**CPEN 291   
Project 1 Report**

**A. Group info**

Lab section: L2A (*L2A or L2B)* Group #: A-G10 (*Example: A-G1*)   
Group’s Lab Bench #s: 10 and 15 (*Example: 1 and 2*)   
Student names:

|  |  |
| --- | --- |
| Justin Hua | Steven Huang |
| Eric Lyu | Nicholas Ng |
| Angus Wu | Winston Wu |

All the underlined text in this template are for your info. You should remove any text that is underlined before submission to Canvas.

Please keep the format of the report as is (e.g. do not omit any section, or change the font size or margins).

In Sections B and C of the report:

Explain the design and implementation procedures, and thoroughly provide documentation for the circuitry and software. During the project, you must have selected any method over another for some steps, describe the alternative (e.g. the second best) you considered. Include block diagrams or drawing to identify the main components and their interactions.

* You may include code segments in this part of the report only whenever needed for the explanations of the software design and approach. Your code must include comment statements, so do not repeat what is already included in the comment statements. As usual you will need to submit the complete code file separately, and also to include the complete code as an appendix to this report. The code must be readable in the first place and include sufficient comments (per code segment and per line, when needed) for documentation.

**B. Technical documentation for the main functionality**

Fully document your design and implementation for the main functionality here. In particular explain:

* The hardware including the circuit for the reflective optical sensors, and why you chose this configuration (i.e. the number of sensors you used and the way you arranged them relative to each other)
* The algorithm for the line tracking functionality
* The headless Pi use, implementation, and challenges
* Battery-operated robot implementation and challenges

**Hardware: 3 reflective optical sensors** – arranged in a row, 1 cm from each other. The first sensor was used to determine if the robot should turn left, the middle sensor for going straight and the third sensor for going right. We arranged the sensors in this configuration to account for turns and curves in the track.

**Raspberry Pi Camera** – Mounted on the front of the robot, giving us an optical view of where the robot was headed. Also used for one of our additional functionalities: streaming the footage to a website/server. Besides that, we also use camera to capture multiple pictures at a time whose resolutions are set 128x128 in order to fit into our LCD screen.

**TFT LCD –** Mounted on the back of the robot to allow for easy viewing. The real-time “video” which is implemented by using multiple pictures was displayed on the TFT LCD in order to help users identify the situation in front of the robot. For better usage of our website, we also included using a stream(threading) to create a concurrent program to display updated stuff on LCD while other functionalities can still work at the same time.

**The Algorithm –** we implemented the algorithm to check whether the line is either straight or turning left or right. To detect if the line was straight, the code just checked whether the middle reflective optical sensor was on and the other two sensors were off. For left, the code checked whether the middle and left sensors were on or just the left sensor was on and turned left slightly. Similarly, for turning right. The goal for the algorithm is to re-center the robot so that only the middle sensor is reading on and the others are off.

(need more information on the algorithm like the PID and stuff like that)

**Headless Pi use –** Using the VNC Viewer app we were able to access the Raspberry Pi without connecting to it physically. Some challenges include it not working at first and having to manually reconfigure the Raspberry Pi’s desktop resolution to 1920 x 1080 instead of auto. Using the Pi in headless mode allowed for us to mount the Pi onto the robot without it having a power cable attached to the wall, making the robot fully portable.

**Battery-operated robot –** we implemented the battery-operated robot using a portable power bank to power the Raspberry Pi and the 5 AA batteries for the motors. The challenges that came with using AA batteries was that the dc motors are super power hungry as well as decently heavy at the same time. And so, the way to combat this was to have batteries for testing and batteries for the actual demo. Challenges related to using the power bank was that there is a minimum current requirement for the raspberry pi to function properly and so we needed to find a power bank with the right specification. The caveat with using one that satisfied the specification was that it was heavy and the motors would have to carry the extra weight of the power bank.

**C. Technical documentation for the additional functionality**

Fully document your design and implementation for the additional functionality. In particular explain:

* What the additional functionalities are
* Include the list of the additional components you used
* How camera is used as a part of an additional feature
* The hardware implementation
* The software implementation

**Website/Server** – HTTP server that allows the user to input commands to the robot through the internet. Footage of the robot’s journey, recorded using the Raspberry Pi Camera, is also streamed to the server. The server is implemented using the HTTPServer class found in Python’s standard library. And is hosted on a local host.

The server class contains four main methods: do\_HEAD, \_redirect, do\_GET, and do\_POST. do\_HEAD sets up the webpage with the html contents (html code embedded as string in do\_GET). The host address is the Raspberry Pi’s IP address. The port number is set to 8000. The user can connect to the webpage using the host address and port number. For us, the address to connect is https://137.82.226.231:8000. \_redirect is used for refreshing the page back to the main page after each request from the user is handled. do\_GET sets up the website with several form submission buttons (“Forward”, “Backward”, “Left”, “Right”, and “Stop”) and an info about the current CPU temperature of the Raspberry Pi. When the user clicks on the buttons, a request is sent to the server, the webpage is implicitly redirected to a specific subpage. For example, when the user clicks on the “Forward” button, the page is redirected to https://137.82.226.231:8000/Forward. The do\_POST can then extract the request “Forward” from the subpage link and calls the specific function (moveForward(speed)) in (motorFunctions) to control the robot, thus handling the request. Then, the do\_POST method calls \_redirect to go back to the main webpage to wait for the next user input.

**Camera:**

In this project, we projected the screen of camera on to the LCD, so as the robot is moving, LCD will display the camera screen with some delay. We first save the pictures taken from the camera to local desktop, then we let the LED to read the picture. We repeat this step continuously in a while loop with framerate set to 30, so that LED is like playing a video. Since LED needs to read the picture every time from the desktop, there will be a little be of delay between each picture. Also, since the LCD’s screen size is limited to 128\*128, we need to rescale the image taken from the camera to an appropriate size.

We simply place the camera onto the robot so that it can take pictures as it moves, and then the LED will display the pictures that’s taken.

**D. Test and evaluations**

Explain your evaluation and testing procedures for hardware and software. Please demonstrate systematic testing, debugging and continuous integration. Include the problems you have encountered and how you resolve them, as well as best practices you have incorporated.

**Robot-** We made many test tracks using electrical tape including: a circle, a figure-8, a curvy track, a right-angled track, a straight-line and a track with crosses. We first tested if the robot would continue forward by itself using the straight-line track. Then we added crosses and gaps to ensure that the robot would keep moving forward. For testing if the robot would turn left and right, we started with low-angled turns and then moved to the circle track to test a higher-angled turn. Once we made sure that worked, we tested using the figure-8 track which combined curves, circles, crosses and gaps.

**Motor direction**-We found that the back part of the robot will be the better place to put the optical sensors. However, by doing that all the movements were reversed. We simply switched the power and ground wire for M2, then everything works fine.

**Website**-We firstly did not include the CPU temperature in the webpage. However, we found that the webpage cannot be correctly shown without the instruction: self.wfile.write(html.format(temp[5:]).encode("utf-8")). So, we added the CPU temperature to trigger the instruction to fix the problem. It also becomes an additional feature. We can now monitor the temperature of the Raspberry pi remotely.

**E. Conclusions and Reflections**

Reflect and conclude on the lessons, tricks or interesting concepts you have learned during the project.

Also reflect on other aspects such as team work, project management, time management, ...

**F. References and bibliography**

Provide any relevant references.

Also include the list and description of the files submitted for this lab (including code and Fritzing breadboard view)

Adafruit Motor Hat

* <https://learn.adafruit.com/adafruit-dc-and-stepper-motor-hat-for-raspberry-pi/overview>

HTTP Server

* <https://www.afternerd.com/blog/python-http-server/>
* <https://docs.python.org/3/library/http.server.html>

**Appendix A – Robot pictures**

Include pictures of your robot here. The pictures should clearly show the robot as a whole, as well as all electronics, wiring and parts. Include photos taken from the top, and from the sides. Show the location/installation of circuits and components as clearly as possible.

**Appendix B - Code**

Include the complete Python code with comment statements. This code must be the same code as the files you demo and submit. Clearly identify the portion of the code for the main functionality and the Additional functionality.

The code must be readable, with proper indentation, syntax highlighting (that is, copy with colour coding), and on white background. The code must be in text (that is, absolutely no snapshots of the code).

**Appendix C - Fritzing**

Include the snapshot of your fritzing breadboard view. Include as many as you have, but clearly describe which is which. This is in addition to the fritzing file that you submit to the Canvas.

**Appendix D - GitHub**

Every group member must have reasonably and equally contributed to the project github repository. If that is not the case for any member and there is a valid reason as to why, please include an explanation here.

https://github.com/CPEN-291/P1\_A\_G10

**Appendix E – Complete Component list**

Include the list of all the components used for the project.

If you have used any component you have purchased on own, include full info, a link to datasheet, and cost.

**Appendix E – Complete Component list**

1Raspberry Pi 4

1DFRobot 2WD Mobile Platform

1Adafruit 1.44’’ color TFT LCD

1numeric matrix keypad

**Appendix F – Answer the following questions:**

Q1 – Teamwork: Explain in details the methods your group has used to communicate effectively among team members.

We will first assign different job to each member, then we use github to update and share our code. Also, we have our Facebook group chat to communicate and to solve the problems together.

Q2 – Design Process for the additional functionalities: Describe clearly the process you used for the following design aspects of your own additional functionalities. Please spend time to carefully answer each of them.

1. **Use of process**: Describe your approach to adapt and apply a general design process for any additional feature. What was your approach?

We were thinking to use MCP3008 for our reflective light sensor, because it can give us a more accurate result other than 0 and 1. Also, we keep using the LCD to display image. This time, we added more feature to it. We try to project the camera screen on the LCD. To implement this, we need to save the image we took from camera first and let the LCD read the image. If the process is fast enough, the LCD will show the screen of camera with some delay. Also, we have were thinking of using website to control our robot, such as moving forward, backward, stop etc.

1. **Constraint identification**: Explain the constraints that you must consider in the design of the additional functionalities.

To get the LCD working, we have to download some addition packages from the website to set it up properly, and this is time consuming. Also, when we try to create our own website to control the robot, we have to use HTML to create buttons and send request to the server. Again, this requires lots of searching and reading.

1. **Solution generation**: Explain at least two possible alternative additional features that your group rejected due to technical reasons and explain why.

We were thinking of using joystick to control the robot, but somehow, we used the website instead since it’s fancier. Also, we reject the ultrasonic sensor since it’s not helpful in terms of our design.

1. **Solution Assessment**: Explain how you tested and assessed the viability and then correctness of your group’s additional features.

We always test the code we wrote and see if it’s actually working before, we added it to the robot. Also, we always test how the code will affect the movement of the robot. If something goes wrong, we will always try to communicate and figure out a solution.

**Appendix G - Other**

Include any other relevant info that does not fit in any other section in the report.