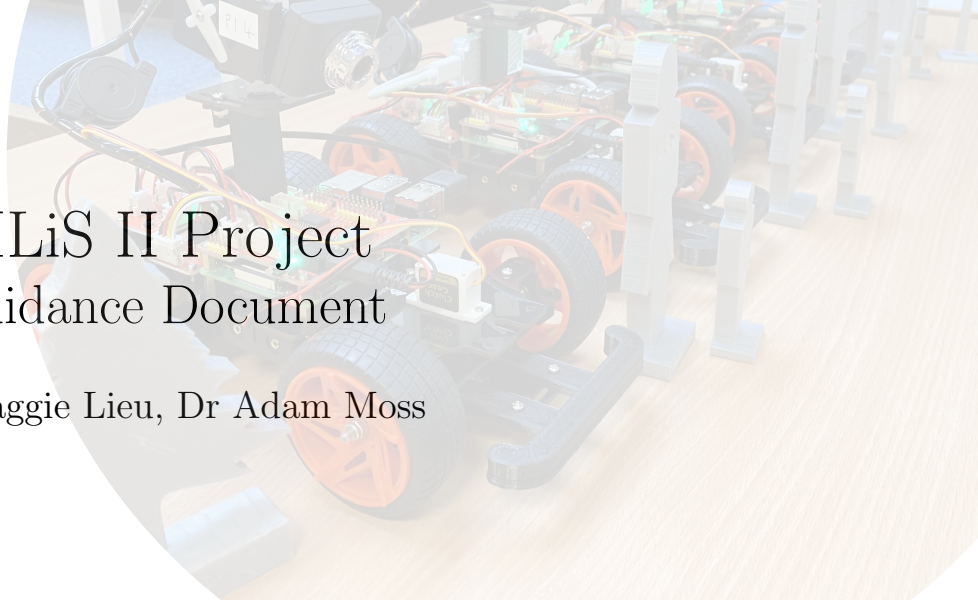

MLiS II Project
Guidance Document

Dr Maggie Lieu, Dr Adam Moss

1 Hardware

1.1 SunFounder Kit

The main body of your car is the SunFounder PiCar-V kit V2 (`https://www.sunfounder.com/smart-video-car-kit-v2-0.html`). This has been pre-assembled and intially calibrated for you (we discuss how to adjust the calibration in section 2.1.3).

Please note, the front wheels and camera are controlled by servos, and moving them by hand may cause them to break. **Do not attempt to move the front wheels and camera other than via software**. You are responsible for taking care of your car!

1.2 Raspberry Pi

Your car is equipped with a Raspberry Pi (RPi) 4. You can SSH into your RPi by opening a terminal on your machine and entering `ssh pi@<YOUR_RPI_IP>`, where `<YOUR_RPI_IP>` is the IP address of your RPi. Please note that due to University restrictions you can only SSH in while on campus, unless you use a VPN. The password is `rasp`.

1.2.1 Finding your IP address

You can connect to your RPi using one of two methods. The first is via the University WiFi network, and the second is by connecting directly to an access point on the RPi.

Method 1 (University WiFi network): The University WiFi IP address can be found by connecting the RPi to to an external monitor, mouse and keyboard using the USB ports. At a terminal type `ifconfig`. The IP address can be found under `wlan0`, e.g. `inet 10.16.183.173`.

Once you have the IP address, you can login remotely by e.g. SSH or VNC and can disconnect the external monitor, mouse and keyboard. These IP addresses aren't fixed, and can rotate periodically. At the time of setting up the cars the IP addresses were:

Table 1: Initial IP addresses of the cars.

Car	IP
Pi 1	10.16.187.115
Pi 2	10.16.180.21
Pi 3	10.16.183.173
Pi 4	10.16.177.99

Table 2: RPi wireless access points and passwords.

Car	Network	Password	IP
Pi 1	raspi1	backprop	192.168.50.1
Pi 2	raspi2	backprop	192.168.50.1
Pi 3	raspi3	backprop	192.168.50.1
Pi 4	raspi4	backprop	192.168.50.1

Method 2 (Connect directly to the RPi): Alternatively, you can connect to the wireless network of the RPi. The wireless access points and passwords are:

Once you have logged in to the wireless access point, the IP address is 192.168.50.1. This is fixed and the same for each car. **This method doesn't require going though the University network, so will work even when the University network is unavailable. You do, however, have to be in range of the RPi.**

1.3 Car Sharing

4 cars will be shared between groups, so please be considerate in your use. The SD card is specific to each car, so please ensure the car is re-calibrated when you use it. **After use please make sure any training data you have collected is copied to another device and deleted, and remember to uninstall your self-driving package (see below), otherwise the next group will have access to your code!**

1.4 Tracks

We have 6 tracks in total, these are stored next to the filing cabinet.

1.5 Training Machines

As well as Google Colab and any local resources you may have, each group will have access to either the MLiS1 or MLiS2 machines (each with 2 2080Ti GPUs) to perform training. These are accessible by ssh'ing into the machine, by typing `ssh username@mlis1.nottingham.ac.uk` or `ssh username@mlis2.nottingham.ac.uk`, where `username` is your University username. You should use your University password.

In order to install custom packages on your machine, you will need to set up a conda environment. To install conda, type the following command,

```
bash /shared/Anaconda3-2019.10-Linux-x86_64.sh
```

Once installed, you will need to add a start up script,
`echo . ~/.bashrc >> .profile`
Lastly to create your conda environment use,
`conda create --name my_env python=3.9`
where `my_env` is the name of the environment. For more information on the usage of conda see: `https://docs.conda.io/projects/conda/en/latest/user-guide/cheatsheet.html`

1.6 TPU

The Coral Edge Tensor Processing Unit (TPU) is a custom device to run machine learning models for edge computing. The machine learning runtime used to execute models on the Edge TPU is based on TensorFlow Lite. It is only capable of accelerating forward-pass operations, which means it's primarily useful for performing inferences.

At present we have a single Edge TPU (although will be purchasing another unit). It plugs into the USB port on your RPi. To make use of the Edge TPU, your models should be trained on another machine and converted to tensorflow lite. Not all operations are comptatible with tensorflow lite, so it's important to bare this in mind when coding your model. Please see the link in the resources below for more details.

2 Software

Your RPi is setup with almost all of the software you will need – you will develop the self-driving software.

2.1 Remote Control Package

The remote control package is used for manually driving the car, recording video and actions while driving, and starting and stopping the self-driving.

2.1.1 Code Repository

The remote control repository can be found at `https://github.com/adammoss/SunFounder_PiCar-V` and is located on your RPi at `/home/pi/SunFounder_PiCar-V/remote_control`. This has been cloned and adapted from SunFounder for this project by the module conveners. You should not modify the master branch of this code, as this can cause merge conflicts if we update it. Instead, you will develop an **installable package** to drive the car using deep learning, according to the API specifications below, and only set environment variables where needed.

If major changes are made we will inform you, and these can be merged by opening a terminal and entering `git pull origin master` within the repository. If you would like to add features that you think would be beneficial, this should be done by submitting a **pull request**. An example would be an improved UI to manually control the car. If approved, we can then merge this into the master branch of the repository and everyone will benefit.

2.1.2 Setting up remote control

Remote control will automatically load when you boot your RPi4. In case you need to run it manually, login to your RPi and enter
`cd /home/pi/SunFounder_PiCar-V/remote_control` at a terminal. You can start remote control by running the `./start` shell script. Please note this requires terminating any existing processes using the `pkill -f runserver` command.

Remote control will start a webserver, which you can access from a browser at `http://<YOUR_RPI_IP>:8000/`. Note the port 8000, that due to University restrictions you can only access this while connected to the University WiFi if using method 1 for connecting to your RPi. This should bring up the screen shown in Fig. 1. If your camera is working correctly, it should show an image (this image is a snapshot and not a stream).

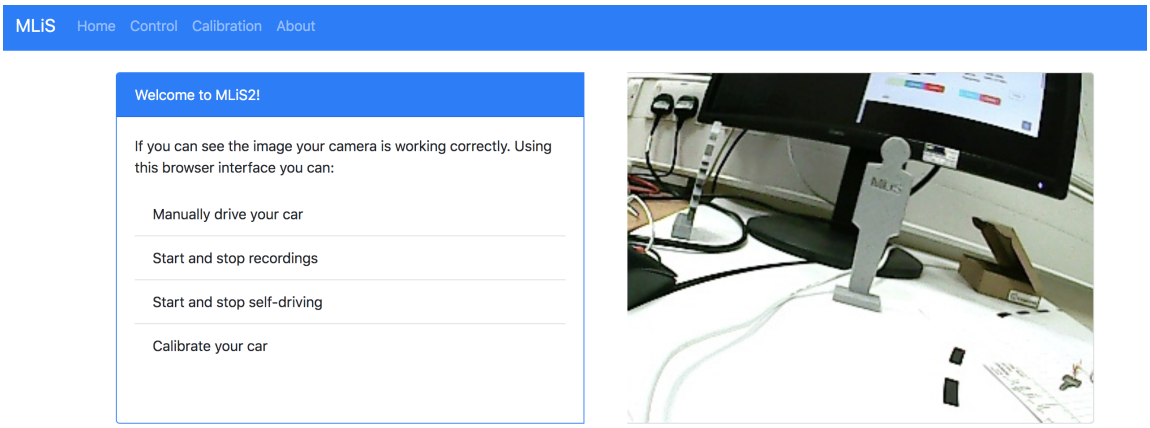

Figure 1: Home screen.

The car can be controlled by clicking on the control option in the top menu. This should bring up the screen shown in Fig. 2. Manual control of the car can be achieved by selecting the speed, then using the slider to select the steering angle.

You can record your driving by toggling the record driving switch. This will output frames at a specified resolution (`CAPTURE_WIDTH` and `CAPTURE_HEIGHT`), time interval (`RECORD_TIME_DELAY_SECONDS`) and location (`RECORD_DIR`), given in the `settings.py` file at
`/home/pi/SunFounder_PiCar-V/remote_control/remote_control/settings.py`. These options should be changed by **defining environment variables**, and you should not need to modify code in the repository (see below).

Recording can be stopped by selecting the stop option. We have added a condition to limit the size of the recording directory to less than 1 GB, in case you forget to stop recording. This is enough space for approximately 8000 images at 320×240 resolution. You should copy across images to another machine to clear space on your RPi. Files are named in the format `<TIMESTAMP>_<STEERING_ANGLE>_<SPEED>`.

2.1.3 Calibration

The car and camera can be calibrated (i.e. centered) by clicking on the calibration iterm icon in menu. This should bring up the screen shown in Fig. 4. The current configuration is shown in the table. Adjustments can be made by clicking on the

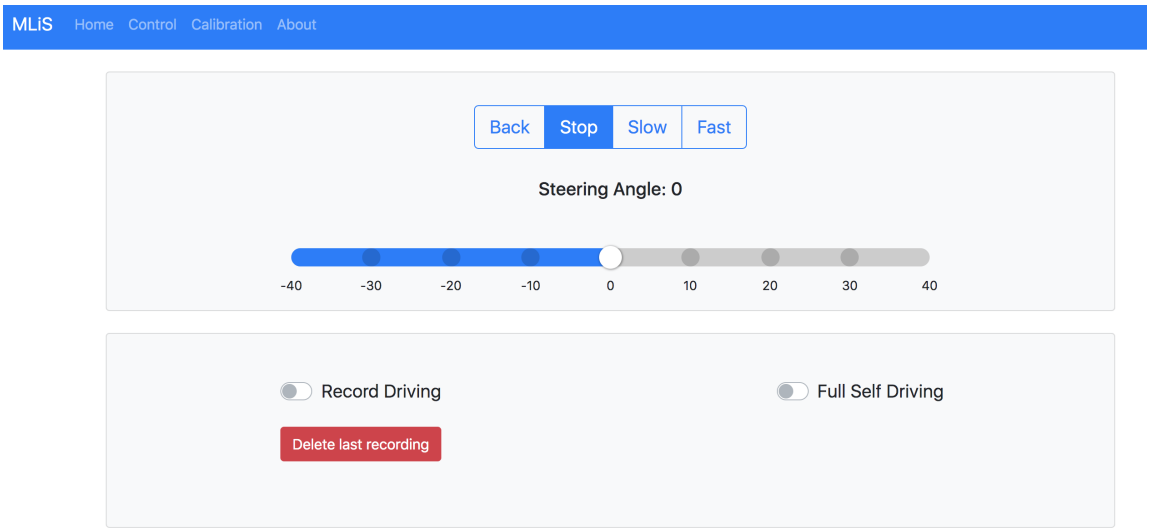

Figure 2: Control screen.

```
# PiCar

CONFIG_FILE = BASE_DIR + '/remote_control/driver/config'
RECORD_DIR = os.environ.get('RECORD_DIR', BASE_DIR + '/capture')
RECORD_TIME_DELAY_SECONDS = int(os.environ.get('RECORD_TIME_DELAY_SECONDS', 1))
CAPTURE_WIDTH = int(os.environ.get('CAPTURE_WIDTH', 320))
CAPTURE_HEIGHT = int(os.environ.get('CAPTURE_HEIGHT', 240))
```

Figure 3: Settings.

buttons. Note that the adjustments to the camera and front wheels are quite small, and you may need to click them multiple times. Changing the back wheels will have no observable affect, as it sets the direction they travel in. Click OK to save these settings to the `CONFIG_FILE` file of `settings.py`, so each time you start your car it will begin with the same configuration. The new settings will be reflected in the current configuration table.

2.1.4 Autopilot (Full Self Driving)

A skeleton installable package has been created for you are `https://github.com/adammoss/autopilot`, from which you can develop your code. Please refer to the github readme for instructions.

3 Training

3.1 Tracks

We have 3 tracks on which to train your cars, shown in Fig. 5.

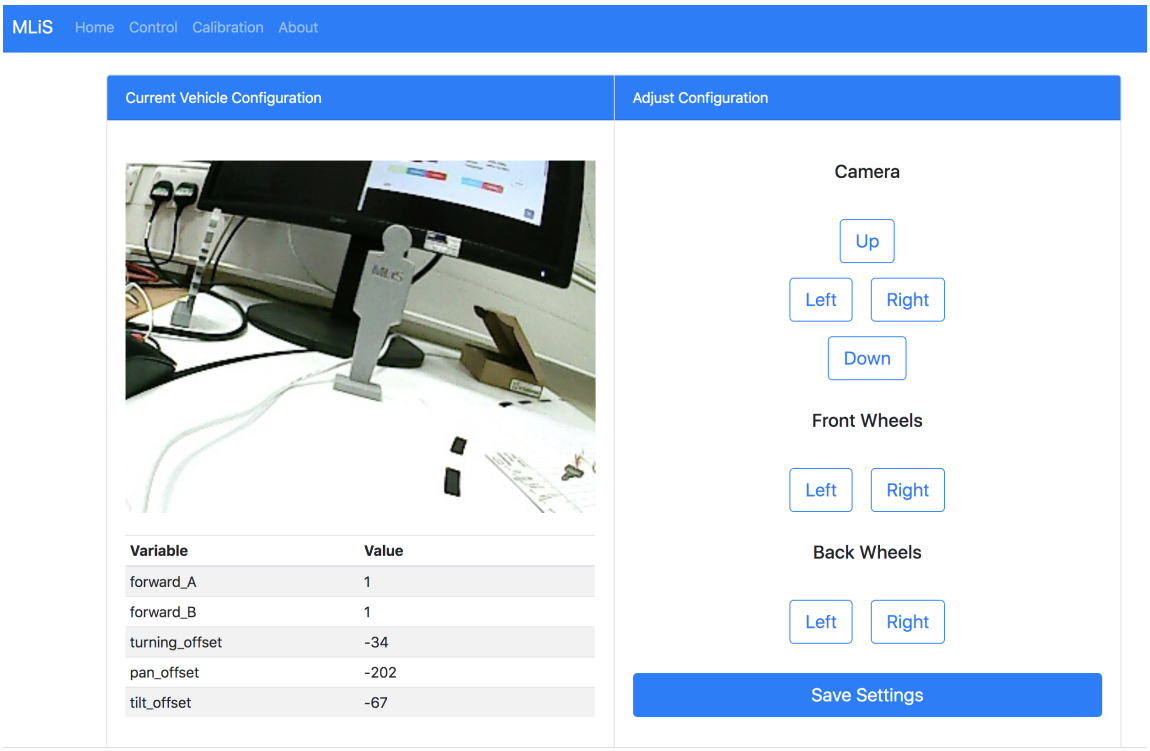

Figure 4: Calibration screen. Note this image is a snapshot when the page loads and not a live stream.

3.2 Driving Scenarios

Please refer to the MLiS2 performance marking document for details of the driving scenarios and scoring.

Resources:

You may find the following resources helpful in designing your own model,

- `https://towardsdatascience.com/deeppicar-part-1-102e03c83f2c`
- `https://coral.ai/docs/edgetpu/models-intro/`

Acknowledgements

Special thanks to Simon Dye for assembling the cars, modifying the camera position and 3D printing the objects. Without him this project wouldn't be possible.

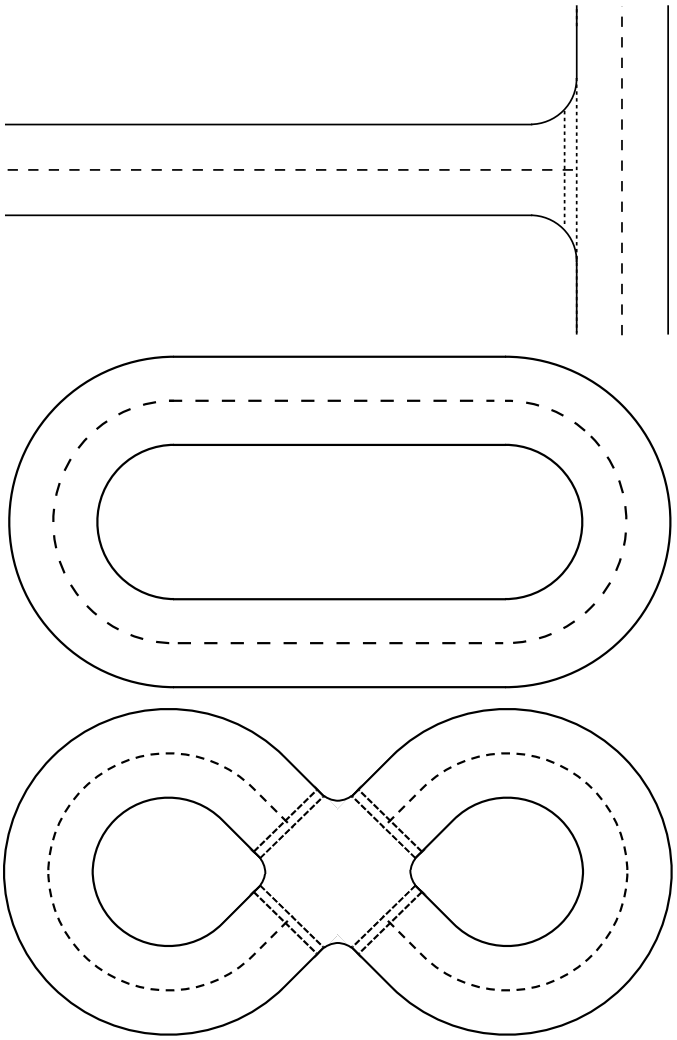

Figure 5: (Top) T-junction tracks (Middle) Oval Track (Bottom) Figure-of-eight track.