

## ME 435 – Mechanical Engineering Systems Lab

Level

**4A**

### Autonomous Vehicle Lab 3

#### Activities:

##### Path Planning

1. Where am I?
  - 1.1 Networking
  - 1.2 Camera Calibration
  - 1.3 April tag
2. (Lab 4) Where should I go next?
  - 2.1 Path Planning using MATLAB

#### Learning Objectives:

After completing this lab, students will be able to:

- Understand basic networking to exchange information between your computer and multiple Jetbots and operate them interactively.
- Use April tag to identify robot locations
- Perform path planning with occupancy grid maps

#### 1. Materials:

- 1.1. A Jetbot
- 1.2. A second Jetbot (without wheels and mounted on a stand), called a CameraBot.
- 1.3. A Computer
- 1.4. Road map set up by the TAs with several intersections and a final goal.
- 1.5. Map image (a .jpg file) provided by a TA.

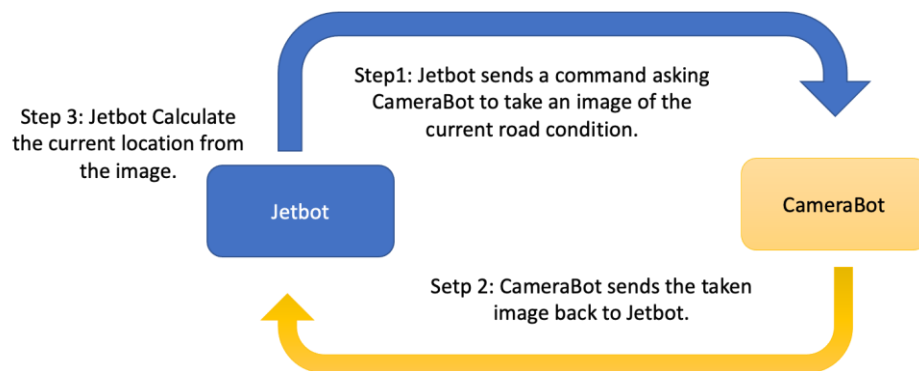
**Note: All questions you must answer are highlighted in yellow in this document.**

#### Part 1 - Where am I?

##### 1.1 Networking

Networking allows interaction between robots and computers. In Part 1 of this lab, we will use networking techniques to have the Jetbot send a command to the CameraBot to image the current road condition. Then, the CameraBot sends the image back to the Jetbot. The Jetbot will calculate its current location from the image, and then plan for the next step. This will be discussed in Part 2 of this lab and will continue in Lab 4. This process is relatively simple compared to how a commercial self-driving car defines its current

location using a variety of sensors, e.g., radar, lidar, sonar, GPS, odometry, and inertial measurement units. Here, we will focus on learning basic network communication.



## 2. Procedures:

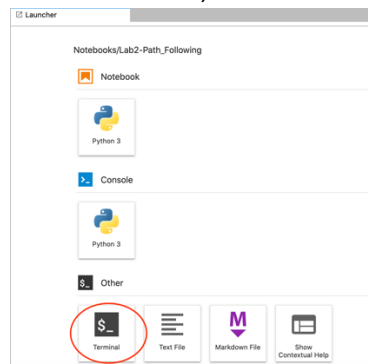
### 2.1. Understand MAC Address, IP address and hostname.

A **MAC address** is a **unique permanent identity** of a device provided by the manufacturer, and an **IP address** is an identity provided by the internet service provider, which will **change when you change a network**. MAC address has a six-byte hexadecimal address format, e.g., 00:1B:44:11:3A:B7, while IP addresses are coded in 32 bits e.g., 192.158.1.38. MAC address is like your given name, and IP address is like your PSU student ID. While MAC address is very reliable and stable, it's clumsy to work within computers. Therefore, IP address is often used due to its ease of convenience. However, it is hard for a human to remember the IP address of all the devices. Therefore, we often define a "**nickname**" to each device, which are called the **hostnames**.

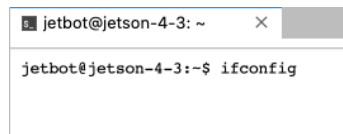
### 2.2. Find the IP address and hostname of your Jetbot.

2.2.1. Turn on the Jetbot. Then open a web browser on your computer to access JupyterLab using the IP address of your Jetbot.

2.2.2. Under Launcher, choose **Terminal**



2.2.3. We will use the command "ifconfig" to find the network properties of the Jetbot, and the command "hostname" to find the hostname. Under the Terminal window, type `ifconfig`



```
jetbot@jetson-4-3: ~  
jetbot@jetson-4-3:~$ ifconfig
```

You should find the IP address in the paragraph labeled "wlan0", which is your wireless interface. The number after "inet" is your IP address. That should match with the number displayed on the OLED of your Jetbot.

2.2.4. Type `hostname` under Terminal. You should find the hostname of your Jetbot.

**What is the hostname of your Jetbot?**

Hostname: nano-4gb-jp45

```
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 104.39.24.50 netmask 255.255.0.0 broadcast 104.39.255.255  
    inet6 fe80::d5e6:8148:2b7a:212 prefixlen 64 scopeid 0x20<link>  
    ether e8:84:a5:f6:c7:23 txqueuelen 1000 (Ethernet)  
    RX packets 8960 bytes 10864112 (10.8 MB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 27530 bytes 38242280 (38.2 MB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2.3. Access another device on your local network using SSH (Secure Shell).

Now, we will use the Jetbot to access the CameraBot.

**\*\* Two teams will share one CameraBot to complete lab (sections 1.1 and 1.2). We recommend two teams perform sections 1.1 and 1.2 together**, while section 1.3 has to be done separately.

2.3.1. **Locate and choose a CameraBot.** Two Camerbots (two black boxes) are mounted to the motorized ceiling elevator in the UAV room.

2.3.2. **Check and make sure the CameraBot is turned ON.**

2.3.3. **Find the IP address of your CameraBot.** Each CameraBot is connected to a wireless HDMI display, mouse, and keyboard, placed near the window. Each station is labeled with the corresponding CameraBot is close to the window or door.

Similar to the Jetbot, CameraBot uses the Linux Ubuntu operating system. The desktop display should look similar to the picture below. Click on the **Dash** and **search for the Terminal** application.



2.3.4. In the **Terminal** app and use what you learn in the previous section to find the CameraBot's IP address.

**Record the Camerbot IP address here:** [104.39.91.114](#)

2.3.5. You will use your **Jetbot (the Green Robot)** for the following steps.

2.3.6. **Use your Jetbot to remote into the CameraBot.** Under the **Terminal** of your **Jetbot**, type `ssh jetbot@XXXXXX`, **replace XXXXXX by the IP address of your CameraBot.** If two teams are doing this section together, please take turns to remote into the CameraBot.

2.3.7. If it asks for a password, use jetbot.

2.3.8. You should see a message similar to this example:

```
jetbot@jetson-4-3:~$ ssh jetbot@192.168.1.47
jetbot@192.168.1.47's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.9.140-tegra aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

511 packages can be updated.
408 updates are security updates.

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Mar  9 23:27:45 2021 from 192.168.1.36
jetbot@camerabot-01:~$
```

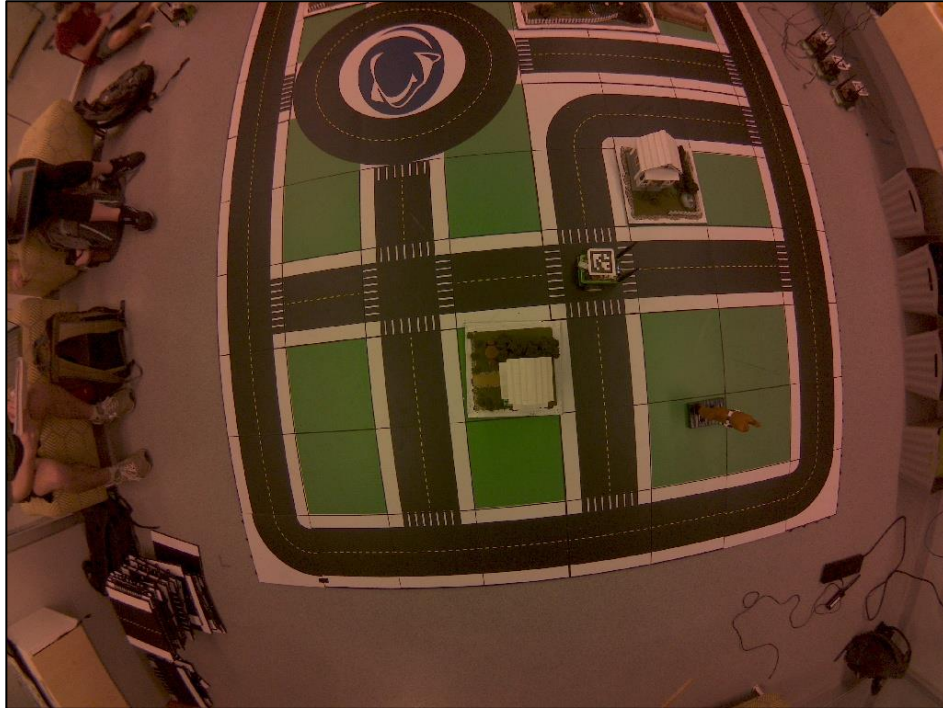
**2.3.9. What is the hostname of your CameraBot?**

[Camerabot-01](#)

2.3.10. Use Ctrl+D to close the connection to the CameraBot and exit. You should see a message "Connection to XXX.XXX.X.XX closed".

2.4. Use Jetbot to command the CameraBot to take a picture and send it back to the Jetbot.

- 2.4.1. Place the Jetbot on an intersection of the road map.
- 2.4.2. Open the Jupyter notebook called: Lab3\_Remote\_access.ipynb, then follow the notebook's instruction. **Ensure that the correct IP address and local path directory is entered in the script.**
- 2.4.3. **Place the obtained image here after you complete section 1.1 in the Jupyter notebook.**



## Part 1 - Where am I?

### 1.2 Camera Calibration

In our next step, we will place the Jetbot on an intersection of a road and have the CameraBot take an image. We will use the image to determine the current location of the Jetbot. However, many pinhole cameras, especially the cheap ones, have a lot of distortion. Two significant distortions are radial distortion and tangential distortion.

**Radial distortion:** Straight lines bends into circular arcs.

**Tangential distortion:** Some part of an image of a parallel object looks nearer or further apart.

Since we plan to obtain the robot location from an image, these distortions will affect the calculation. We must produce an undistorted image for proper calculation. To do that, we must find the intrinsic and extrinsic parameters of the camera and the distortion coefficient.

#### Procedures:

- 2.5. On your computer, open a **new web browser**. Then, **navigate to [http:// XXX.XXX.X.XX:8888](http://XXX.XXX.X.XX:8888)**, and replace XXX.XXX.X.XX by the IP address of the CameraBot.

- 2.6. Under Notebook, select the CameraCalibration folder, then open **Capture\_and\_Save\_Images.ipynb**, and follow the instructions to collect calibration images.
- 2.7. Once you have collected all the images, you will open the **CameraCalibration.ipynb** file to perform camera calibration and obtain the intrinsic and extrinsic parameters of the camera, and the distortion coefficient.

## Part 1.2 - Results and Discussions:

**What is the camera matrix of your CameraBot?**

```

$$\begin{bmatrix} 508.84921408 & 0. & 533.2761346 \\ 0. & 507.00179869 & 397.95447098 \\ 0. & 0. & 1. \end{bmatrix}$$

```

**What is the distortion coefficient of your CameraBot?**

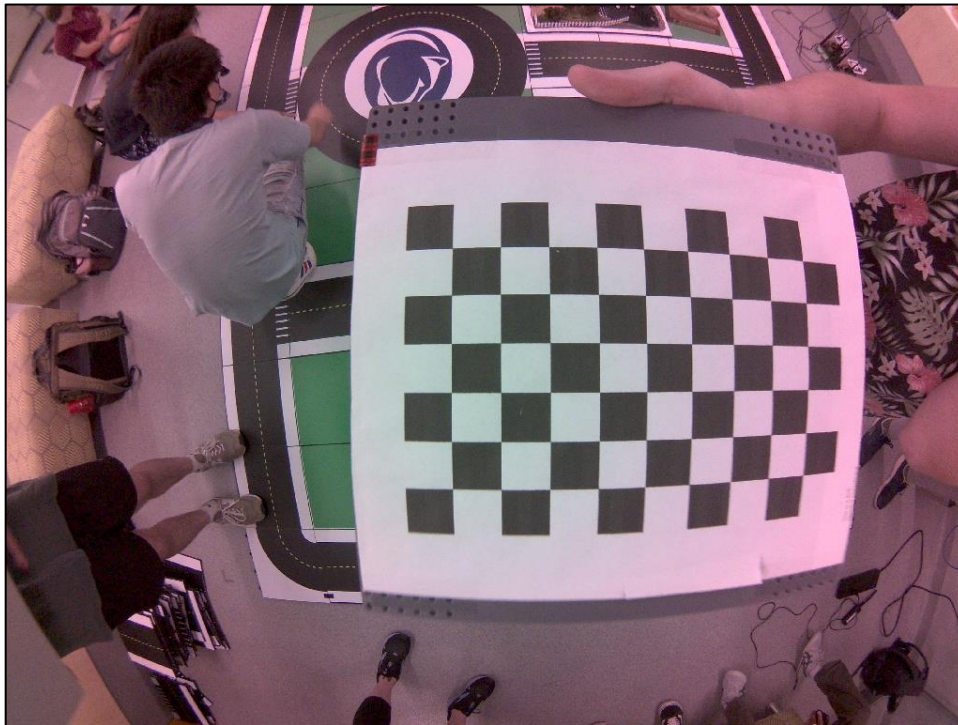
```

$$[k_1 \ k_2 \ p_1 \ p_2 \ k_3] = [-3.37993547e-01 \ 1.39216484e-01 \ -7.54021835e-04 \ -2.70849647e-04 \ -2.99805269e-02]$$

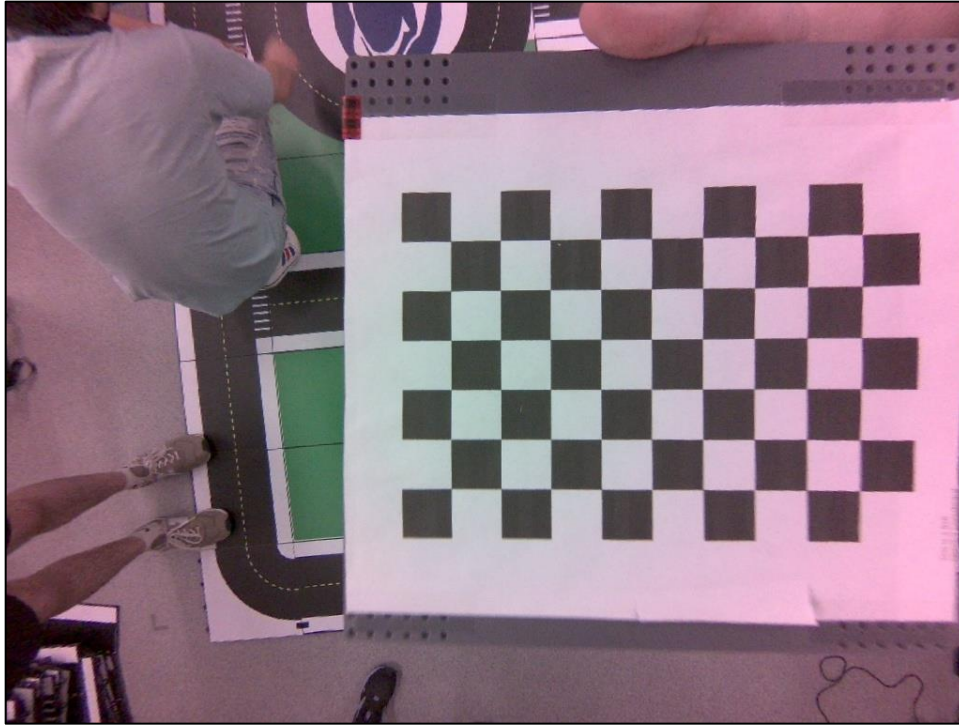
```

**Copy and paste an undistorted image example here. Describe the difference between the two images.**

The undistorted image makes the chessboard appear as a flat surface. Whereas, the distorted image makes the chessboard look slightly rounded/warped. The first figure is the distorted image and the second figure is the undistorted image.







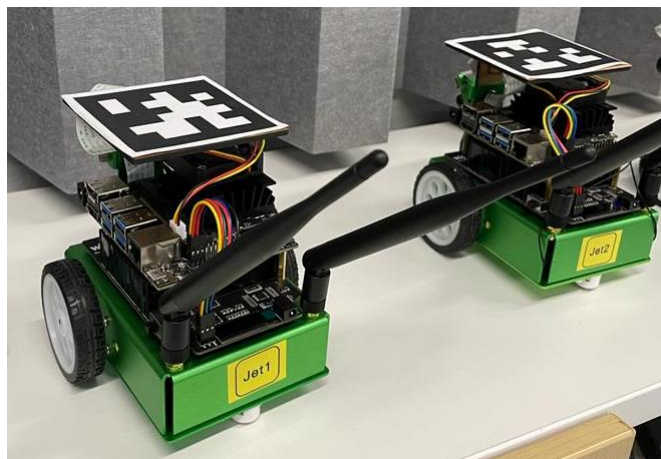
## Part 1 - Where am I?

### 1.3 April Tag

We will continue with the rest of the lab using the Jetbot. Make sure you close the web browser controlling the CameraBot to reduce confusion.

#### Procedures:

- 2.8. In the JupyterLab of the **Jetbot** (Make sure you check the IP address), open **Lab3\_AprilTag.ipynb**, and follow the instructions there.
- 2.9. **Make sure that the antenna on the Jetbot is angled forward so that the camera's view of the April Tag is not obstructed, as shown in the picture below.**



## Part 1.2 - Results and Discussions:

**What is the x, y coordinate and heading angle of your Jetbot on the image from Step 2.4?**

The x,y coordinate and heading angle of jetbot19 are represented in the matrix (645.744116151798, 485.1642619778119, 2.5159753940048417).

X: 645.744116151798

Y: 485.1642619778119

Heading angle: 2.5159753940048417 radians

## Part 2 – Where should I go next?

In Part 2, we will perform path planning using MATLAB. We will be using MATLAB since most students are familiar with the language, and some coding is required for this part of the lab. In JupyterLab, there are multiple ways to run and execute MATLAB code.

### (Optional – Additional knowledge)

Different ways to run and execute MATLAB code in JupyterLab:

1. Using MATLAB Engine API for Python (<https://www.mathworks.com/help/matlab/matlab-engine-for-python.html>)
2. Installing JupyterLab- MATLAB and running MATLAB within JupyterLab (if MATLAB is installed in the Jetbot)
3. Using SSH to access MATLAB in your local computer like how we access the CameraBot
4. Many other methods.

**For the purposes of the lab, we will be running MATLAB on your laptops, and not on the Jetbot.**

We will perform path planning using MATLAB and the data obtained from Part 1. The goal of this part is to use the shortest distance approach to reach the final target location. Open the BOM\_Path\_Finding.mlx file and following the instructions in the MATLAB Live Script.

### 3. Materials:

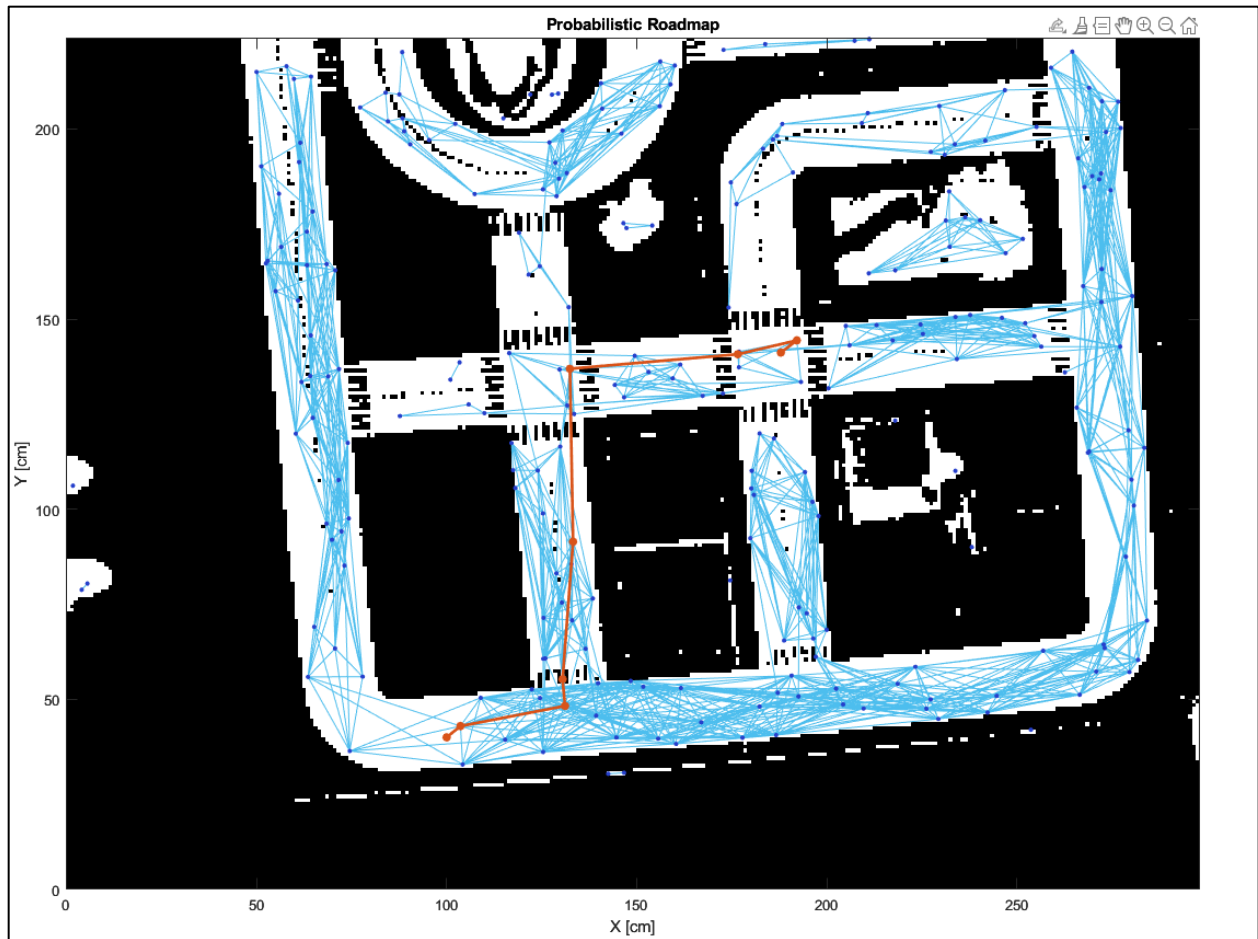
- 3.1. A Jetbot
- 3.2. A second Jetbot (without wheels and mounted on a stand), called a CameraBot.
- 3.3. Road map set up by the TAs with several intersections and a final goal.
- 3.4. Map image (a .jpg file), available on Canvas. Please select the proper image based on which CameraBot you used.

## Part 2 - Results and Discussions:

### MATLAB LiveScript Step 4:

**Copy and paste your Probabilistic Roadmap with a found path here.**





List the x and y coordinates of the found path here.

```
path = 9x2
    187.9115    141.1828
    192.0972    144.3873
    176.5740    140.7453
    132.4200    136.9226
    133.2539     91.4463
    130.4885     55.2877
    131.1350     48.2608
    103.6381     42.9602
    100.0000     40.0000
```

#### MATLAB LiveScript Step 5:

Label the manually selected current node at the first intersection in the Probabilistic Roadmap, which the team provided in the previous question. Use your human decision, should the Jetbot turn left, turn right or go straight at that intersection?

The current node at the first intersection would be, node 2,  $x=192.0972$   $y=144.3873$ .

The jetbot should go straight at that intersection and turn left when it reaches the node 3. Then, the robot will go straight and turn right when it reaches the node 7 and go straight until it reaches node 9.

In Step 5d, how does your algorithm define each of the following outputs: "Go straight", "turn left", and "turn right" signals? Explain your choice.

Test your algorithm by considering the following two scenarios. **Show your work to your TA:**

- Does your algorithm provide the same decision as your human prediction? If not, check your code.
- Assuming your robot is entering the intersection from another way, is the algorithm's decision still correct?

**TA signature:** \_\_\_\_\_

#### Submission

- Save this document into a PDF for submission. Besides, you must upload your MATLAB code in PDF format.
- ***A Friendly Reminder:*** the team needs to submit a video journal as the final report of the UAV modules—document the process by taking lots of videos!