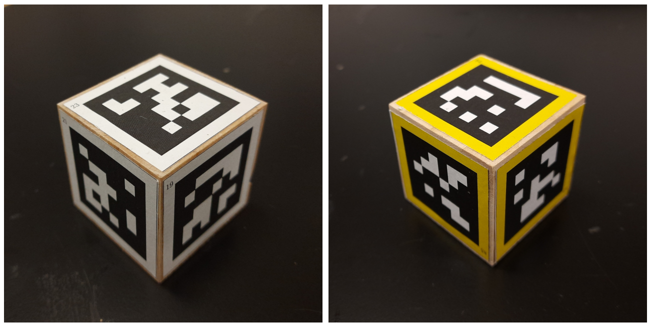


Figure 2: Static scoreable block (left) and dynamic scoreable block (right) covered in AprilTags

2. During the match, teams will manipulate objects to move them onto the goal platform matching their team's color (Red/Blue), located .731m from the world frame x-axis and .562m from the world frame y-axis. This goal platform seen from above is .250m $\times$ .250m, with its surface .200m above the world's x−y plane.

3. Each individual scoreable object supported by a team's goal platform will receive points according to the formula Points = Value $\times$ Altitude, where:

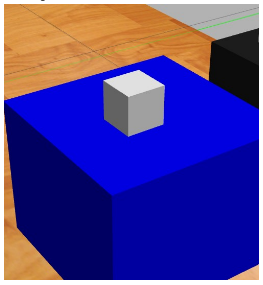
   (a) Altitude is the distance from the center of the object to the surface of the goal platform in millimeters

   (b) Value is 10 for Static Blocks and 20 for Dynamic Blocks.

   The team's final score is the sum of the point value of each scoreable object.

4. **All points will be scored after the match concludes, so that only the final state of the blocks determines the awarded points.** Below are some example final configurations of blocks and their corresponding point values.

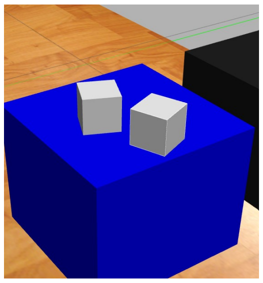
Example 1
Single static block



individual blocks:
10 $\times$ 25 = 250

Total Points: 250

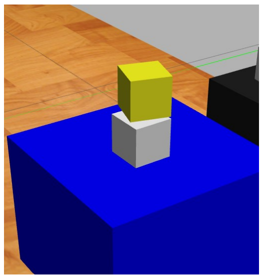
Example 2
Two static blocks side by side



individual blocks:
10 $\times$ 25 = 250
10 $\times$ 25 = 250

Total Points: 500

Example 3
Dynamic block over static block



individual blocks:
10 $\times$ 25 = 250
20 $\times$ (25 + 50) = 1500

Total Points: 1750

1.3 Tournament

1. The exact tournament schedule will be announced via Ed Discussion. The tournament will consist of two phases:

   (a) **Qualification Rounds** *(20 matches, each team plays 2 times)*

      i. The teams are divided randomly into 4 divisions of 5 teams. Each team will play against two other randomly assigned teams in their division.

      ii. The teams will be ranked, determined successively by:
         - Number of wins
         - Total points scored by the team
         - Total points scored by the team's opponents in each round

   (b) **Knock-out** *(12 matches total)*:

      i. The top 2 teams of each division in qualifying rounds will be promoted to the knock-out rounds, where teams will go head to head with the winner of each match advancing to the next round.

      ii. The last *Panda* standing is the champion!

2. In the event of a tie, the winner of a match will be determined successively by:
   - `Altitude` of the highest scored block
   - Number of Dynamic blocks scored
   - Stacking speed - the first team to finish stacking

1.4 Extra Fun! (Optional)

To have some extra fun in the competition for this semester, we have added more elements to the setup. You will now have access to an extra camera in the middle of the two robots, looking down at the turntable as shown in the Figure 3. The camera is located 0.8m away from the center of the turntable in the x-direction and 0.25m above the world origin. This camera will provide you with streams of RGB images and depth images. However, they will all be raw images (meaning they don't come with April tag detection). The purpose of these extra sensing is to allow you to use your creativity for more possibilities in this competition. You could try using different libraries (e.g., OpenCV) for image processing to help with your dynamic block grasping task!

More interestingly, for more advanced teams, you could even try training lightweight neural network models using the readings from this extra camera. To facilitate that, we have prepared Nvidia Jetson Nano, a mini-computer that can deploy computer vision and deep learning software. Here is a community list for projects created with the Jetson Nano: `https://developer.nvidia.com/embedded/community/jetson-projects`. If you are interested in using the Jetson Nano for your project, please create an Ed post to explain your plan during the proposal period.

We hope that this extra element can help you achieve more in this competition. You are welcome to use codes from your previous projects.

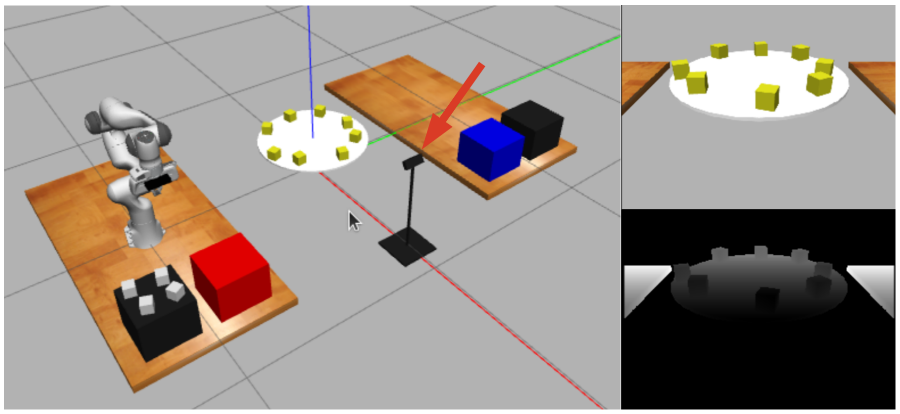


Figure 3: The extra camera

2 Register Your Team (due 11/20/2024 @ 11:59pm)

In order to reserve times on the robot hardware, you will need to register your team. Create your group by adding full names of all your team members under the same team id block `https://docs.google.com/spreadsheets/d/16cNQGd2AztLK3oOtyJZlyC5ROvb8HZMlxvKxa_0B1Wo/edit?usp=sharing`. Memorize your team id after signing up because it will later be used to reserve robot slots.

2.1 Reserving Robot Slots

By the end of this week (11/17/2024), the link to an excel sheet will be posted on Ed for you to reserve time in the robot lab. You should schedule timeslots as a team. Each team is restricted to 3 30-minutes time slots per week. Please fill out your team number in the excel timeslots you would like to reserve for the upcoming weeks.

`https://forms.gle/KM6vv72Y4ikj8PQ16`

3 Simulation Development and In-Lab Testing

We will adopt our usual workflow, where you develop your solution in the simulated environment and later test it on the real robot. The Final Competition will take place with two real robots.

3.1 Simulation Setup

As usual, update your own private fork of the `meam520_labs` repo for the Final Project:

```
$ cd /home/$USER/meam520_ws/src/meam520_labs/$
$ git pull upstream main
```

You will need to install two new ROS packages:

```
sudo apt install ros-noetic-velocity-controllers
sudo apt install ros-noetic-ros-control
```

You will also need to install/upgrade libraries:

```
pip3 install numba
sudo pip install --upgrade scipy
```

Build and source the workspace using:

```
$ cd /home/$USER/meam520_ws/
$ catkin_make_isolated
$ source devel_isolated/setup.bash
```

To launch the simulated testing environment, with randomized block arrangement, run **either** of the below commands to test as that team.

```
$ roslaunch meam520_labs final.launch team:=red
$ roslaunch meam520_labs final.launch team:=blue
```

The script with which you will control the robot is located at

```
/home/$USER/meam520_ws/src/meam520_labs/labs/final/final.py
```

and contains some starter code which sets up the arm controller, automatically determines the team you are playing, and waits for you to press enter to begin. **All your computations should begin after pressing ENTER. Any teams that violate this rule will be directly eliminated.**

3.2 Controlling the Robot

Important: To avoid self-collision and collision with tables, in this project the **ONLY** commands you are allowed to use to control the robot are the following "safe" methods of the `ArmController` class, which function similarly to their non-"safe" counterparts but with additional checks against environmental collision, etc.

- `safe_move_to_position(q)`

  Under the hood, this command is calling a trajectory planner which moves the arm from one position to another position along a linear path through configuration space. For usage examples, you can consult Labs 0, 1, 3, and 4.

- `safe_set_joint_position_velocities(q, q_d)`

  This command sets the joints of this limb using specified positions and velocities using impedance control.

- `exec_gripper_cmd(pos, force)`

  This command directs the gripper to close to a width of `pos` meters, and apply up to `force` newtons to do so. (An example force would be 50 N).

3.3 Detecting Blocks

In order to manipulate the blocks in the environment, you must first know where they are located! To make this possible, each block in the real-world environment will have unique AprilTags (visual fiducials) on each face of the block.

The cameras that detect the position of these tags have been mounted on the end-effectors of each robot. The teaching staff have calibrated the cameras, and are using OpenCV to estimate the position and orientation of each static and dynamic block relative to the camera mounted at the end effector. The pose (position & orientation) of camera in the end-effector frame can be obtained by the function `get_H_ee_camera(self)`. Only blocks that are in the field of view of the camera will be reported. **The camera in simulation will not be exactly the same as the Hardware. Their field of view might differ**.

You will be provided with the center of each block (in camera frame) along with the information if the block is dynamic or static.

It's vital to keep in mind that while the simulation data is "ground truth", meaning it has virtually no noise, the real data generated by the vision system in performing AprilTag detection can be noisy. This is particularly true for the orientation. We encourage you to consider what priors (structural information about the environment) you can leverage to improve your estimates. For a block sitting on the platform or turntable, ask yourself questions like:

- What orientations are permissible?
- What information does the height of the table give me?
- How does the block pose evolve (or not) over time?

To read detections from the (real or simulated) vision system, use the following:

```
from core.interfaces import ObjectDetector
detector = ObjectDetector()
for (block_name, pose) in detector.get_detections():
      ... # do something
```

This method returns all AprilTags found in the most recent frame. This example is also provided in `final.py`. Using the same `ObjectDetector`, we can retrieve the RGB image and the depth image through

```
mid_depth = detector.get_mid_depth()
mid_rgb = detector.get_mid_rgb()
```

3.4 Handling Blocks

We are providing the block locations in both simulation and on hardware. So the transition between simulation to the real world should be comparable, but be warned, there are many things that can go wrong on the robot that do not go wrong in simulation! You should come early to lab to incrementally test your solutions!

One question that you should be asking yourself is: How will you know you grabbed a block if you don't have pose information (as the camera cannot see the block in gripper)? You may address this question using the `get_gripper_state` method of the `ArmController`, which returns a position and force measurement for each finger of the gripper.

In past competitions, we have seen several groups successfully pick up dynamic blocks in hardware and it was exciting every time. Do your best in hardware, and wow us with your closed-loop dynamic block grasping in simulation and your report.

3.5 Timing

Because your simulation does not run with a Real-Time Factor of exactly 1, it's important that any time-based reasoning your code performs uses the current ROS time, not the Wall time (i.e. the time that a clock on the wall of the lab would read). Check out the function below that we provide for this purpose.

```
from core.utils import time_in_seconds
...
t = time_in_seconds() # matches ROS time
```

That way, in simulation `t` will match the Gazebo time, while on hardware it will match the robot time i.e. realtime.

3.6 Transferring Your Code

We will use Gradescope to allow each team to submit their code. **All teams should submit their code by 12/10/2024 11:59 PM.** TAs will use this code for the qualification stage. After the qualification stage, the teams will have a short break in which they can update their code (if needed) for the knock-out stage. TAs will pull the code again before starting the knock-out stage.

3.7 Testing on Hardware

We anticipate the bulk of your development and preliminary testing will happen in simulation. Once you are relatively confident your code works in simulation, you and your team should reserve time in lab to do additional testing and practice your strategies on the robots.

Note: while in lab you saw things work in simulation and then work the same in the real world. During this challenge, there are additional sources of noise which may impact the performance of your method on the hardware (e.g. noise in the perception). So it will be important to practice on the hardware.

Every member of your team must have completed Robot Lab 0 and Robot Lab 1 to test on the robot, and at least 2 members of your team must be present to operate the robot. We will expect that you and your group mates respect all of the safety instructions covered. Each group will get one warning if safety rules are broken. The second violation will result in you and your team being asked to leave no matter where you are in the reservation time. If you have questions about how to safely operate the robots, please ask the TA.

We will ask that you use the same pipeline on the robots that you used in the robot labs: **Test in simulation, and then test on the robot.** We will ask that you show the TA your code working as expected in simulation on the lab computer. Once you have a 'go-ahead' from the TA, you are good to run tests on the robot. It will be important for the team member manning the software stop to be vigilant and stop at any time they are concerned something may not be working on the robot as they expect.

You and your team will be allowed **three 30 minute reservations a week**, which should be reserved **by one team member** through the Google form. This lab time is yours, and if you want to discuss your strategy with a TA or test on the robot, you are welcome to. We will be strict about following reservations and kindly request your cooperation in promptly wrapping up before the end of your slot. To optimize your time in the lab, we strongly recommend preparing a testing plan and verifying your code in simulation *before* your lab session begins.
Open a new terminal, launch the vision pipeline for the camera first,

```
$ roslaunch meam520_labs vision_pipeline.launch camera_number:={ask the TA}
```

Similar to your previous robot lab, in a new terminal, you will first run,

```
$ ./franka.sh master
```

Then you will launch the robot interface for the final project,

```
$ roslaunch franka_interface interface_final.launch team:={blur or red}
```

After that, you should be ready to run your code in the `final.py`.

4 In-Class Presentation (12/03/2024 & 12/05/2024)

The goal of the in-class presentation is to give each team the opportunity to describe and share their insights with the rest of the class. Similar to the proposal, this assignment should be completed with your team, not individually. Your in-class presentation should cover the following topics:

- **Methodology:** What are the component technologies you plan to employ for the final project? How are the component technologies integrated? What are the design choices that went into these decisions? What is the architecture of your proposed planning and control framework?
- **Experimental Evaluation Plan:** What is your test and evaluation plan (this should consist of a combination of simulation tests as well as physical tests with the robot)? What is the setup and the procedures you are using to test and evaluate the various subcomponents and the integrated system? How are these tests designed? What are the metrics you are using to evaluate success?
- **Results and Discussions:** Report any preliminary results and analysis you have done and how these preliminary results will impact your work forward.

Note that it is not expected for your implementation to be complete by the time you give your presentation. Instead, you should plan to report on what you have completed so far and any challenges that you have run into that you think might be interesting to the rest of the class.

The project presentation is worth 15 points. Presentations will be scored by your peers and the teaching team. Your team's final presentation grade will be determined by a weighted average of the two (60% teaching team and 40% peers).

4.1 Submission via Gradescope

Each group should submit slides as a PowerPoint file to Gradescope by **11:59am the night before their presentation day (12/02/2024 for 12/03/2024 presentations and 12/04/2024 for 12/05/2024 presentations).** If you do not submit your slides by the deadline for your presentation day, your group will have to present without slides. Presentation order will be decided in class the week before presentations are due.

5 Report (due 12/09/2024 @ 11:59pm (no penalty extension until 12/19/2024))

Your written deliverables for the project include both a group and individual report.

5.1 Group Report

You will submit a written report describing your work in this project, similar in nature to the lab reports you have written throughout the semester. **The report must be no more than 12 pages in Times New Roman 11pt font with 1-in margins.** References are not included in the 12 page limit and may take as many pages as necessary. You are free to use whatever typesetting workflow you are comfortable with, but we highly recommend using LaTeX. Handwritten submissions will not be accepted. If you need to sketch some diagrams by hand, this is okay.

5.1.1 Report Rubric

The report will be graded according to a similar rubric as our lab reports. Your report should be thorough, addressing in detail all aspects of the task at hand, your approach, your implementation, the testing process, and analysis of the results. The gold standard is that another team who completed the prior labs in the course would be able to reproduce your approach in the final project from your report alone.

1. **Scope** *(5 pts)*: Was the project of appropriate scope? Did the report address the task with no obvious holes?
2. **Method** *(10 pts)*: Was the approach technically sound and reproducible? Was it complete and free of error or bias?
3. **Evaluation** *(10 pts)*: Were all relevant results reported? Are the cases chosen sufficient to demonstrate advantages and limitations?
4. **Analysis** *(10 pts)*: Was the analysis complete, free of error, and based on data/observations?
5. **Lessons Learned** *(5 pts)*: What are some insights that you gained? What worked well, what did not? Why?
6. **Clarity** *(10 pts)*: Was the report clear and organized?

We wish to emphasize that the report is the **primary deliverable** for the Final Project, since your performance in the Final Competition does not directly impact your grade, rather serving as an opportunity to evaluate the performance of your solution in a practical setting.

5.2 Individual Report

The individual write-up is worth 5 pts and should contain a peer evaluation. Describe the specific contributions of each individual to the group effort, and assign a grade to each of your team members (including yourself). Each team member should be scored out of 5 points. The score assigned to each member for the individual report will be the average of scores given to that member from this peer evaluation.

5.3 Submission via Gradescope

- **Group Submission:** Each group should only submit once to Gradescope per Gradescope assignment. While going through the submission please ensure that you **add your group members to the submission**. Failure to do so could impact the grades of all parties involved.

  Submit a PDF of your report to the Group Report assignment, and your code to the Code assignment (solely as supplementary material). Note: You may edit your code after the competition. Please make sure that your final submission to the Code assignment is the version discussed in your report.

- **Individual Submission:** Submit a copy of your individual report to the Individual Report assignment.