

EECE8040 – Engineering Capstone Project

Robot Tank Quality Assurance and Design Evaluation

Group #1: LED Robotics

Group Members: Eduardo Campos, Leah Carinci, Daniela Diaz Tenorio

Due: April 27th/2021

## 1. Abstract/Summary

The focus of this project was the quality assurance and design evaluation of a remote-controlled robot tank. This project is a continuation of work completed by previous capstone groups. Existing bugs and design flaws were identified, revised, and fixed. All software developed for this project has been written in Python. The project’s sponsor, Robert Elder, has also aimed to keep the budget of this robot tank under \$100.00 CAD to build.

## 2. Acknowledgements

We would like to thank our project mentor Ralph Stacey for his assistance throughout this project. We would also like to thank our project sponsor Robert Elder for providing us with this opportunity.

## 3. Table of Contents

### Table of Contents

1. Abstract/Summary .....	2
2. Acknowledgements.....	2
3. Table of Contents.....	2
4. Introduction/Problem Statement .....	3
5. Proposed Solution.....	3
6. Background Theory .....	3
7. Methodology .....	4
7.1. Build Instructions .....	5
7.1.1. Soldering Components to the PCB .....	7
7.1.2. Connect the Raspberry Pi.....	8
7.1.3. Connect the L298N (Headers J2 and J3) .....	8
7.1.4. Connect the Camera Module .....	9
7.1.5. Connect the Servos to the PCB.....	10
7.1.6. Camera.....	10
7.1.7. Place the PCB on the Robot Chassis .....	11
7.1.8. Connect the Power Sources .....	11
7.2. OS/Software Instructions .....	12
7.2.1. Software Set Up .....	14
8. Results/Analysis.....	16

8.1. Software Changes .....	16
8.2. Hardware Changes .....	16
8.3. Quantitative Data.....	18
9. Cost Analysis .....	18
10. Recommendations/Improvements .....	18
11. Appendixes .....	19
12. Schematic.....	19
13. References .....	20

## 4. Introduction/Problem Statement

The original robot tank that was provided at the start of this project was robot tank version 1.0. This robot tank was functional and was able to be controlled as generally expected, but there were some bugs to be fixed and improvements to be made. One major issue with the original tank was that while the cameras were not in use, but the tank was fully powered and on, the servos would jitter. Another known issue was that the power bank for the tank motors would turn off if the tank was idle for a short time. There were also design labels on the schematic that were incorrect and some that were vaguely labelled, which made building the robot tanks a bit hard as we had to decipher what components were what.

## 5. Proposed Solution

The proposed solution for this project was to identify, document, and resolve existing bugs and design flaws in the original robot tank version 1.0. Our group planned to build two additional robot tanks with the provided materials and implement changes and improvements on these new builds. The hardware and software were improved simultaneously throughout the duration of this project. After assembling the new robot tanks, certain hardware fixes were implemented, and the code was updated to reflect these changes.

We also planned to compile all relevant documentation into this paper so finding information pertaining to this project is easier. Hardware and software configuration steps were designed to help users of this robot tank set the project up. Future improvements and/or functions the tank could be designed to have are also included.

## 6. Background Theory

In order to get the robot tank moving from a standstill position, a large amount of current is required. The power banks used for robot tank version 1.0 were not designed to provide this initial spike of current but instead provide a steady stream, which wasn't enough to move the tank. The approximate weight of robot tank version 1.0 with the two power banks is 1.6 lbs. With the added weight of the

power banks, it became too heavy to move the tank with not enough power. This meant we would have to improve the power to the tank motors.

## 7. Methodology

The start of this project focused on testing robot tank version 1.0 and compiling a bug list. We began this process by developing a test plan to perform on version 1.0 and the new builds that would be assembled. This test plan serves as the first checkpoint of this project so that the basic functionality of the robot tank can be verified. The test plan can be found in the document titled Test\_Plan.pdf. This allowed us to find bugs and determine how to fix them and optimize the next tank version that we would build. We then put together two new robot tanks and implemented some hardware changes to these that would improve the functionality of the tanks.

**Table #1: Bug Report**

ID	Problem Type	Problem Description	How Was it Detected?	Severity
1	Hardware	Servo's power supply is connected to the Raspberry Pi's PIN 4. In case of large angles of rotation without acceleration control (or if a servo gets stuck), it will drain a high current and will turn off the Raspberry Pi.	KiCad Schematic	Moderate
2	Software	The wire assembly and python code do not match with the PCB. For an unknown reason, the built tank has a wire connection that is different from the schematic design. It created confusion when assembling the other tanks.	Assessing the Python code and PCB schematic	Low
3	Hardware	Battery shutdown. This model of power bank has an internal sensor that shuts down the battery when the current is too low.	Testing the battery with other devices.	Low
4	Hardware	Tank losing power. While in motion, the tank seems weak and does not respond to the key commands well. The L298N has a jumper wire connection to power up its logic using the same source used to power up the motor. However, when the motor is starting to spin, it drains a lot of current which drops the voltage below the minimum for the L298N.	Assessing the L298N module	Low

## 7.1. Build Instructions

These build instructions are based on the following documents:

**Table #2: KiCad Files**

Document	Description	Version
Robot Tank.sch	Robot Tank Schematic	N/A
Robot Tank.kicad_pcb	PCB Drawing	N/A

The following components were used to build one robot tank. These components were provided by our sponsor, Robert Elder. Some component prices are not listed as they were a part of bulk orders or could be found on hand (ie. capacitors, resistors). Prices with a letter next to it indicate that multiple pieces were included in that overall price but were further broken down into the specific components. Note that these prices are subject to change as there are many different options and online retailers to choose from.

**Table #3: Components and Price List**

Quantity	Designator	Comments	Price (\$CAD)
1	PCB	Printed Circuit board (contains the components below)	5.40
2	C1, C3	10uF Capacitor (SMT)	N/A
2	C2, C4	100pF Capacitor (SMT)	N/A
2	R1, R2	2K Resistor (SMT)	N/A
2	LED1, LED2	LEDs	N/A
1	J1	Raspberry Pi 3 Connector	C
2	J2	L298N Control Signals	B
2	J3	L298N Power	B
1	J4	Servo Motor 1	A
1	J5	Servo Motor 2	A
1	J6	USB B Micro	N/A
1	Raspberry Pi	Raspberry Pi 3	47.79 (C)
1	Micro SD	Preferably class 10	10.00
1	L298N	DC Motors driver	2.21 (B)
<b>Power Components</b>			
5	AA batteries	7.5 V total	5.00
1	5x AA battery mount/case	Case to hold the batteries.	4.35
1	Raspberry Pi power cable	Micro USB to USB C cable	N/A
1	DC Motors power cable	Mini USB to USB C cable	N/A

1	Battery Pack	Can find from Walmart, Amazon, or other online retailers.	18.00
<b>Camera structure</b>			
2	Servo Motors SG90		A
1	Camera Tower Structure	Tower-Pro for SG90 9g.	9.49 (A)
1	Camera Mount	Printed with 3D printer. Find the files attached.	(3D printed)
1	Camera	Raspberry Pi 3 - Camera 1080p version2	7.98
1	Flexible Ribbon Cable	Ribbon cable for the camera. It is preferred to be 30 cm.	5.34
1	Robot Tank Chassis (SN900)		23.27
~20	Female to Female Jumpers		1.99
8	Spacers		(3D printed)
8	Screws		
<b>Total Price</b>			<b>140.82</b>

This is approximate pricing.

Prices of individual components that were provided for this project can be found from the following links:

<https://blog.robertelder.org/raspberry-pi-streaming-video-tank/>  
<https://blog.robertelder.org/cheaper-pcb-for-robot-tank/>  
<https://hardware.robertelder.ca/>

The following tools were used in the assembly of the robot tank:

**Table #4: Tools List**

Quantity	Name
1	Soldering Iron
1	Heat Gun
1	Solder Paste
1	Solder Wire
1	Tweezers
1	Double Sided Tape
1	Screwdriver

### 7.1.1. Soldering Components to the PCB

After gathering all the components, it is required to inspect the PCB and proceed with soldering. For this, a soldering iron and solder should be use for the 5 headers and 2 LEDs. The rest of the components are surface-mount technology and can be soldered using a heat gun and solder paste.

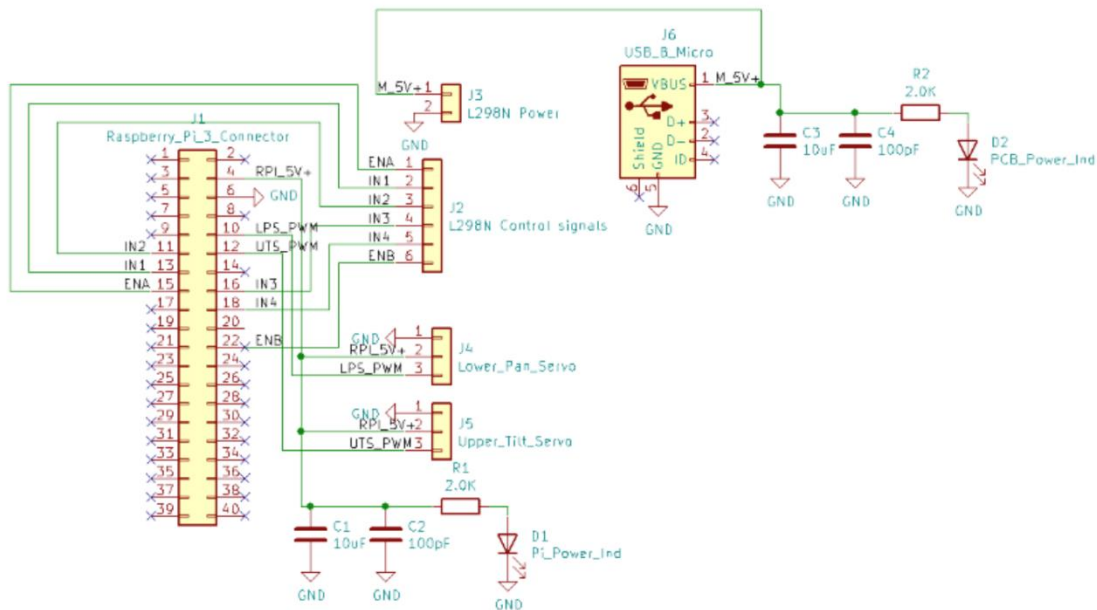


Figure 1: Revised schematics of robot tank version 2.0. Labels were updated to be more descriptive and several connections were changed.

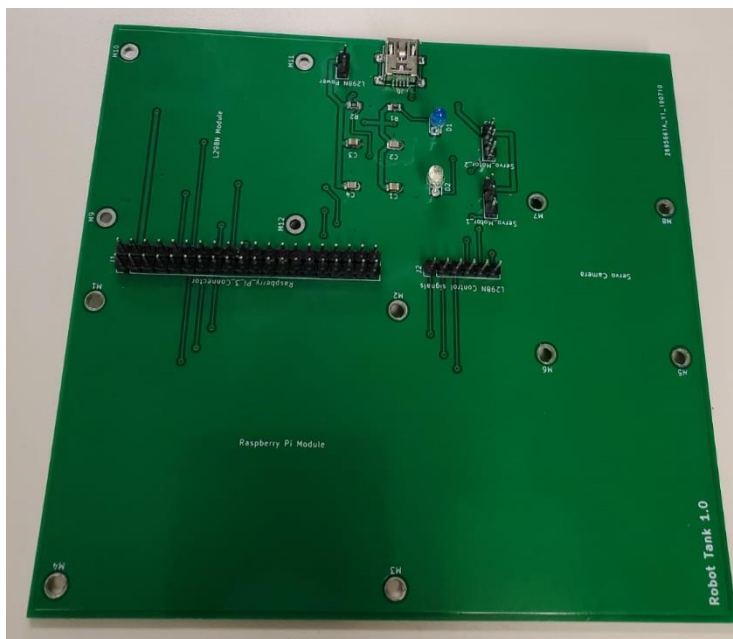


Figure 2: PCB with soldered components.

### 7.1.2. Connect the Raspberry Pi

Place the Raspberry Pi on top of the PCB and attach using the spacers, screws, and nuts.

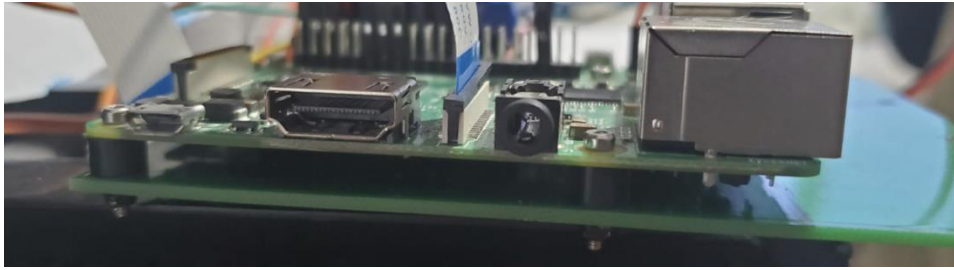


Figure 3: Side view look at the Raspberry Pi connected to the PCB.

Proceed to connect the header J1 to the Raspberry Pi pins.

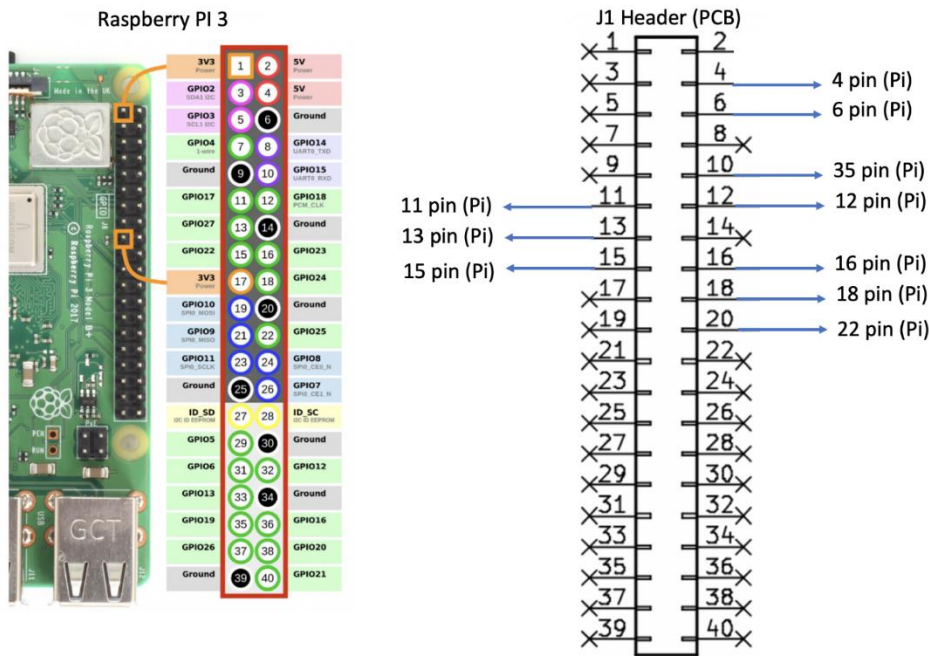


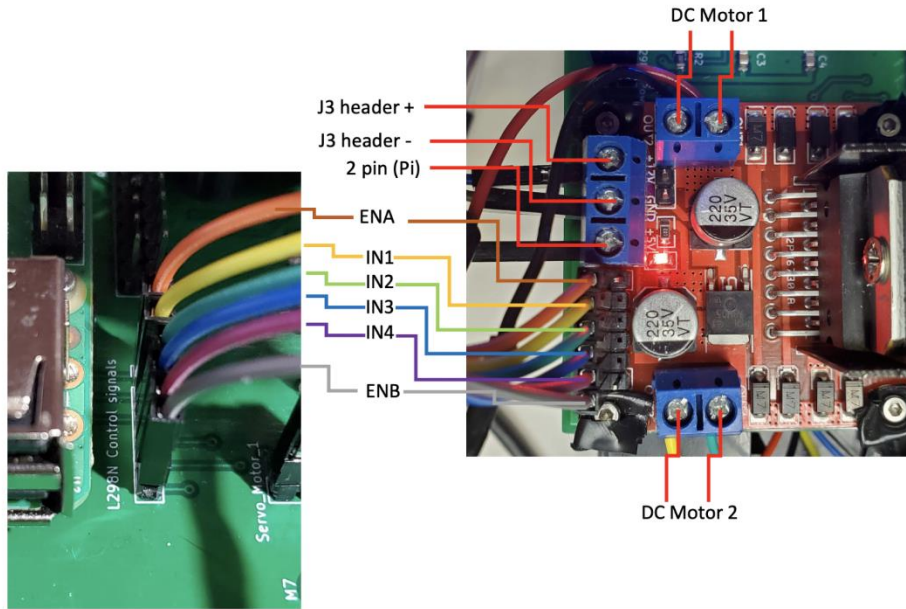
Figure 4: J1 pins and the Raspberry Pi pin configuration.

### 7.1.3. Connect the L298N (Headers J2 and J3)

Place the L298N driver on top of the PCB and attach it with screws, spacers, and nuts.

Connect the logic pins to the J2 (L298N control signals) and J3 headers. After this, proceed to connect the DC motors 1 and 2 to the driver pins.





**Figure 5:** L298N module connections and how they should correspond to the PCB. Connections for DC motors 1 and 2 are also shown.

#### 7.1.4. Connect the Camera Module

The first step is to assemble and attach the camera tower structure.



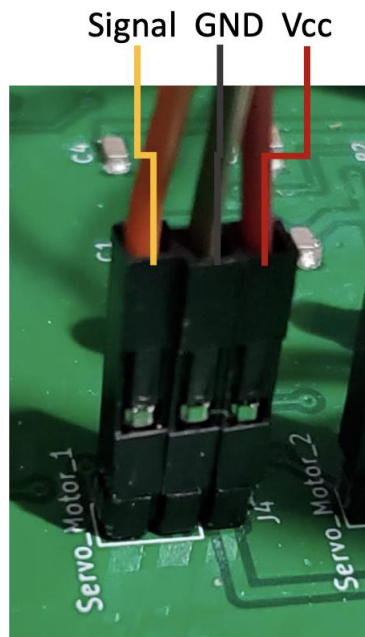
**Figure 6:** Camera mount structure with both the upper tilt and lower pan servos. (Adafruit, 2015)

The details for assembling it can be found here: <https://learn.adafruit.com/mini-pan-tilt-kit-assembly/getting-started>

### 7.1.5. Connect the Servos to the PCB

Connect the header J4 as the lower servo that is used to pan (Lower\_Pan\_Servo). The header J5 should be connected to the upper servo that is used to tilt (Upper\_Tilt\_Servo).

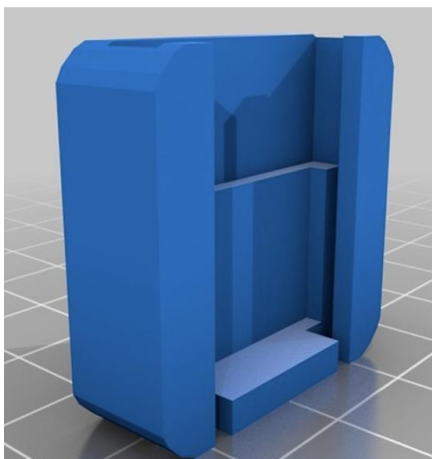
Make sure that the wires from both servo motors are connected in the order shown in the picture below.



*Figure 7: Servo connections to the PCB.*

### 7.1.6. Camera

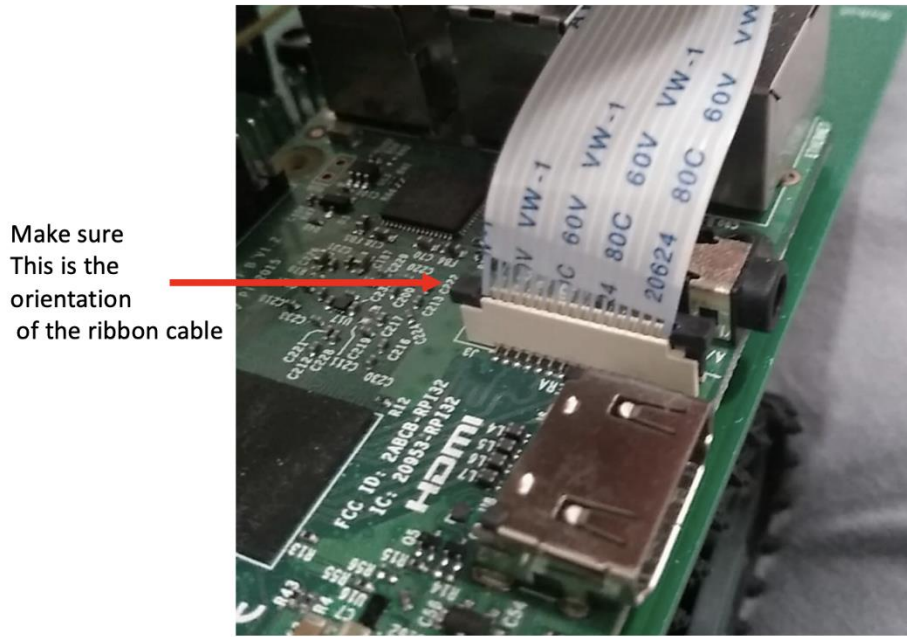
Insert the camera in the camera mount and attach it to the camera tower structure.



*Figure 8: Camera mount that holds the Raspberry Pi camera.*

The camera mount files can be downloaded from: <https://www.thingiverse.com/thing:3004766>

Connect the camera cable to the ribbon cable slot in the Raspberry Pi, making sure that the orientation of the cable is correct. Also, check that the length of the cable will let the structure move without pulling it while in motion. For this reason, we used a 30 cm ribbon cable.



**Figure 9:** Ribbon cable orientation for Raspberry Pi camera.

#### 7.1.7. Place the PCB on the Robot Chassis

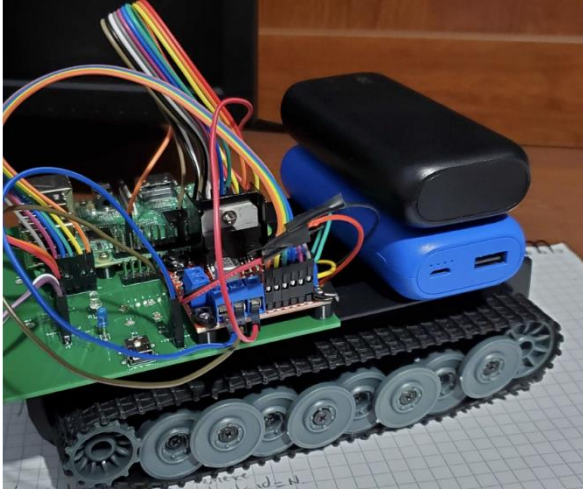
Attach the PCB to the front part of the tank chassis. You can use a double-sided tape. Make sure that there is enough contact surface to provide stability to the structure.

#### 7.1.8. Connect the Power Sources

##### 7.1.8.1 Power Banks (Option 1)

One power bank should be connected to the mini-USB port soldered on the PCB. Once this is plugged in, the D2 LED should light up indicating the DC motor power is on. The other power bank should be plugged into the Raspberry Pi using the micro-USB cable. Once this power bank is connected the D1 LED will light up to indicate power to the Pi.

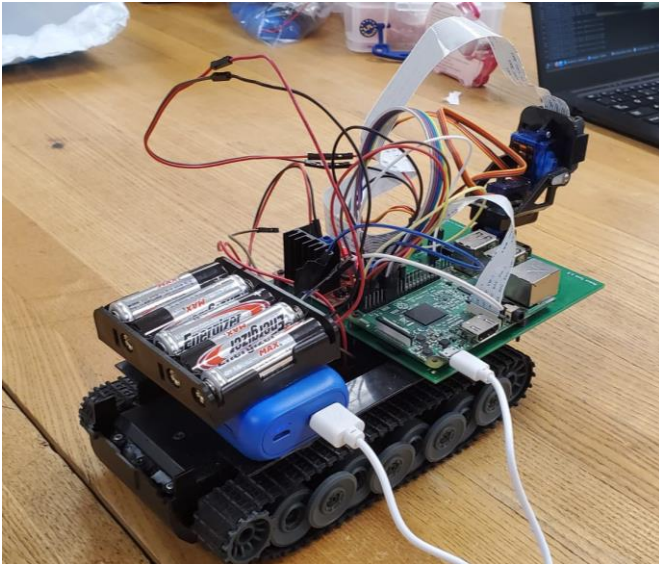
The 2 power banks can be stuck together and attached to the rear base of the chassis as is shown in the below picture.



**Figure 10:** Power banks sitting on the back of the robot tank chassis.

#### 7.1.8.2x AA Batteries + Power Bank (Option 2)

To install the AA batteries for the robot tank, it is required to use a 5xAA battery mount that can be connected directly to the DC motor's power. The mount can be attached on top of the battery tank used for the Raspberry Pi, as shown in the picture below.



**Figure 11:** One battery pack mount (5xAA batteries) to power the tank motors and one power bank for the Raspberry Pi.

## 7.2. OS/Software Instructions

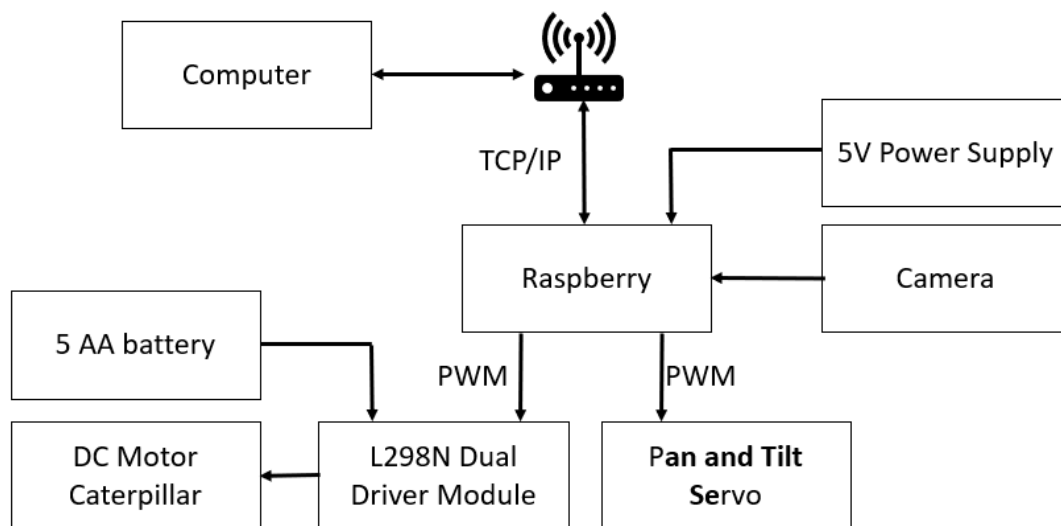
The software to control the tank and cameras was written in Python. To simplify the code, it was broken down into smaller files that were responsible for performing specific tasks. Tables 5 and 6 below show the files for each operating system and provide a short description of their functions.

**Table #5: Raspberry Pi Software Files**

Description	File
This library creates an interface with the Linux kernel to configure and send the command to the hardware.	PWM.py
This library provides a TPC/IP server which receives the JSON packages from the computer.	PyKeyUpKeyDown.py
Robot tank library class that controls each tank thread using a software PWM.	RobotTank.py
Main program which setup and integrates all library.	RunRaspberry.py

**Table #6: Computer Software Files**

Description	File
This program establishes the TCP/IP connection with the raspberry camera and makes the stream available for a program on the computer.	cam.py
It is the main program that reads the keyboard or joystick commands and sends them to the Raspberry Pi. It is also responsible for opening a window to show the raspberry video.	RunComputer.py



*Figure 12: Block diagram of the entire robot tank system. The computer and the Raspberry Pi communicate via TCP/IP protocol and the peripherals are connected as shown.*

### 7.2.1. Software Set Up

This section details the steps required to configure the system and install the software required to run the tank programs. The Raspberry Pi and host computer configuration are the two main sections.

#### 7.2.1.1. Raspberry Pi – Hardware PWM

The servos, which control the camera pan and tilt, require a PWM as an input signal. Thus, to produce a clear and steady PWM signal using the Raspberry Pi, the best option is to use the PWM employed for sound output. It requires changing the Raspberry Pi configuration to disable the audio output and release the resources for another purpose. Therefore, the following procedure must be executed to change how the Linux kernel uses the PWM peripheral. These instructions are based on the procedures provided by Jumpnow (2017).

1. Open the config.txt

```
sudo nano /boot/config.txt
```

2. Find the line below.

```
# Enable audio (loads snd_bcm2835)

dtparam=audio=on
```

3. Comment out the audio line by including the character #

```
# dtparam=audio=on
```

4. Include a new line at the end of the file.

```
dtoverlay=pwm-2chan
```

5. Save the file and reboot the raspberry.

```
sudo reboot
```

6. Run the command below to verify if the procedure was successful:

```
lsmod | grep pwm
```

7. It should produce the following answer:

```
pwm_bcm2835
```

#### 7.2.1.2. Raspberry Pi – Camera

To enable the camera, Raspberry Pi provides a program that can easily set it up. On the terminal, type the following command:

```
sudo raspi-config
```

Within the program, navigate to option 3 and select it:

3. Interface Options -> P1. Camera

After enabling the camera, you can exit the `raspi-config`. It will ask you if you want to reboot the Raspberry Pi, however, you should answer **no** to that question. Closing `raspi-config` and coming back



to the terminal, you have to verify if there is an error message. If there isn't any error message, you can reboot the Raspberry Pi by typing:

```
sudo reboot
```

### 7.2.1.3. Computer Setup – Linux

All the scripts to remotely control the tank were written in Python. Therefore, that script requires installing a basic Python interpreter for each operating system. Another requirement to control the tank is a program to receive and decode the video. For this portion of the project, *mplayer* was chosen as the program to handle that task.

On Linux, all software requirements can be installed by running the following command:

```
sudo apt-get install python3-pygame mplayer
```

### 7.2.1.4. Computer Setup – Windows

On Windows10, Python can easily be installed by calling the python3 on the command line terminal. If you don't know how to open a command-line terminal, please see the explanation given by Fisher (2020).

Calling the program in the command line will open another window showing the instructions to install the program. Just follow the instructions. After finishing the installation, open the terminal again and type the following commands:

```
pip install --upgrade pip
```

```
pip install pygame
```

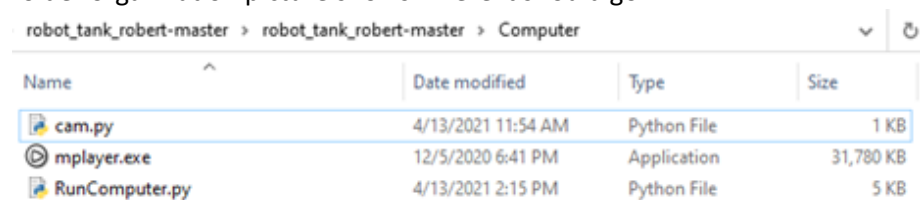
If you are using another Windows version, the Python interpreter can be downloaded from:

<https://www.python.org/downloads/>

Finally, we need to download the *mplayer* version for Windows from:

<http://mplayerwin.sourceforge.net/downloads.html>

Select the appropriate binary version for your Windows version. After downloading the *mplayer*, extract and copy the executable to the same folder where the Python program was placed. The below computer folder organization picture shows where it should go.



Name	Date modified	Type	Size
cam.py	4/13/2021 11:54 AM	Python File	1 KB
mplayer.exe	12/5/2020 6:41 PM	Application	31,780 KB
RunComputer.py	4/13/2021 2:15 PM	Python File	5 KB

Figure 13: Folder organization for the executable file.

### 7.2.1.5. Operating Instructions – Raspberry Pi

To launch the program, we must manually run the python script on the raspberry and computer.

- Connect to the tank using SSH
- Launch the program by calling:

```
python3 Runraspberrypi.py
```

### 7.2.1.6. Operating Instructions – Computer

On the computer side we need to call the Python script and provide the Raspberry Pi IP address:

```
python3 RunComputer.py <IP Address>
```

### 7.2.1.7. Keyboard Controls

**Table #7: Camera Controls**

Keyboard Input	Camera Function
I	Move camera down (tilt servo = upper)
K	Move camera up (tilt servo = upper)
J	Move camera left (pan servo = lower)
L	Move camera right (pan servo = lower)
O	Recenter camera
P	Print current duty values of each servo
Q	Quit the program

**Table #8: Tank Controls**

Keyboard Input	Tank Function
W	Move forward
S	Move backward
A	Move left
D	Move right
X	Stop

This program has also enabled a Playstation controller so that the robot tank can be controlled with two joysticks instead of keyboard keys.

## 8. Results/Analysis

### 8.1. Software Changes

A diagnostic code was created to ensure that all connections are correct and that the tank motors are moving as expected. This diagnostic code can be used as a test after a tank is built to ensure that it was assembled correctly. Several revisions of the main code were made to allow for both the tank and camera controls to be used from within the same video window. This program also checks if the user has a joystick plugged in to use instead of keyboard keys.

The video latency was decreased to improve streaming quality and time. The final software programs to run are RunComputer.py and RunRaspberry.py, each of which runs on their respective device.

### 8.2. Hardware Changes

As a result of this project, we have 3 functional robot tanks, working with the latest version of the software.



One of the hardware changes implemented was the removal of the jumper wire in the DC motors driver L289N. This jumper connects the logic voltage with the power supply. After removing the jumper, we had to connect a different wire for the 5V logic supply. For this we used the 5V provided by the Raspberry Pi.

The PIN-20 was initially defined as the output to enable the channel B on the L289N. However, this pin is a ground in the Raspberry Pi, so the enable signal was moved to PIN-22.

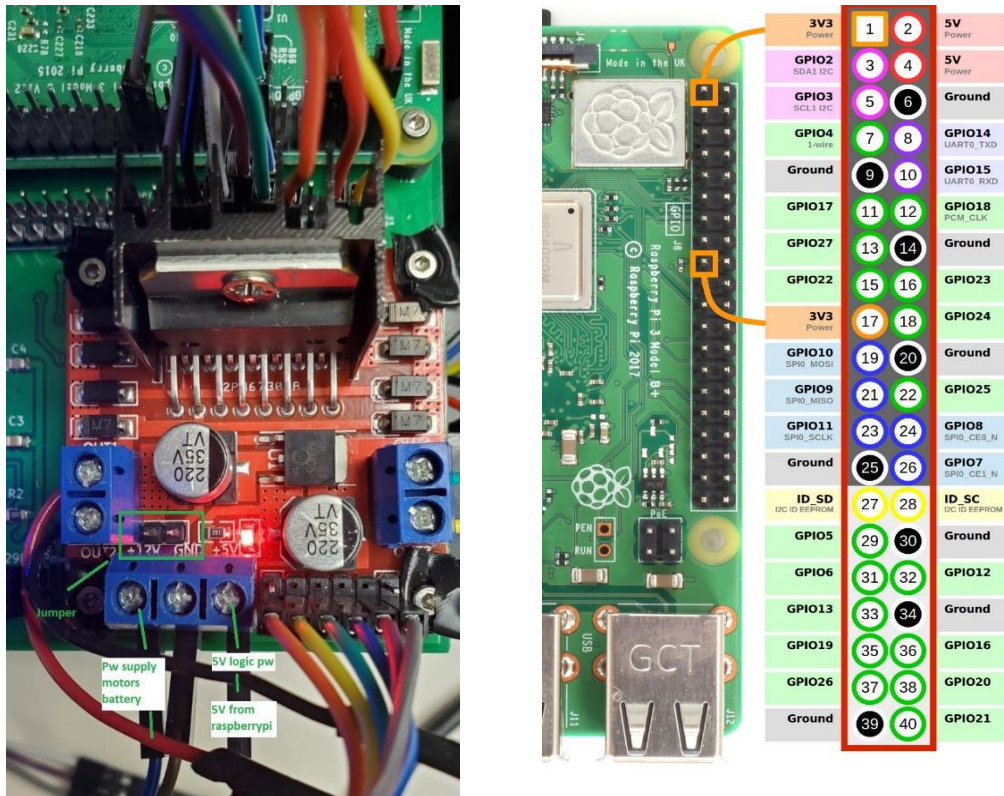


Figure 14: The image on the left shows the hardware changes to the L289N module. On the right is the Raspberry Pi 3 Model B pinout.

We changed the soft-PWM to hard-PWM to control the camera's servos motor. Soft-PWM does not provide enough accuracy to control the position which creates the jittering movement. This change requires to move the PIN 10 to PIN 35 for the lower pan servo.

The power bank for the DC motors was changed for AA batteries. For this change, the wire from the batteries was connected directly to the power supply pin (+12) in the L289N and the ground line of the PCB was connected using a jumper directly to the ground of the L289N. The batteries were placed inside the tank chassis.

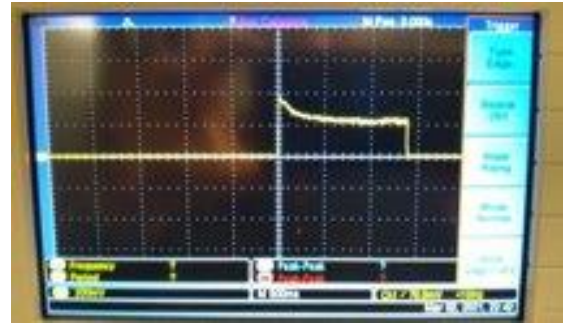
The schematic was also updated, which can be found in the schematic section on page 19. Labels were also revised and updated to have more descriptive names.

### 8.3. Quantitative Data

A ramp test was performed by increasing the angle (and slope) at which the robot tank would move up a metal ramp. We also performed a stall test to find the maximum current output that the DC motors would produce when moving at full speed against an object to simulate a stall. We found that the ramp test angle limit was 40 degrees and that the DC motors have a maximum current output of 1.6A.



*Figure 15:* Ramp test with a maximum slope of 40 degrees.



*Figure 16:* Maximum current output of the DC motors was 1.6A during the stall test.

## 9. Cost Analysis

The original robot tank version 1.0 cost \$132.45 CAD (Vemuri et al., 2018). The new robot tank version 2.0 costs \$140.82 CAD, which is broken down in table #3 on page 5. This cost could be further reduced by choosing a cheaper power bank and chassis as there are lots of options to choose from.

## 10. Recommendations/Improvements

Future work of the robot tank can include the implementation of an accelerometer and/or a GPS module. This would allow for more quantitative analysis to be made of the tank. The software could also be expanded to allow for different types of controllers, joysticks, etc. based on what the user has available to them. Another consideration for future builds is to have a built-in metal contact case to hold the AA batteries within the chassis.

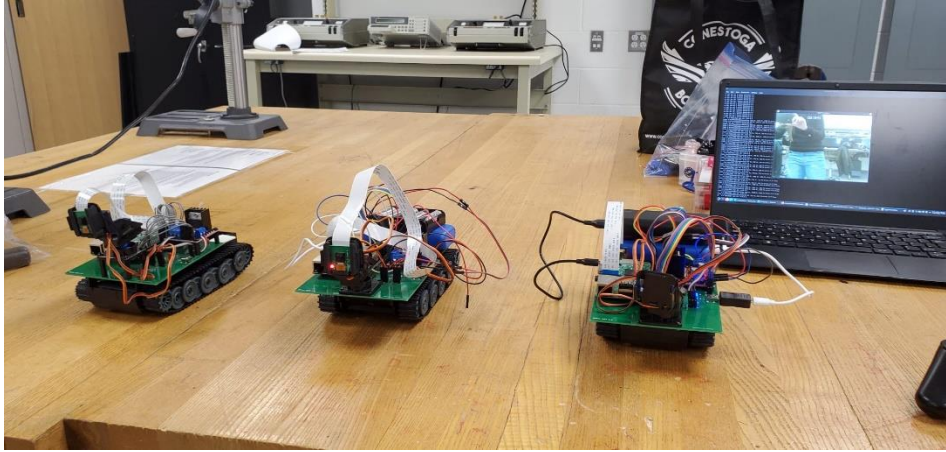


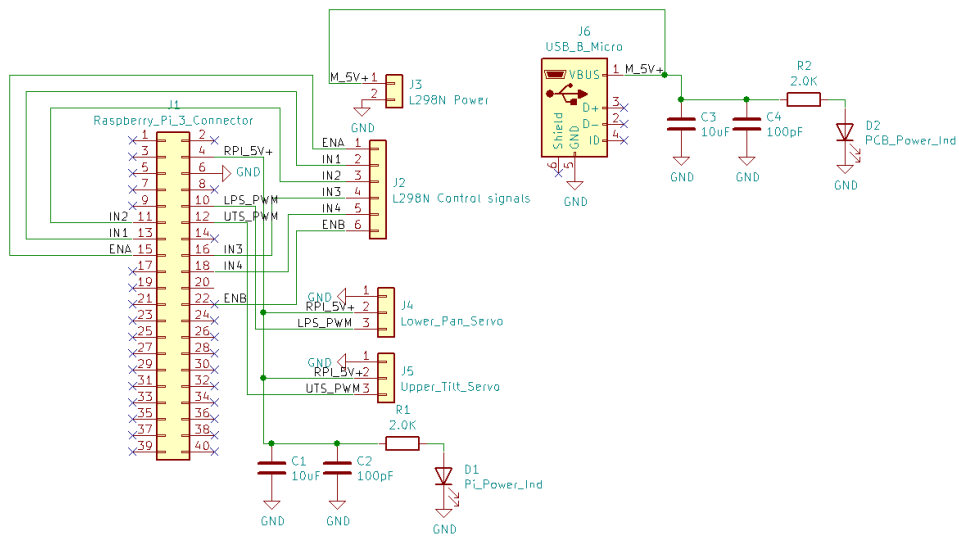
Figure 17: Robot Tank Version 1.0 on the left and two Robot Tanks Version 2.0.

## 11. Appendixes

All Python code for this project can be found at the following link:

[https://github.com/dandiaz10/robot\\_tank\\_capstone](https://github.com/dandiaz10/robot_tank_capstone)

## 12. Schematic



## 13. References

Campos, E. L., Carinci, L., & Tenorio, D. D., (2021). Board Bring-up Plan. Robot Tank Quality Assurance.

Vemuri, D. S., Sebastian, A. A., Placid, A., & Elder, R., (2018, September 1). Can You Build a Streaming Video Robot Tank for ~\$100? *Robert Elder Software*. Retrieved 10 April 2021 from: <https://blog.robortelder.org/raspberry-pi-streaming-video-tank/>

Patel, P., Badhiwala, K., Bhatt, Y., Elder, R., (2019, August 12). A Cheaper PCB For the Robot Tank. *Robert Elder Software*. Retrieved 10 April 2021 from: <https://blog.robortelder.org/cheaper-pcb-for-robot-tank/>

Jumpnow Technologies (2017). Using the Raspberry Pi hardware PWM timers. Retrieved 02 Feb 2021, from: <https://jumpnowtek.com/rpi/Using-the-Raspberry-Pi-Hardware-PWM-timers.html>

Fisher, T. (2020). How to Open Command Prompt. Retrieved 10 April 2021, from: <https://www.lifewire.com/how-to-open-command-prompt-2618089>

Earl, B. (n.d.). Mini Pan-Tilt Kit Assembly. Adafruit Learning System. Retrieved 10 April 2021, from: <https://learn.adafruit.com/mini-pan-tilt-kit-assembly>

Thingiverse.com. (2018, July 15). Raspberry Pi Camera mount for micro servo driven pan-tilt kit by \_0o. Thingiverse. Retrieved 27 April 2021, from: <https://www.thingiverse.com/thing:3004766>.

MPlayer for Windows (2021). MPlayer & MEncoder Builds for Windows. Retrieved 27 April 2021, from: <http://mplayerwin.sourceforge.net/downloads.html>

Python. (2021). Download the latest source release. Retrieved April 02, 2021, from: <https://www.python.org/downloads/>

Badhiwala, K., Patel, P., Bhatt, Y. (2019). Redesigning of Robot Tank.

Dandekar, B.H., Kuncheria, K.A., Gidda, R. (2019). Servo camera (Robot Tank).

Elder, R. (2021). Hardware components page. Retrieved 27 April 21 from: <https://hardware.robortelder.ca/>