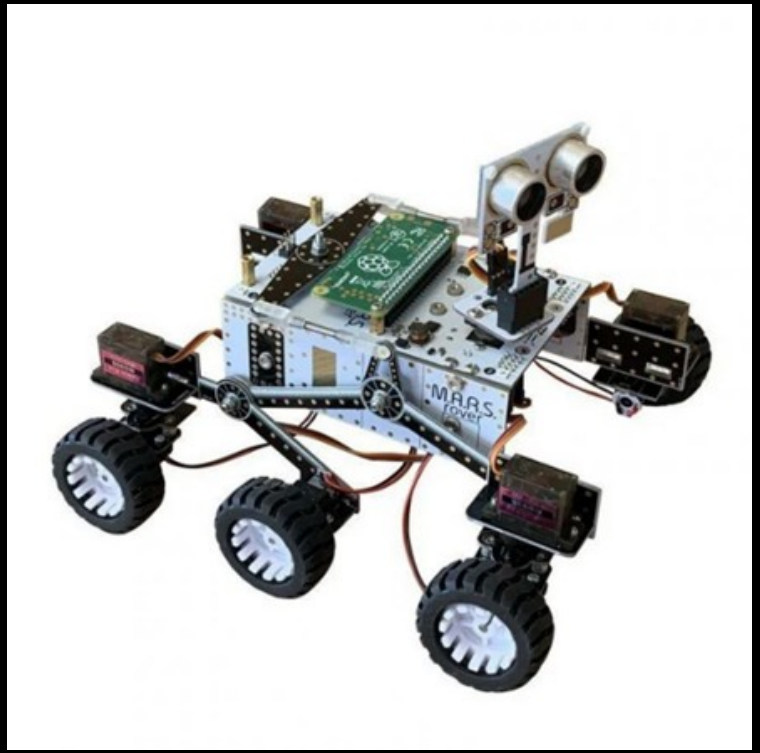
Adept x SUMO Semester 1 Hackathon

Overview

- The hackathon will require you to program and interface with a Raspberry Pi Zero W and replica Mars rover system to navigate through two stages of Mars-like environments.
- All code must be programmed in Python and teams are free to use the included rover.py library as well as any other libraries required.
- Your submission will be two .py files run separately for each stage.



Stages

1. Predetermined Route

- You’re given a predetermined route to navigate the rover through
- You will be given the exact dimensions of the route.
- All edges of the route will have walls to allow the sonar to detect distances.
- There will be a large time penalty for collisions so accuracy is key.
- There will also be a small time penalty for each user input. Minimum user input is desired but manual inputs are also recommended.

2. Unseen Environment

- The second stage will involve testing your rovers in an unseen environment.
- You’ll encounter new obstacles and terrain.
- This task is more focussed on autonomy so collisions will have a smaller time penalty while user input will incur a larger time penalty.

Note: We are aiming for full autonomy through the course to get maximum points but it is recommended that user control is allowed as a failsafe.

Scoring

- The two stages will be scored based on the time of completion.
- Speed is the goal here, the group with the fastest system will win, however time penalties will be applied for the following;
  - Collisions: Each collision with walls or obstacles
  - Excess User Input: The aim is for rover autonomy so user inputs after the start will incur time penalties.

- The time penalties for any of these errors will be dependent on the stage and will be outlined in the stage descriptions of the brief.

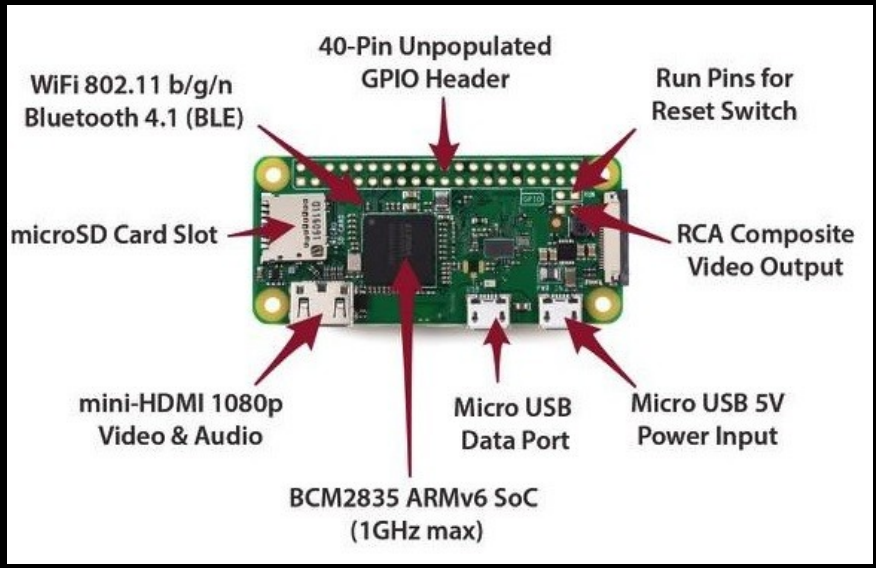
Hardware available on the Rovers

- Raspberry Pi Zero W
- 6 drive system motors (3 controlled for left and 3 for right)
- 5 servo motors
  - 4 servos on the 4 corner wheels to allow for counter phase steering
  - 1 servo connected to the sonar sensor
- Sonar
- Camera module (can be connected if required)

Core System Architecture: Microcontrollers

A microcontroller contains a CPU which executes code stored on a generally small amount of rewritable ‘flash’ memory. The code on this memory can be updated or replaced by ‘programming’.

Code written for a microcontroller will generally use the onboard analog and digital input/output pins to interface with external hardware such as motors or sensors.



Interfacing with the Raspberry Pi

1. SSH

SSH, also known as Secure Shell or Secure Socket Shell, is a network protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network.

When the Pi is on run the following line from the terminal on a device connected to the same network as the pi, replace the numbers with the appropriate IP:
ssh adept@**123.456.78.90**
Enter the password **adept** when prompted.
This will enable users to directly connect to the command line of the raspberry pi. From the command line, bash commands can be used to navigate directories, run and lightly edit files. **This is by far the fastest way to interface with the board.**

2. VNC

VNC or Virtual network Computing is a screen sharing system that allows users to remotely control another computer. This means that a computer’s screen, keyboard, and mouse can be used from a distance by a remote user from a secondary device as though they were sitting right in front of it.

Download VNC viewer on your PC from here:

https://www.realvnc.com/en/connect/download/viewer/

You will be required to create an account then you can enter the Raspberry Pis IP address to connect. The username and password are both **adept**.

This will enable users to connect to the Raspberry Pi OS desktop environment and navigate through the GUI. **This is a good option for people who aren't comfortable with only using a command line interface.**

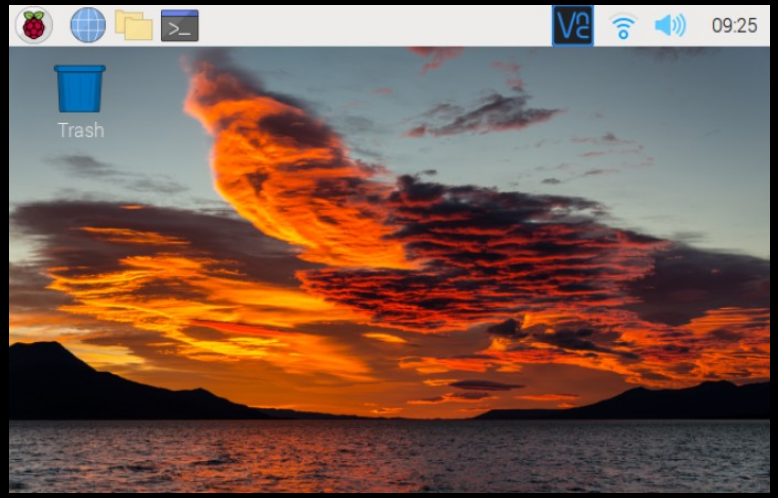
3. Direct

The Raspberry Pi can be interfaced directly through a mouse, keyboard and monitor.

Connect a monitor and keyboard/mouse directly to the board using the adaptors. Only recommended if you are experiencing network problems. This may be unstable if the board is powered by the rover batteries, so only run this when it is USB-powered. **Not recommended if either of the other two options are available.**

Navigating the Linux Terminal

- Raspberry Pi is run on an Arm based processor and runs linux
- The Raspberry Pi OS has a desktop environment that you'll be familiar with
- Navigating the console is very important especially for SSH connection



Raspberry Pi OS Desktop Environment

```
Linux adept1 5.15.32-v7+ #1538 SMP Thu Mar 31 19:38:48 BST 2022 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 13 09:24:33 2022
adept@adept1:~ $
```

SSH Console Terminal

Basic Bash Commands

Change directory

Changes the current directory of the terminal

```
cd <directory>
cd   (move back to previous directory)
```

List Files

Lists the files in the current directory

```
ls
```

Delete file /directory

```
rm <filename>
rm -r <directory>
```

Run code

```
python <filename>.py
```

Basic Bash Commands

Edit text files

`nano <filename>`

Find IP of Device

Prints the IP address of the PI (most likely not needed)

`Hostname -I`

Transfer files

Transfers a files or files to the Pi

`scp <localfilepath> adept@123.456.78.90:marsrover/`

Transfer directory

Transfers entire directory to the Pi

`scp -r <localdirectorypath> adept@123.456.78.90:marsrover/`

Programming for the Rover

- All rover programs are written in python
- The rover.py library has the main functions required to interface with the hardware.
- You are also given the example programs that you can base your own code off
  - calibrateServos.py - sets servo offsets
  - motorTest.py - tests which servo is connected to which port
  - driveRover.py - allows you to manually control the rover
  - ledTest.py - cycles the rover LEDs through various colours
  - sonarTest.py - outputs the distance in front of the sonar

Any Questions?