FERMILAB-POSTER-22-100-STUDENT

RVR Electrical Hardware and Software

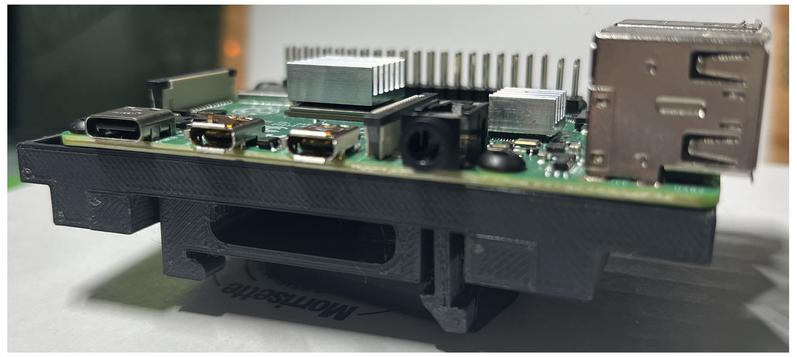
Amanda Hoeksema – Under the mentorship of Adam Watts

RVR's Purpose

In order to streamline the work of technicians and limit their radiation dosage in the accelerator tunnels, the Remote Viewing Robot (RVR) can be deployed instead to pinpoint the source of failure. RVR requires layers of improvement to begin such operation. The project detailed here aims to modernize and modularize the electrical hardware and software for the Remote Viewing Robot. For more information about the improvements made to RVR's design, refer to the posters written by Magdalena Sarna, Emma Stachowicz, and Maryum Fatima.

Organizing the Electrical Connections

All electrical components are mounted to DIN rails using 3D printed clips that are specifically designed for each part. The clips allow for easy adjustment of part position. The DIN rails mount to a 3D printed base that attaches inside RVR's chasse with heavy-duty Velcro, permitting easy removal to work with the electrical hardware and sturdy mounting to not shift during operation.



Closer view of the 3D printed DIN rail clip for the Raspberry Pi 4.

Establishing the Electrical Connections

With the goal to improve connectivity, the system uses the opto-isolators to safely connect the input terminals of the Sabertooth to the Raspberry Pi 4, protecting the Raspberry Pi 4 from damage in the event of a possible surge. The Sabertooth is essential in specifying the behavior of the circuit, setting the input to analog and simplifying the control of RVR's differential-drive. The necessary electrical connections are made and secured through use of soldering, screw terminals, and wire ferrules.

The basic structure for the electrical connections is as follows:

[1] 25V battery (not pictured)

↓

[2] 25W DC-DC converter

↓

[3] Sabertooth

↓

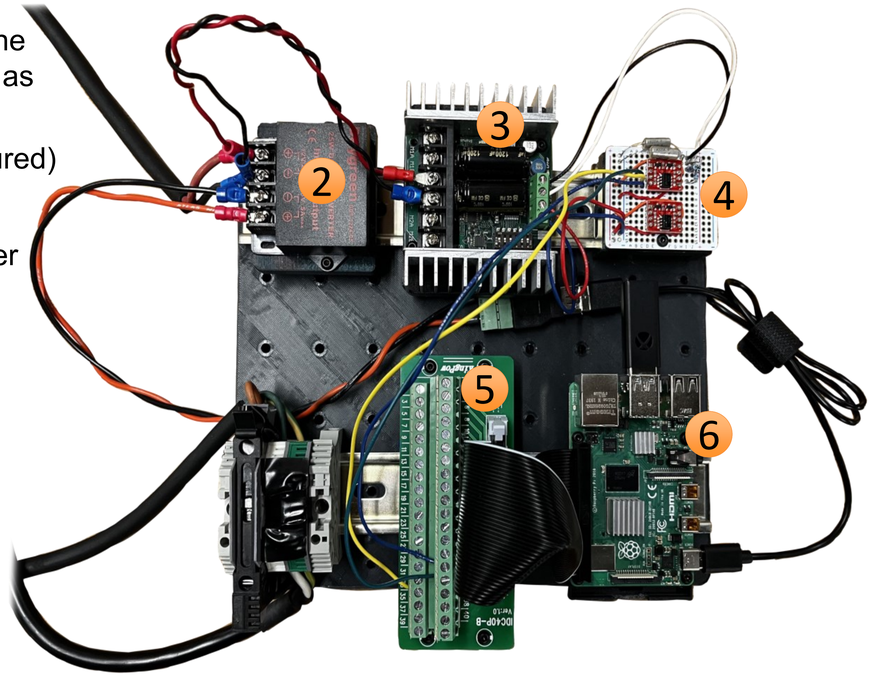
[4] Opto-isolators

↓

[5] Kingpow IDC

↓

[6] Raspberry Pi 4



Overview of the electrical connections that will sit in the RVR's chasse.

The Control Software

The software is written in Python and depends on the "LGPIO" libraries to interface with the Raspberry Pi 4's GPIO (general-purpose input/output) pins. The code receives input from a Bluetooth controller and converts this input to the required duty cycle and PWM (pulse-width-modulation) signal to achieve the desired movement. The process and conversion equations are detailed below:

- Joystick down = 0V = Backwards movement
- Joystick middle = 2.5V = No movement
- Joystick up = 5V = Forward movement

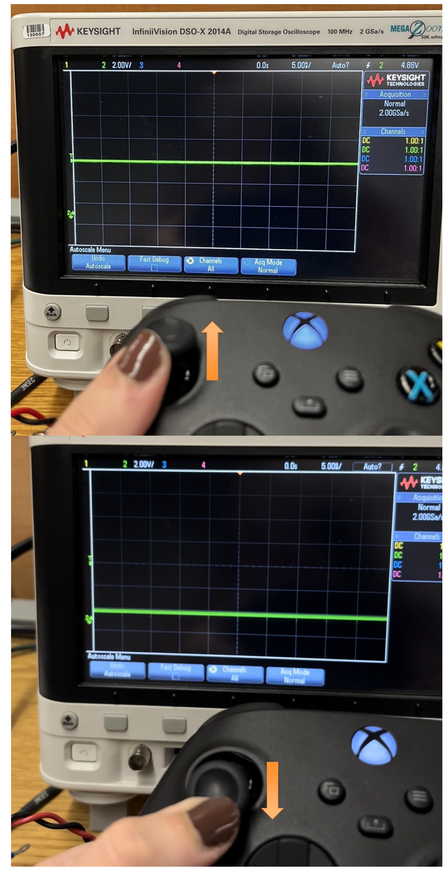
$$[1] DutyCycle = (Voltage + 0.8206) / 0.0527$$

$$[2] Voltage = 2.5Input + 2.5$$

Equation [1] converts the necessary voltage to the duty cycle percentage.
Equation [2] converts the controller input to the needed voltage.

Results and Conclusions

With the main electrical connections established and a functioning code written, the next steps are to write the code for turning RVR and debug the current code to ensure RVR runs without flaw.



Visual representation of the voltage changing on an oscilloscope using the controller.