Remote Pilotless Vehicle

User Guide

Version 1.0

1/19/2024

Table of Contents

Table of Contents

[Introduction 3](#_Toc156602020)

[Workflows 3](#_Toc156602021)

[Workflow 1: Directly connected. 3](#_Toc156602022)

[Workflow 2: Remotely connected. 3](#_Toc156602023)

[Running Programs 4](#_Toc156602024)

[Directories and Program Locations 4](#_Toc156602025)

[RPV-Server 5](#_Toc156602026)

[TCPSimClient\_A/B 6](#_Toc156602027)

[TCPSimClient\_C 7](#_Toc156602028)

[Controls 7](#_Toc156602029)

[TCPSimClient\_A (VRX Thrustmaster wheel) 7](#_Toc156602030)

[TCPSimClient\_B (Xbox Controller) 8](#_Toc156602031)

# Introduction

This user guide is intended to guide you through the procedures to get the vehicle up and running when you need to work on it. If you haven’t already, ensure that your development workstation and the vehicle have been set up according to the setup guide. This is a living document, so if your team adds or removes things on the vehicle, please add it to this document for future reference!

# Workflows

There are a few different ways to connect to the vehicle and start the necessary programs to drive the vehicle. You can follow whichever procedure fits your use case.

## Workflow 1: Directly connected.

This workflow assumes that you want to work with the vehicle while directly connected to it over Wi-Fi or Ethernet.

1. Power on the vehicle.
2. Connect your laptop to the rpv-loca**l** Wi-Fi network.
3. Connect a smartphone to the router and turn on USB tethering.
   1. Adjust router settings if device type changed from the last (refer to setup guide under “USB Tethering”)
4. If you want to enable lane keeping, follow the steps below. Otherwise, skip them.
   1. Ensure the Ethernet cable, webcam, USB-TTL adapter, and power cable are connected.
   2. Power on the computer.
   3. Remotely connect to the OpenCV computer.
   4. Open a terminal and navigate to the Python script.
   5. Run “python3 main.py”
   6. A window showing the camera’s perspective should open.
   7. You can disconnect if you would like.
5. Open PuTTY and connect to the Pi using 192.168.5.132.
6. Continue by referring to the “Running Programs” section.
7. Open a web browser and navigate to <http://192.168.5.132:8080/stream/webrtc>
8. Click the green “Call” button on the bottom of the page.
9. You should now see the video stream.

## Workflow 2: Remotely connected.

This workflow assumes that you want to work with the vehicle while remotely connected to it through a VPN tunnel. These steps are performed on your laptop and VRX simulator if you’re using it.

1. Power on the vehicle.
2. Connect your laptop to the rpv-local Wi-Fi network.
3. Connect a smartphone to the router and turn on USB tethering.
   1. Adjust router settings if device type changed from the last (refer to setup guide under “USB Tethering”)
4. If you want to enable lane keeping, follow the steps below. Otherwise, skip them.
   1. Ensure the Ethernet cable, webcam, USB-TTL adapter, and power cable are connected.
   2. Power on the computer.
   3. Remotely connect to the OpenCV computer.
   4. Open a terminal and navigate to the Python script.
   5. Run “python3 main.py”
   6. A window showing the camera’s perspective should open.
   7. Disconnect from the computer (you will lose connection in step 9).
5. Open VNC Viewer and connect to the Pi’s desktop using 192.168.5.132.
6. Open a new terminal in the desktop environment.
7. Start the OpenVPN connection by running “sudo openvpn --config /path/to/config.ovpn”
   1. Replace *“*/path/to/config.ovpn*”* with the location of the client file.
8. Leave the terminal open and disconnect VNC Viewer.
9. Disconnect your laptop from the rpv-local Wi-Fi network.
10. Connect your laptop or VRX simulator to the university’s network via Wi-Fi or Ethernet.
11. Connect to the VPN server from your laptop or VRX simulator.
12. Open PuTTY and connect to the Pi using its VPN IP address (refer to setup guide under “OpenVPN Client Setup [Linux Setup]”)
13. Continue by referring to the “Running Programs” section.
14. Open a web browser and navigate to http://<VPN IP>:8080/stream/webrtc
15. Click the green “Call” button on the bottom of the page.
16. You should now see the video stream.

# Running Programs

If you only need to start the main programs, refer to “RPV-Server” and the appropriate “TCPSimClient\_X”.

## Directories and Program Locations

All programs are located in the rpv-sdp folder at **/home/cpp-rpv1/rpv-sdp**

|  |  |  |
| --- | --- | --- |
| Directory | Main Program Location | Description |
| OpenCV | Live\_Lane\_Detection\_Scratch/main.py | Source code for the OpenCV PC. |
| prototypes | N/A | Random test programs. Not used, only kept as reference. |
| RPV-Server | main | Main server program. |
| subsystems | N/A | Source code for Arduino subsystems. |
| TCPSimClient\_A | X64/Debug/ControlClient\_AGV.exe | Client program for VRX simulator setup. |
| TCPSimClient\_B | Debug/TCPSimClient\_B.exe | Client program for Xbox 360 controller. |
| TCPSimClient\_C | Debug/TCPSimClient\_C.exe | Client program for keyboard input. |
| XMPP-client | main.py | Source code for XMPP client. |

## RPV-Server

To run the main program, first ensure that the pigpio daemon is running. Run the following to start the daemon:

sudo pigpiod

When it prompts for a password, type **022223** and hit Enter.

You only need to do this once after bootup. If it throws the error below it very likely means the daemon is already running.

A black background with white text

Description automatically generated

Next, navigate to the main executable:

cd /home/cpp-rpv1/rpv-sdp

Finally, run the executable:

./main

If you see any exclamation marks like shown below, it means one or more I/O devices did not initialize. If all of them failed, the daemon is not running.

A screen shot of a computer

Description automatically generated

At startup, the main program will ask you a few questions:

A screenshot of a computer

Description automatically generated

The first question asks what address to listen on. If you’re directly connected to the vehicle’s Wi-Fi network, use option 1. If you are connected to the vehicle over OpenVPN, use option 2. If you want it to listen on a different socket, choose option 3.

Next, specify if you want to enable lane keep. If you do, it will enable the serial port.

**WARNING!** If you enable lane keep, ensure that the mini-PC is powered on and its USB-TTL adapter (blue thing) is connected to it **AND** to the Pi’s serial pins (both RX and GND). If the adapter is connected to the serial pins but not the PC, the serial buffer will fill with garbage data and eventually overflow. **This has caused crashes in the past!** Leave the serial pins disconnected whenever lane keep is disabled. Pictured below are the serial pins:

A close-up of several wires

Description automatically generated

The last question is to enable/disable Active Safety. If enabled, the Active Safety state machine will be active. Its behavior is fully documented in the project report. In short, if the vehicle senses an obstacle in the front or rear, it will engage its brakes and set throttle to zero. To regain control, let go of the accelerator and press it again. You have a few seconds to get the vehicle away from the obstacle. If it still senses something, it will repeat the cycle.

After answering the questions, the main program will begin listening at whatever address you specified. Nothing will happen until a client program connects.

## TCPSimClient\_A/B

Ensure that the Xbox controller or the Thrustmaster wheel and its accessories are connected before launching the client. You can either run the .exe directly or open the project in Visual Studio and run debugging from there. Select the server’s socket and press Enter. If the vehicle doesn’t respond to your inputs, try re-running the client.

## TCPSimClient\_C

This client uses your keyboard as an input device. It asks the same question as above and asks what the acceleration speed should be since the keyboard input is binary. Here is how the keyboard is mapped out:

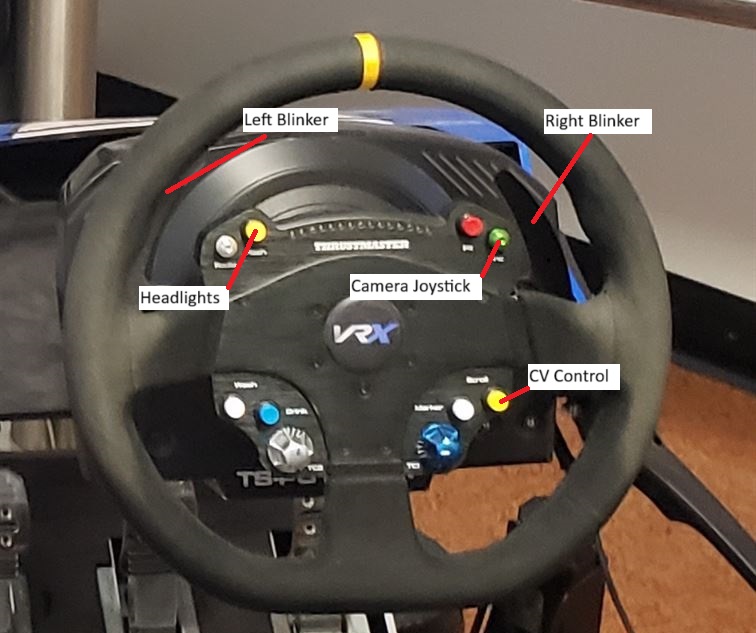
Steering & Throttle

|  |  |
| --- | --- |
| Key | Function |
| Left arrow | Steer left |
| Right arrow | Steer right |
| Up arrow | Accelerate forward |
| Down arrow | Reverse |
| Space | Brake |
| Q | Pan camera left |
| W | Camera center |
| E | Pan camera right |
| X | Signal right |
| Z | Signal left |
| T | Toggle CV control |
| H | Toggle headlights |

# Controls

## TCPSimClient\_A (VRX Thrustmaster wheel)

* Steering is controlled by the steering wheel.
* Acceleration is controlled by the rightmost pedal.
* Braking is controlled by the middle pedal.
* The leftmost pedal does nothing.
* Shifting the gear selector into any gear makes the vehicle go forwards.
* Shifting the gear selector into reverse makes the vehicle go backwards.
* The camera is controlled by the green joystick. It can move left, right, and can be pushed inward to center the camera.
* The other buttons are shown below:



## TCPSimClient\_B (Xbox Controller)

The vehicle’s controls are mapped as such on the Xbox controller:



