MTRN4230 T2 2022 Project 2: Path Planning

Learning Outcomes

- LO1: Employ a robot and robot programming environment effectively and efficiently to achieve a given task
- LO4: Compare and evaluate different robot manipulator designs and their application

Aims

- To use the UR5e robotic arm to solve a simulated industrial palletising task.

Due Date

- Report (pdf), Code (matlab), Video submission: Week 12, Friday 17:00 pm (AEST)
- Upload the video to YouTube or any cloud storage medium such as Google Drive and insert a link to it in your project report.

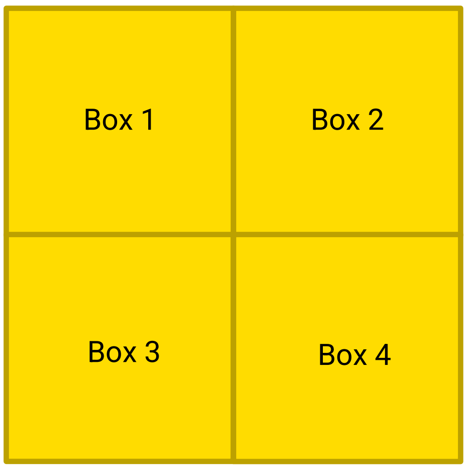
The Task:

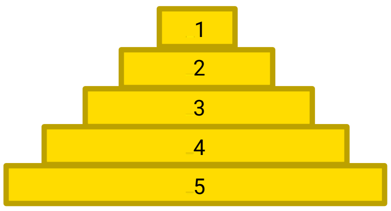
You are working at a company and have been assigned a task of programming a robotic arm to complete a palletising task. Your company receives various parcels stacked according to size and must distribute them according to destination. The number of parcels that you receive can vary and their destinations can vary as well.

You will receive several parcels stacked according to size in Box 1. The largest parcel will be at the bottom of the pile with the smallest parcel at the top of the pile. Box 1 can have a stack of 3 – 5 parcels at any one time. Box 2 - 4, are going out to customers and need to be filled with the correct parcels. Your task is to design a path planning like algorithm to program the robotic arm to move the correct parcels from Box 1 to their correct destination boxes.

There are a few restrictions that you need to take into consideration when solving this problem.

1. If a larger parcel is placed on top of a smaller parcel, the smaller parcel will be damaged. So, this is not allowed. You can only ever place smaller parcels on top of larger ones.
2. When moving a parcel, the robotic arm is only allowed to place parcels in one of the boxes, not in a location outside the working area.

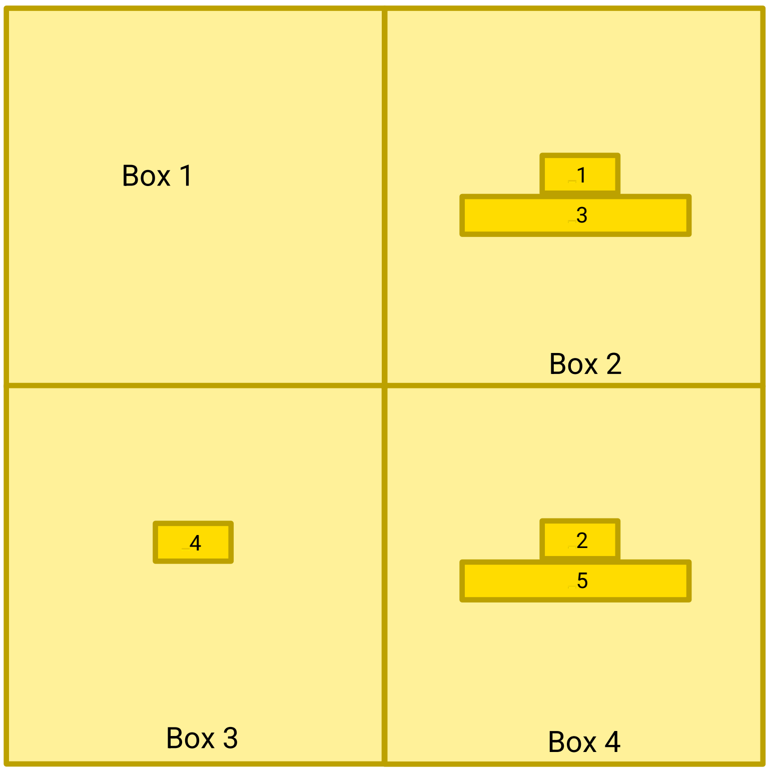




As an example, scenario Box 1 may receive 5 parcels stacked according to ascending size order. Parcel 1 is the smallest and is at the top of the stack. Parcel 5 is the largest and is at the bottom of the stack.

Parcels 1 and 3 need to go to Box 2. Parcels 4 needs to go to Box 3 and finally parcel 2 & 5 need to go to Box 4.

Figure 1 Example Stack



This is summarised below:

Box 1 Contains : 1 & 3

Box 2 Contains : 4

Box 3 Contains : 2 & 5

You will always receive a minimum of 3 parcels in Box 1 and a maximum of 5 parcels.

Part A: Using the Vacuum Gripper

This is a pre-requisite for the remainder of the assessment. It will be assessed in the video in Part C.

Write a function to pick up a parcel from a given box and put it down in another box. This functionality will be necessary for the remaining parts. The vacuum gripper functionality can only be tested in the lab.

The (x,y) coordinate from the base of the UR5e of each of the boxes are as follows. They have already been marked out on the tables with masking tape.

Box 1: [-700, -100]

Box 2: [-500, -100]

Box 3: [-700, -300]

Box 4: [-500, - 300]

The design of this is up to you. You may have one function indicating where to pick up from and another to put the parcel down. You may decide to just have one function for the entire motion.

The parcels that will be used in the lab are squares of varying sizes laser cut from **3mm plywood**.

Parcel 1: 7 x 7 mm

Parcel 2: 8 x 8 mm

Parcel 3: 9 x 9 mm

Parcel 4: 10 x 10 mm

Parcel 5: 11 x 11 mm

Part B: The Algorithm (9/20)

This section requires students to write an algorithm capable of sorting any arrangement of received parcels. Your solution will be auto marked using the project2sim provided, it will not be tested on the real robot. You can expect that you will only receive 3, 4 or 5 parcels in box 1 which will then need to be sorted. Your solution needs to be able to handle as many of the cases as possible. Only a few example cases have been listed in the following sections, but the cases used for auto marking will be much more exhaustive. You are only allowed to submit one solution for sorting all cases of the parcel. Separate files for sorting 3 parcels, 4 parcels and 5 parcels will not be accepted. Your solution could be one file or many files If you create a class for example, but it must only ever return one project2 function in the format specified.

Even if you are unable to complete all parts (B.1, B.2, B.3) successfully, we will auto mark for entire list of tests cases prepared. You may pass all the cases for sorting 3 parcels, so you will receive the entire 3% of the mark. You may only pass 10 out of 30 (this is just an arbitrary number) test cases for sorting 4 parcels so you will receive 1%. Similarly, you may only pass 1 or 2 test cases for sorting 5 parcels, so you will get marks according to how many test cases you pass.

When testing your algorithm on the real robot, you are to create a separate file. Have 1 separate file for testing on the simulator and have another file for testing on the real robot. You will be required to submit both for plagiarism checking,

B.1 Sort 3 Parcels (3 %)

Write an algorithm for a dynamic solution to solve the task of receiving 3 parcels and distributing them to the correct boxes.

Some examples include: (The following is not an exhaustive list of all possible cases)

Box 2: 1 2 3 **Box 3:** Empty **Box 4:** Empty	**Box 2:** Empty **Box 3:** 1 2 3 **Box 4:** Empty	**Box 2:** Empty **Box 3:** Empty **Box 4:** 1 2 3	**Box 2:** 1 **Box 3:** 2 **Box 4:** 3
Box 2: 1 **Box 3:** 3 **Box 4:** 2	**Box 2:** 1 2 **Box 3:** 3 **Box 4:** Empty	**Box 2:** 1 3 **Box 3:** 2 **Box 4:** Empty	**Box 2:** Empty **Box 3:** 2 **Box 4:** 1 3

B.2 Sort 4 Parcels (3 %)

Extend your algorithm from the previous question, to receive 4 parcels and distributing them to the correct boxes.

Some examples include: (The following is not an exhaustive list of all possible cases)

Box 2: 1 2 3 4 **Box 3:** Empty **Box 4:** Empty	**Box 2:** Empty **Box 3:** 1 2 3 4 **Box 4:** Empty	**Box 2:** Empty **Box 3:** Empty **Box 4:** 1 2 3 4	**Box 2:** 1 **Box 3:** 2 **Box 4:** 3 4
Box 2: 1 **Box 3:** 3 **Box 4:** 2 4	**Box 2:** 1 2 **Box 3:** 3 4 **Box 4:** Empty	**Box 2:** 1 3 **Box 3:** 2 **Box 4:** 4	**Box 2:** Empty **Box 3:** 2 4 **Box 4:** 1 3

B.3 Sort 5 Parcels (3 %)

Extend your algorithm from the previous question, to receive 5 parcels and distributing them to the correct boxes.

Some examples include: (The following is not an exhaustive list of all possible cases)

Box 2: 1 2 3 4 5 **Box 3:** Empty **Box 4:** Empty	**Box 2:** Empty **Box 3:** 1 2 3 4 5 **Box 4:** Empty	**Box 2:** Empty **Box 3:** Empty **Box 4:** 1 2 3 4 5	**Box 2:** 1 **Box 3:** 2 5 **Box 4:** 3 4
Box 2: 1 5 **Box 3:** 3 **Box 4:** 2 4	**Box 2:** 1 2 **Box 3:** 3 4 **Box 4:** 5	**Box 2:** 1 3 5 **Box 3:** 2 **Box 4:** 4	**Box 2:** 5 **Box 3:** 2 4 **Box 4:** 1 3

Part C: Report and Video (11/20)

Video (3%)

As there is no in person demonstration for students to show us their solution working, students are required **to take a video and upload it to Youtube or any cloud storage medium such as google drive and insert a link in the report**. Describe what is happenning in the video.

The video must show your solution implemented on the UR5e in the labs. Depending on what you have completed, you have the option of demonstrating your solution for **Sorting 3 Parcels**, **Sorting 4 Parcels OR Sorting 5 Parcels.**

The video must display the following:

1. Show the destination of the parcels in your MATLAB code before the sorting begins.
2. The vacuum gripper picking up a parcel.
3. The vacuum gripper placing a parcel.
4. Ability to pick up and place more than one parcel.
5. Verification that it has completed the task successfully.

The most important objective is to show that you can pick up and put down many parcels using the vacuum gripper. Your algorithm will not be considered in this stage.

We would love to see your solutions with the entire algorithms implmented if you have the opportunity to show us!

Video Length: 10 mins max. If the link is corrupted or inaccessible you will not be awarded the marks.

Report: (8%)

1. Explain your solution:
   a. Consider things such as:
      i. Did you create a hardcoded solution for **Sorting 3 Parcels?** Did consider all of the cases. If yes, how?
      ii. Did you come up with a dynamic solution? How did you come up with this solution? How does it work?
   b. What are some advantages to your solution? (Software design as well as effectiveness of implementing it in the real world)
   c. What are some limitations to your solution? (Software design as well as effectiveness of implementing it in the real world)
   d. Is it scalable? What if you received "N" number of parcels and had "M" number of destination boxes?
   e. What challenges did you have when implementing your solution? This includes both any software/algorithm related challenges as well implementing the solution in the real world. I.e. Your solution may have worked perfectly with the Project2Sim. Did it work immediately when you tried it out on the real UR5e? Were there any unexpected things that you had to take into consideration?
2. What considerations did you/can you make to improve the repeatability and efficieny of your solution and why? Consider both software changes and physical changes to the workspace.
3. Compare and contrast the different types of robotic arms configurations presented in the lectures to identify the best suited robotic arm for this task and to discuss the suitability of the UR5e for this application.

Marking Criteria

Algorithm (9/20 %)

This section will be auto marked. Marks will be allocated according to the number of cases passed during testing. Automaking for starting with 3, 4 and 5 parcels in Box1 will be tested. Only one solution can be submitted. There must not be separate solutions for sorting 3, 4 or 5 parcels.

Video (3/20 %)

Item	Value	Description
Pickup Parcel	1 %	Video clearly shows the vacuum gripper is able to pick up a parcel from a box.
Putdown Parcel	1 %	Video clearly shows the vacuum gripper is able to put down a parcel in a different box from which it was picked up
Multiple Parcels	1 %	Video clearly shows that the ur5e is able to pick up and move multiple parcels in the same program without any issues. You must have at least 3 parcels in box1.

Report (8/20 %)

Question 1: Explain your solution (4/8 %)

Poor (0 – 1%)	Insufficient (1 – 2 %)	Developing (2 – 3 %)	Accomplished (3 – 4 %)
(i) Provides little information about their solution. Difficult to understand. (ii) Provides limited or no discussion of the progression, challenges faced, advantages, limitations, or scalability of the solution.	(i)Provides a basic explanation of how the solution works. (ii) Limited or no details outlining the progression of the solution (how they started and how the solution was improved over time). (iii)Limited or no details outlining the challenges (software and real world) faced and how it influenced the design of the solution. (iv) Provides some information of the advantages, limitations and scalability of the solution. (Software and implementation of the real world)	(i) Provides a detailed explanation of the concepts and procedure behind the solution. (ii) Provides some details on the progression of the solution from start to finish detailing why and how changes were made. (iii) Provides some details on the challenges faced (software and real world) and how it influenced the design of the solution. (iv) Provides detailed information of the advantages, limitations, and the scalability of the solution. (Software and implementation of the real world)	(i) Provides in depth details and easy to understand explanation of the concepts and procedure behind the solution. (ii) Provides in depth and easy to understand details of the progression of the solution from start to finish detailing why and how changes were made. (iii) Provides in depth and easy to understand details on the challenges (software and real world) faced and how it influenced the design of the solution. (iv) Provides in depth details and easy to understand information of the advantages, limitations, and the scalability of the solution. (Software and implementation of the real world)

Question 2: Improvements for repeatability and efficiency (2/8 %)

Poor (0.25 - 0.5)	Insufficient (0.5 – 1 %)	Developing (1 – 1.5 %)	Accomplished (1.5 – 2 %)
(i) Identifies limited software or physical modifications to	(i) Identifies some software and/or physical modifications to	(i) Identifies software and physical modifications to	(i) Identifies software and physical modifications to improve repeatability or efficiency.

improve repeatability or efficiency.	improve repeatability or efficiency. (ii) Provides some justification on why the suggested changes are benifical.	improve repeatability or efficiency. (ii) Provides some justification on why the suggested changes are benifical by analysing the cause and effect. (iii) Provides some evidence of having done some experiements with qualitative data to support that the identified changes do improve reliability a efficiency.	(ii) Provides some justification on why the suggested changes are benifical by analysing the cause and effect. (iii) Provides evidence of experiementation with qualitativeobservations to support that the identified changes do improve reliability and efficiency (iv) Considers the effects of the improvements if the solution was implemented in a real factory and how it can benefit the day to day work flow.

Question 3: Best Suited Robotic Arm for the task (2/8 %)

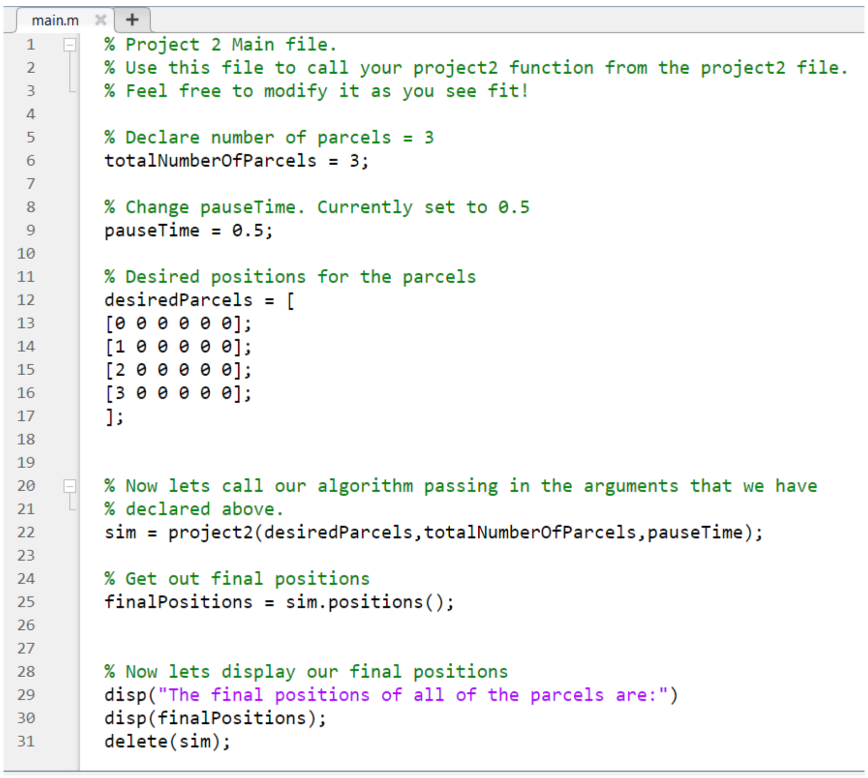
Poor (0.25 - 0.5)	Insufficient (0.5 – 1 %)	Developing (1 – 1.5 %)	Accomplished (1.5 – 2 %)
(i) Identifies a best suited robotic arm for this task. (ii) Provides limited or no discussion on justifying the chosen robotic arm.	(i) Identifies a best suited robotic arm for the task. (ii) Provides a limited comparisons between the chosen robotic arm and the UR5e robotic arm to justify chosen robotic arm.	(i) Identifies a best suited robotic arm for the task. (ii) Provides detailed comparison between multiple robotic configuration types mentioned in the course (cartesian, scara, articulated & delta) to justify the chosen robotic arm.	(i) Identifies a best suited robotic arm for the task. (ii) Provides detailed comparison between multiple robotic configuration types mentioned in the course (cartesian, scara, articulated & delta) to justify the chosen robotic arm. (iii) Shows evidence of having performed external research to identify what specific robots are employed in an industrial setting for palletising to justify the chosen robotic arm.

Starter Code Details

Please watch the video released along with the assessment.

Main.m

Use this file to call your algorithm in the project2.m file with arguments that you want.


main.m

```
% Project 2 Main file.
% Use this file to call your project2 function from the project2 file.
% Feel free to modify it as you see fit!

% Declare number of parcels = 3
totalNumberOfParcels = 3;

% Change pauseTime. Currently set to 0.5
pauseTime = 0.5;

% Desired positions for the parcels
desiredParcels = [
[0 0 0 0 0 0];
[1 0 0 0 0 0];
[2 0 0 0 0 0];
[3 0 0 0 0 0];
];


% Now lets call our algorithm passing in the arguments that we have
% declared above.
sim = project2(desiredParcels,totalNumberOfParcels,pauseTime);

% Get out final positions
finalPositions = sim.positions();


% Now lets display our final positions
disp("The final positions of all of the parcels are:")
disp(finalPositions);
delete(sim);
```

Variables:

totalNumberOfParcels

This variable indicates the total number of parcels that are to be received in box 1. The project2simulator will create however many parcels specified by this variable. It can spawn a minimum of 1 parcel and a maximum of 6 parcels.

pauseTime:

This is the pause time in seconds required to visualise the transitions for the project2Simulator. You can increase or decrease this value. At the moment it is set to 0.5 seconds.

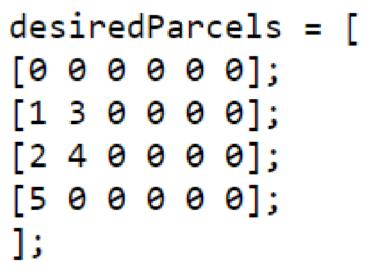
desiredParcels:

This 4x6 matrix contains the desired final positions of all the parcels. The rows indicate the box; row 1 indicates box 1, row 2 indicates box 2 and so on. The format will be exactly for all test cases during marking. Row 1 will never contain any parcels as this is the starting location. Only rows 2, 3 and 4 will contain parcels. Parcel 1 is the smallest parcel and parcel 5 is the largest parcel. Each row in the matrix will always be in ascending order, with the stack in real life having the smallest parcel on the top and the largest parcel on the bottom.

Box 2 contains parcel 1

Box 3 contains parcel 2

Box 4 contains parcel 3



```
desiredParcels = [
[0 0 0 0 0 0];
[1 3 0 0 0 0];
[2 4 0 0 0 0];
[5 0 0 0 0 0];
];
```

Another example is shown on the left.

In this array there are a total of 5 parcels that have been declared.

Box 2 contains parcels 1 & 3 (Parcel 1 is stacked on top of parcel 3)

Box 3 contains parcels 2 & 4 (Parcel 2 is stacked on top of parcel 4)

Box 4 contains parcel 5

Line 22:

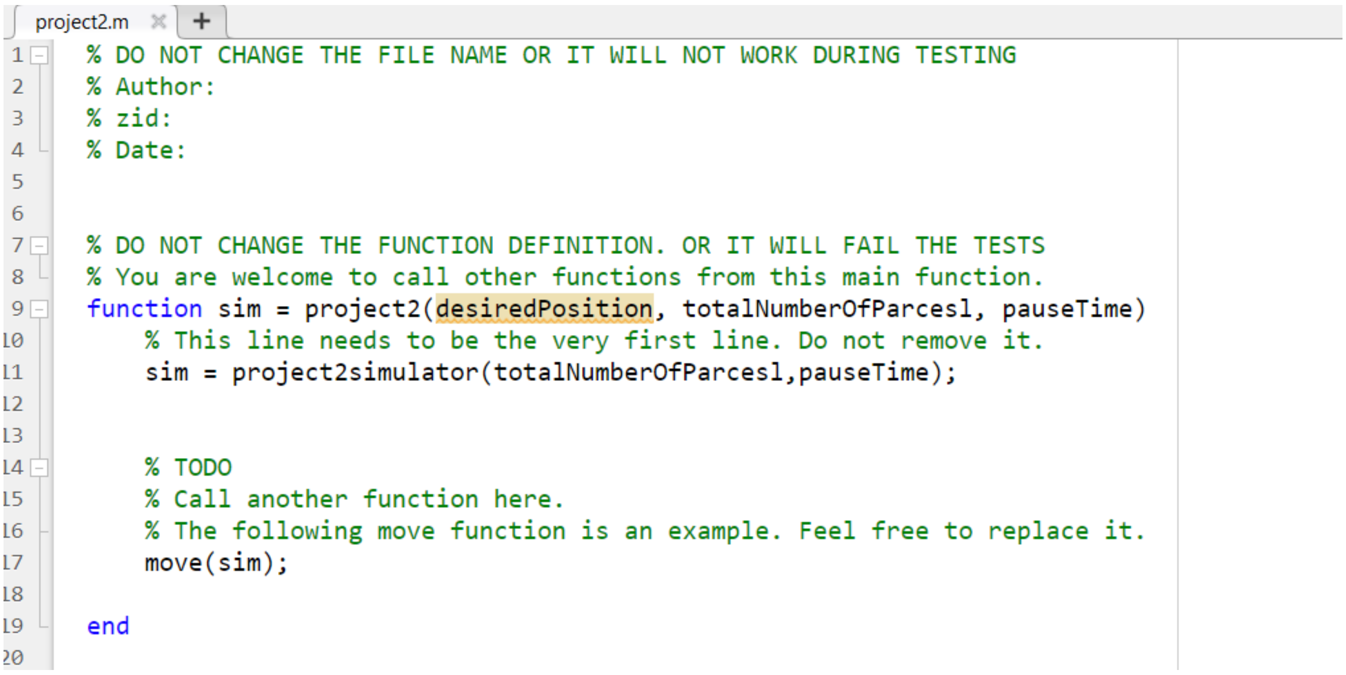
Here we are calling the project2 function which is inside the project2.m file. We are passing in all of the arguments that we declared above. You will need to go to the project2.m file and place your algorithm/solution in this function.

Line 25 onwards:

Once the algorithm has finished, we access the final positions of all the parcels and print it out. If your algorithm was successful, it should be identical to the desiredParcels matrix.

Project2.m

This is the file where you will need to create your algorithm. You must not change the function definition of the projec2 function as the automarker will fail if changed. You must not also change the name of this file for the same reason. The project2 function must also be the first function in this file. You can create additional functions after this function and call it from inside the project2 function.

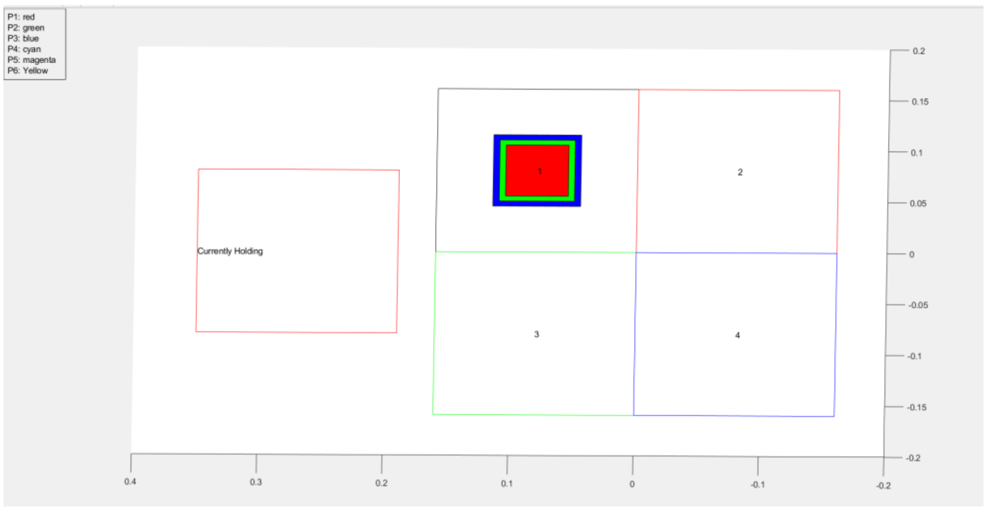


project2.m

```
1  % DO NOT CHANGE THE FILE NAME OR IT WILL NOT WORK DURING TESTING
2  % Author:
3  % zid:
4  % Date:
5
6
7  % DO NOT CHANGE THE FUNCTION DEFINITION. OR IT WILL FAIL THE TESTS
8  % You are welcome to call other functions from this main function.
9  function sim = project2(desiredPosition, totalNumberOfParcesl, pauseTime)
10     % This line needs to be the very first line. Do not remove it.
11     sim = project2simulator(totalNumberOfParcesl,pauseTime);
12
13
14     % TODO
15     % Call another function here.
16     % The following move function is an example. Feel free to replace it.
17     move(sim);
18
19 end
20
```

Line 11:

This function calls the constructor for the project2 simulator. This function takes in the 'totalNumberOfParcels' and 'pauseTime' which were set in the main function. When you call this function, it will show a figure that looks like the following. There are four rectangles labelled 1 – 4. There is a 'currently holding' area which indicates what parcel the vacuum gripper is currently holding and there is a legend on the top left indicating what colours each parcel is. In this you can also see that there are currently 3 parcels stacked in box 1.

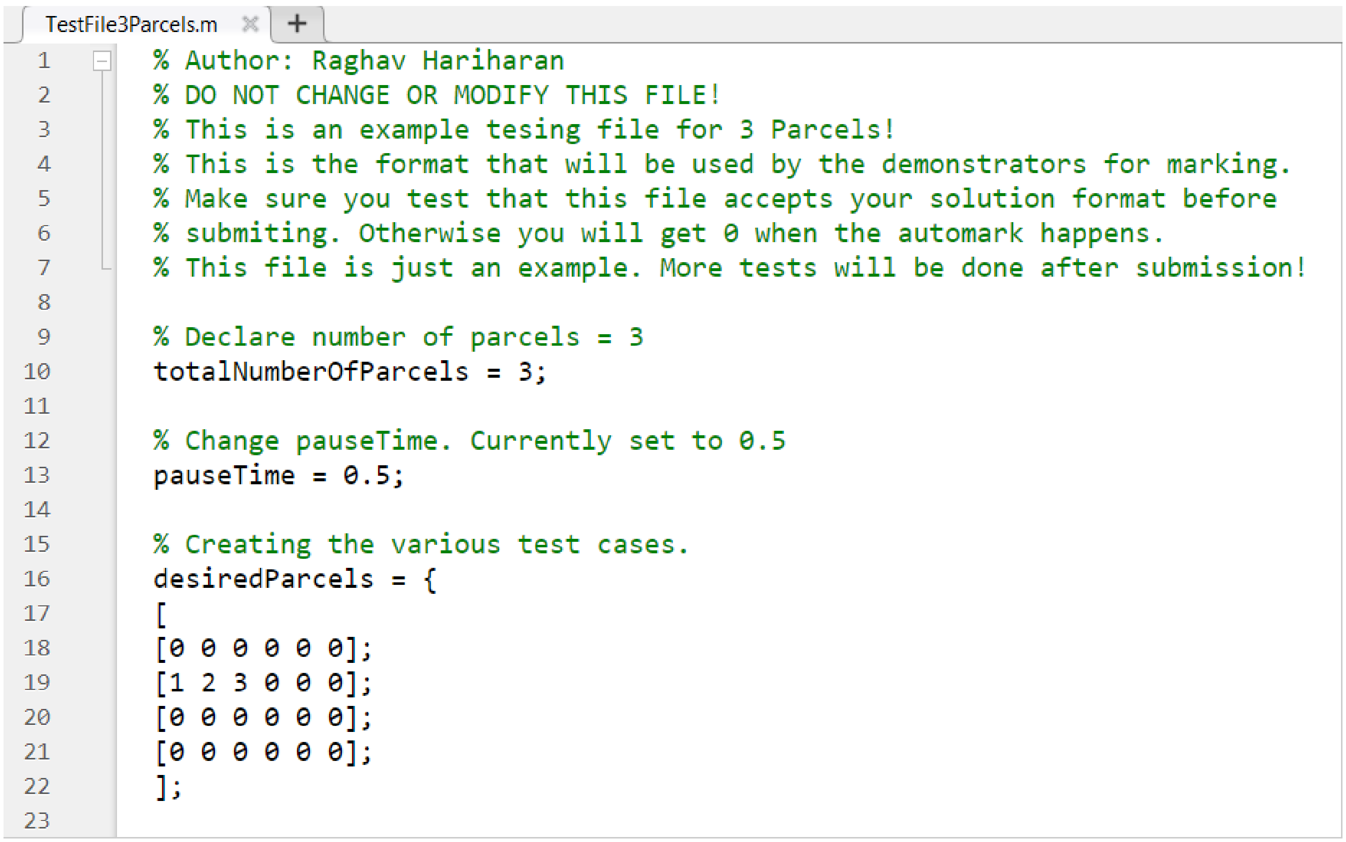


Line 17:

This line calls a different function called move passing in the constructor for the simulator that was created. It has been declared under the project2 function. There is some example code in that function that you can go through in your own time. There will a be a video (possibly during a lecture) that will go through this function as well as the other provided starter code in detail.

TestFile3Parcels.m

A sample testfile has been provided to students. This test file is identical to the one that will be used during auto marking. You need to ensure that your solution is compatible with the format that this test file requires otherwise your solution will not be marked properly. Please watch the demonstration video (could be during a lecture) to understand how this file works.



```
% Author: Raghav Hariharan
% DO NOT CHANGE OR MODIFY THIS FILE!
% This is an example tesing file for 3 Parcels!
% This is the format that will be used by the demonstrators for marking.
% Make sure you test that this file accepts your solution format before
% submiting. Otherwise you will get 0 when the automark happens.
% This file is just an example. More tests will be done after submission!

% Declare number of parcels = 3
totalNumberOfParcels = 3;

% Change pauseTime. Currently set to 0.5
pauseTime = 0.5;

% Creating the various test cases.
desiredParcels = {
[
[0 0 0 0 0 0];
[1 2 3 0 0 0];
[0 0 0 0 0 0];
[0 0 0 0 0 0];
];

```

Project2Simulator docs.pdf

This file provides the api for the project2simulator. It describes what functions are available for you to call and provides a description of what they do. Please read through it.

Hints: How to start:

1. Try to first complete Part A. Make sure you can pick up more than one thing and move it.
2. Try solving the task of moving an entire stack of 3 parcels to another box.
3. This task is a modified version of the tower of Hanoi puzzle. There are plenty of resources online which provide a solution to it. For example, here are a couple YouTube videos:
   a. https://www.youtube.com/watch?v=YstLjLCGmgg&t=105s&ab_channel=GeeksforGeeks
   b. https://www.youtube.com/watch?v=rf6uf3jNjbo&ab_channel=Reducible

   Try implementing this algorithm first and developing it further. In the video they have 1 auxiliary rod (aka box) but in our task we have 2 spare boxes. How can you modify the solution to take advantage of the space box? How do you decide which spare box to move the parcel to?

   In the tower of Hanoi puzzle the entire stack is move to a different rod. In our task each parcel can end up in a different box. How can you modify the tower of Hanoi solution to take this into consideration?