USING A RASPBERRY PI AS A VIDEO COPROCESSOR

Using a Raspberry PI as a video coprocessor

Table of Contents

Vision processing with a coprocessor

Using a Coprocessor for vision processing

Vision processing using libraries like OpenCV for recognizing field targets or game pieces can often be a CPU intensive process. Often the load isn't too significant and the processing can easily be handled by the roboRIO. In cases where there are more camera streams or the image processing is complex, it is desirable to off-load the roboRIO by putting the code and the camera connection on a different processor. There are a number of choices of processors that are popular in FRC such as the Raspberry PI, the intel-based Kangaroo, the LimeLight for the ultimate in simplicity, or for more complex vision code a graphics accelerator such as one of the nVidia Jetson models.

Strategy

Generally the idea is to set up the coprocessor with the required software that generally includes:

- OpenCV - the open source computer vision library
- Network tables - to commute the results of the image processing to the roboRIO program
- Camera server library - to handle the camera connections and publish streams that can be viewed on a dashboard
- The language library for whatever computer language is used for the vision program
- The actual vision program that does the object detection

The coprocessor is connected to the roboRIO network by plugging it into the extra ethernet port on the network router or, for more connections, adding a small network switch to the robot. The cameras are plugged into the coprocessor, it acquires the images, processes them, and publishes the results, usually target location information, to network tables so it is can be consumed by the robot program for steering and aiming.

Using a Raspberry PI as a video coprocessor

Camera connected to coprocessor which is connected to the roboRIO. Vision code runs on the coprocessor, roboRIO runs only robot program.
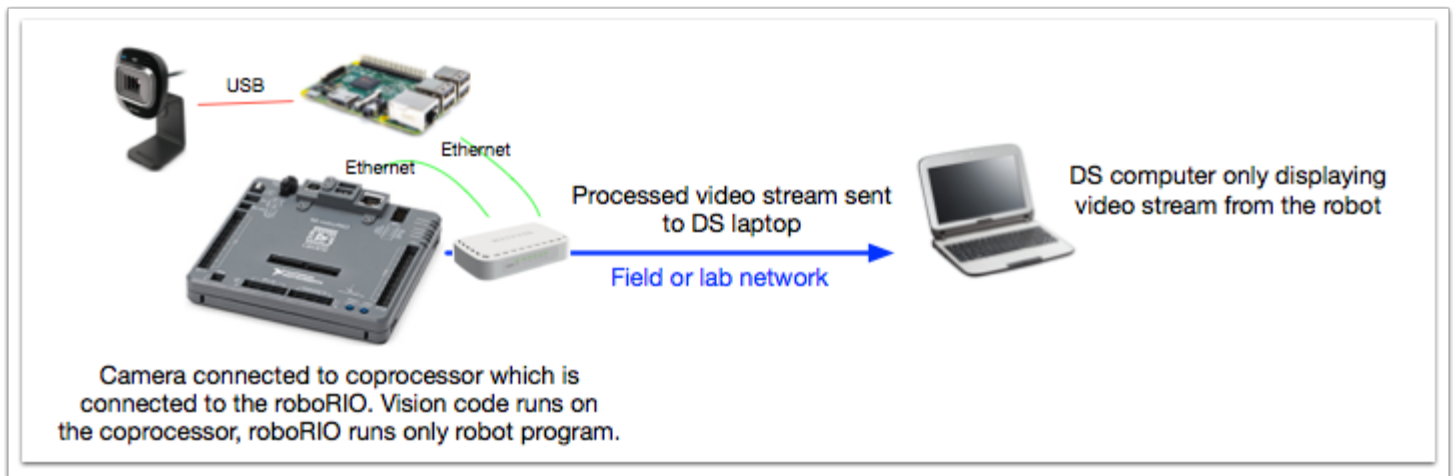
Streaming camera data to the dashboard

it is often desirable to simply stream the camera data to the dashboard over the robot network. In this case one or more camera connections can be sent to the network and viewed on a dashboard such as Shuffleboard or a web browser. Using Shuffleboard has the advantage of having easy controls to set the camera resolution and bit rate as well as integrating the camera streams with other data sent from the robot.

It is also possible to process images and add annotation to the image, such as target lines or boxes showing what the image processing code has detected then send it forward to the dashboard to make it easier for operators to see a clear picture of what's around the robot.

Using the Raspberry PI for FRC

One of the most popular coprocessor choices is the Raspberry PI because:

- Low cost - around $35
- High availability - it's easy to find Raspberry PIs from a number of suppliers, including Amazon
- Very good performance - the current Raspberry PI 3b+ has the following specifications:
    - Technical Specifications:
    - - Broadcom BCM2837BO 64 bit ARMv8 QUAD Core A53 64bit Processor powered Single Board Computer run at 1.4GHz
    - - 1GB RAM - BCM43143 WiFi on board
    - - Bluetooth Low Energy (BLE) on board
    - - 40 pin extended GPIO - 4 x USB2 ports
    - - 4 pole Stereo output and Composite video port
    - - Full size HDMI
    - - CSI camera port for connecting the Raspberry
    - Pi camera - DSI display port for connecting the Raspberry
    - Pi touch screen display - MicroSD port for loading your operating system and storing data
    - - Upgraded switched Micro USB power source (now supports up to 2.5 Amps.
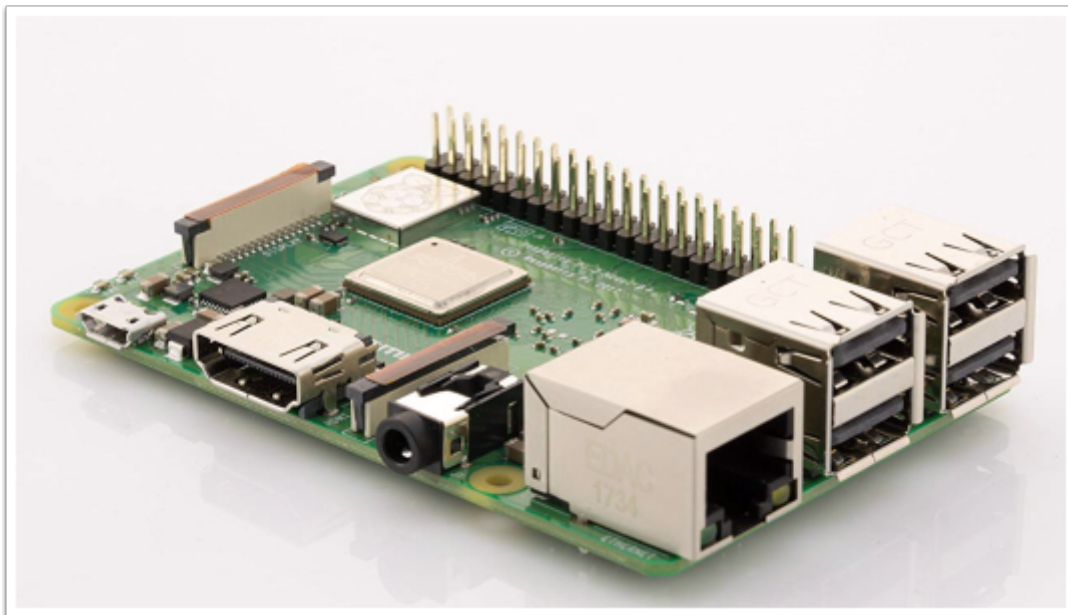
Using a Raspberry PI as a video coprocessor

Pre-built Raspberry PI image

To make using the Raspberry PI as easy as possible for teams, there is a provided Raspberry PI image. The image can be copied to a micro SD card, inserted into the PI, and booted. By default it supports:

- a web interface for configuring it for the most common functions
- supports an arbitrary number camera streams (defaults to one) that are published on the network interface
- OpenCV, Network Tables, Camera Server, and language libraries for C++, Java, and Python custom programs

If the only requirement is to stream one or more cameras to the network (and dashboard) then no programming is required and can be completely set up through the web interface.

The next section discusses how to install the image onto a flash card and boot the PI.

Installing the image on a Raspberry PI

What you need to get the PI image running

To start using the Raspberry PI as a video or image coprocessor you need the following:

- A Raspberry PI 3 B or Raspberry PI 3 B+
- A micro SD card that is at least 4Gb to hold all the provided software
- An ethernet cable to connect the PI to your roboRIO network
- A USB micro power cable to connect to the Voltage Regulator Module (VRM) on your robot. It is recommended to use the VRM connection for power rather than powering it from one of the roboRIO USB ports for higher reliability
- A laptop that can write the MicroSD card, either using a USB dongle (preferred) or a SD to MicroSD adapter that ships with most MicroSD cards

Shown is an inexpensive USB dongle that will write the FRC image to the MicroSD card.

Installing the image to your MicroSD card

Getting the FRC Raspberry PI image

The image is stored on the GitHub release page for the FRC PI-gen repository: https://github.com/wpilibsuite/FRCVision-pi-gen/releases.

In addition to the instructions on this page, see the documentation on the github web page (below).

The image is fairly large so have a fast internet connection when downloading it. Always use the most recent release from the top of the list of releases.
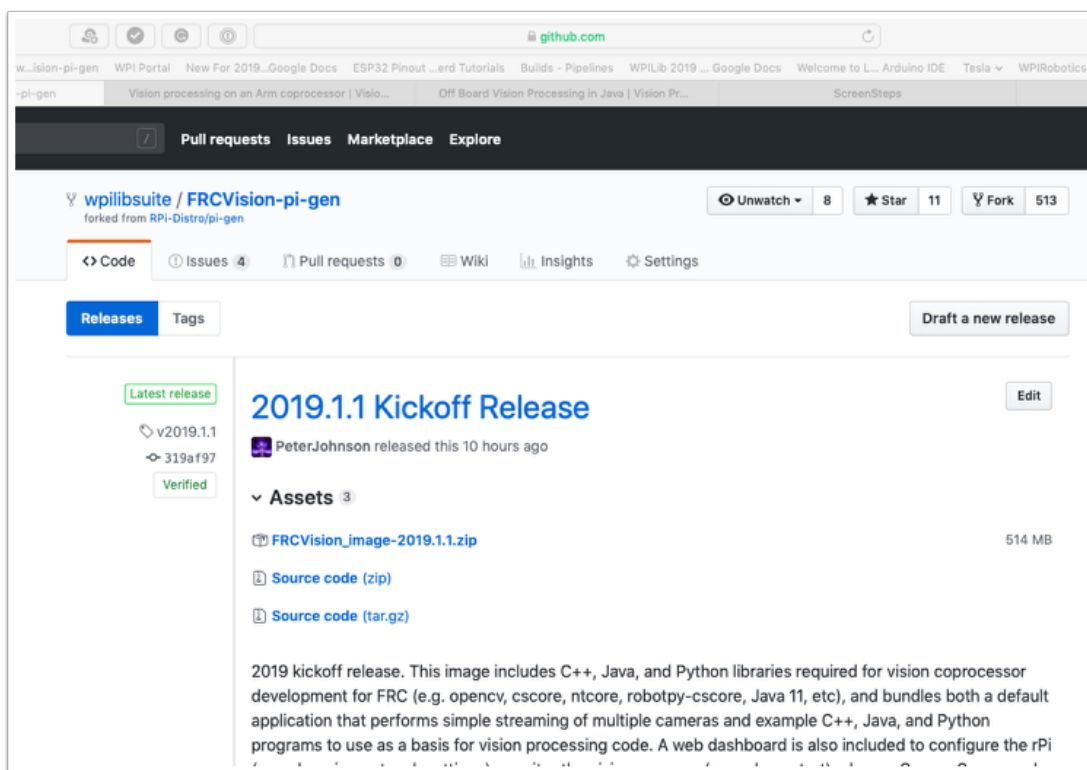

Copy the image to your MicroSD card

Download and install Etcher (https://www.balena.io/etcher/) to image the micro SD card. The micro SD card needs to be at least 4 GB. Note: a micro SD to USB dongle such as https://www.amazon.com/gp/product/B0779V61XB works well for writing to micro SD cards.

Using a Raspberry PI as a video coprocessor
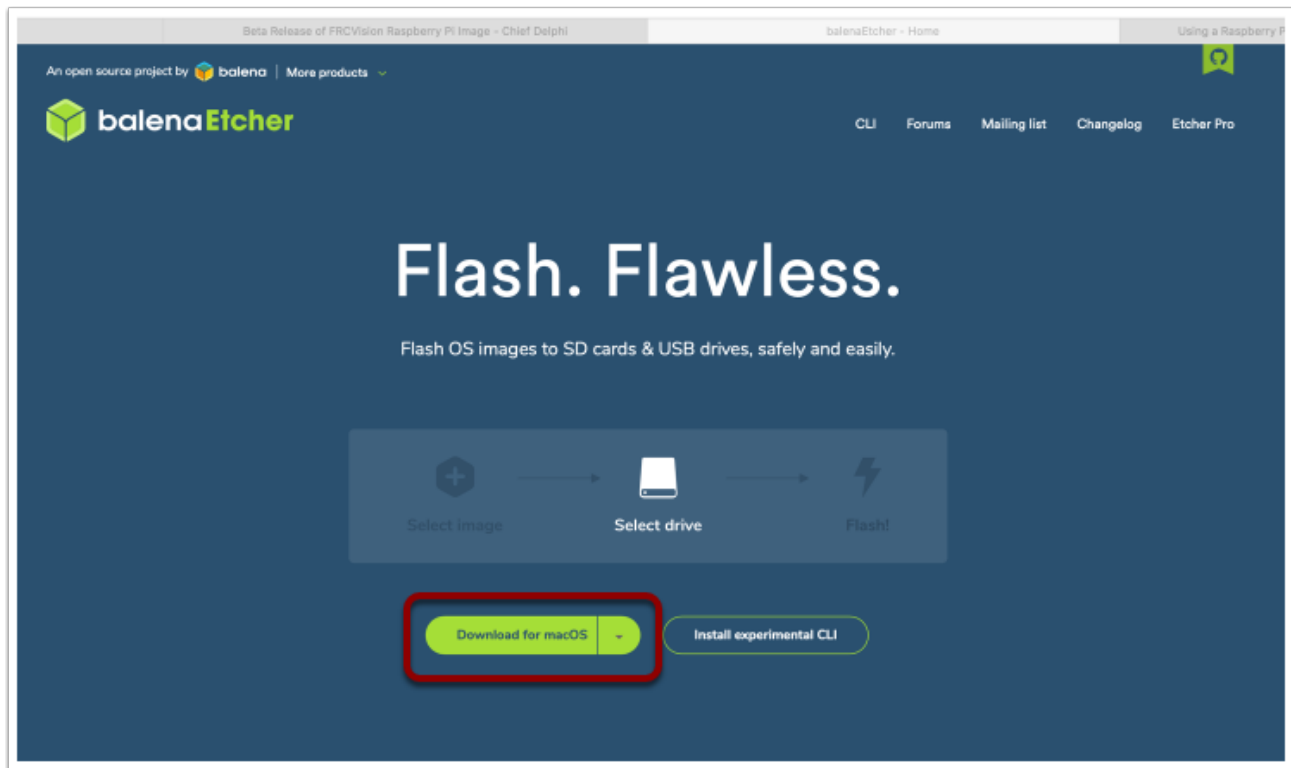
Flash the MicroSD card with the image using Etcher by selecting the zip file as the source, your SD card as the destination and click "Flash". Expect the process to take about 3 minutes on a fairly fast laptop.

Using a Raspberry PI as a video coprocessor
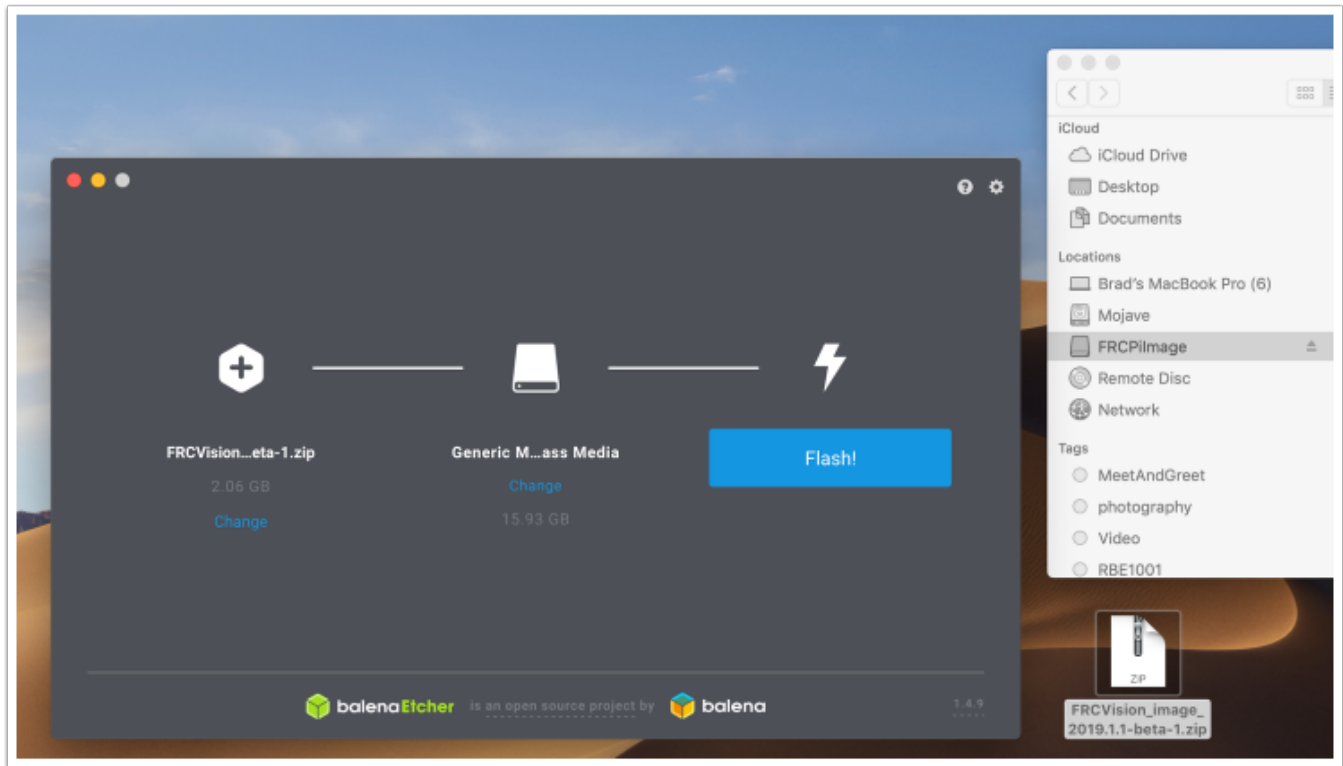
Testing the Raspberry PI

1. Put the micro SD card in a rPi 3 and apply power.
2. Connect the rPi 3 ethernet to a LAN or PC. Open a web browser and connect to http://frcvision.local/ to open the web dashboard. On the first bootup the filesystem will be writable, but later bootups will default to read only, so it's necessary to click the "writable" button to make changes.

Logging into the Raspberry PI

Most tasks with the rPi can be done from the web console interface. Sometimes for advanced use such as program development on the rPi it is necessary to log in. To log in, use the default Raspberry PI password:

```
Username: pi
Password: raspberry
```

Using a Raspberry PI as a video coprocessor

Known issues with this release (errata). These will be fixed in the next release.

- Downloading the Java example program on the web dashboard does not work. Grab it manually from /home/pi/zips instead (using ssh).
- Console output enable button on the web dashboard doesn't persist correctly through a rPi reboot (it needs to be disabled and re-enabled to again get console output).

Using the Raspberry PI as a camera server

The Raspberry PI FRC Console

The FRC image for the Raspberry PI includes a console that can be viewed in any web browser that makes it easy to:

- Look at the Raspberry PI status
- View the status of the background process running the camera
- View or change network settings
- Look at each camera plugged into the rPI and add additional cameras
- Load a new vision program onto the rPI

Setting the rPI to be Read-Only vs. Writable

The rPI is normally set to Read-Only which means that the file system cannot be changed. This ensures that if power is removed without first shutting down the rPi the file system isn't corrupted. When settings are changed (following sections), the new settings cannot be saved while the rPI file system is set as Read-Only. Buttons are provided that allow the file system to be changed from Read-Only to Writable and back whenever changes are made. If the other buttons that change information stored on the rPI cannot be press, check the Read-Only status of the system.
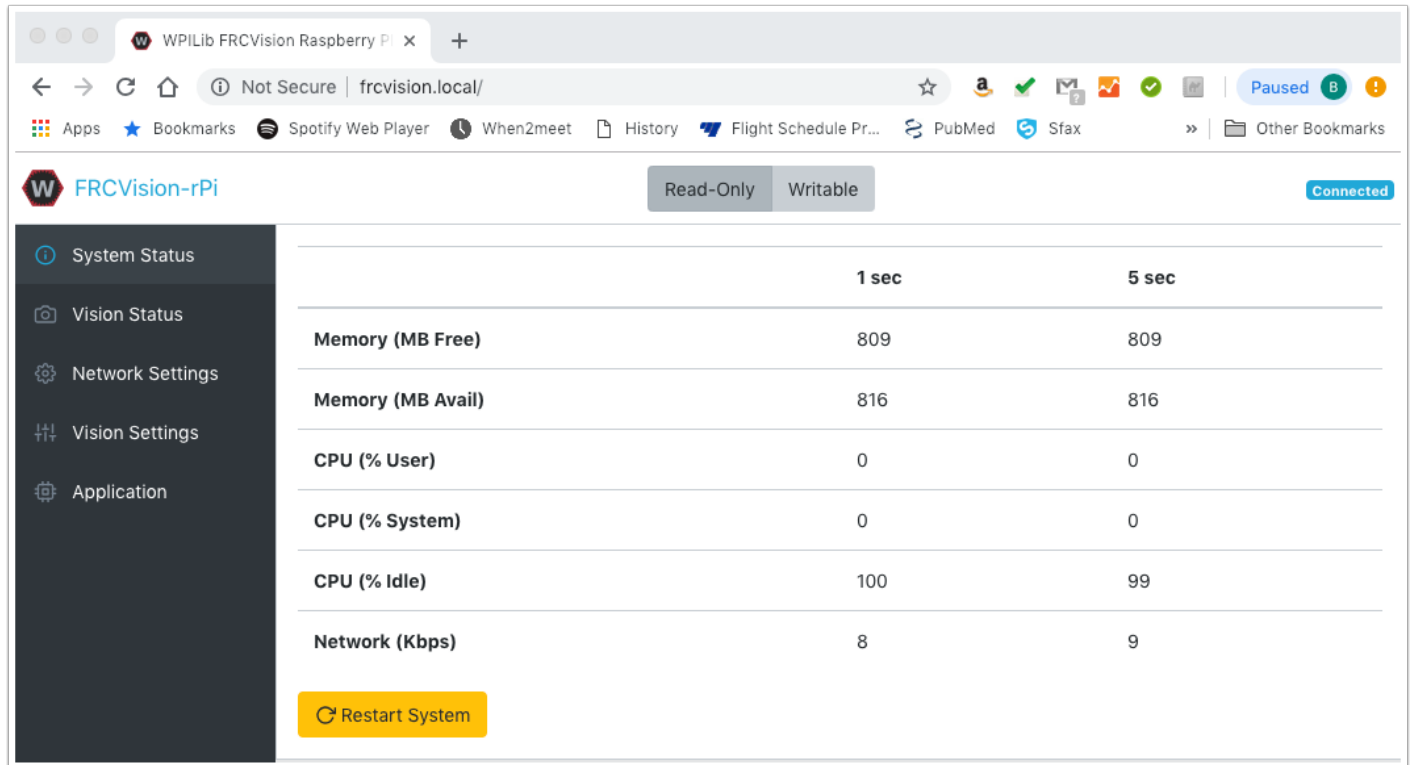
Status of the network connection to the rPI

There is a label in the top right corner of the console that indicates if the rPi is currently connected. It will change from Connected to Disconnected if there is no longer a network connection to the rPi.
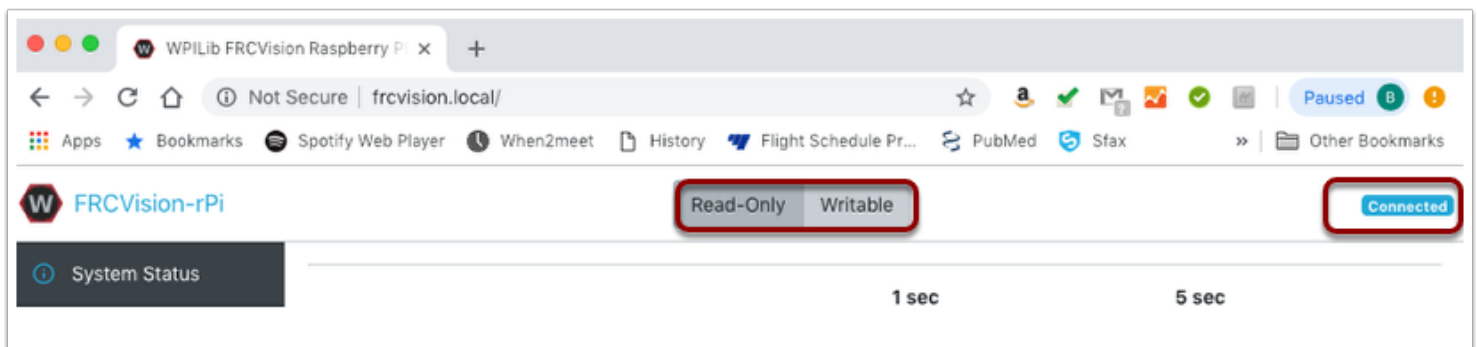
Using a Raspberry PI as a video coprocessor

System status

The system status shows what the CPU on the rPI is doing at any time. There are two columns of status values, on being a 1 second average and the other a 5 second average. Shown is:

- free and available RAM on the PI
- CPU usage for user processes and system processes as well as idle time.
- And network bandwidth - which allows one to determine if the used camera bandwidth is exceeding the maximum bandwidth allowed in the robot rules for any year.

Using a Raspberry PI as a video coprocessor

Vision Status
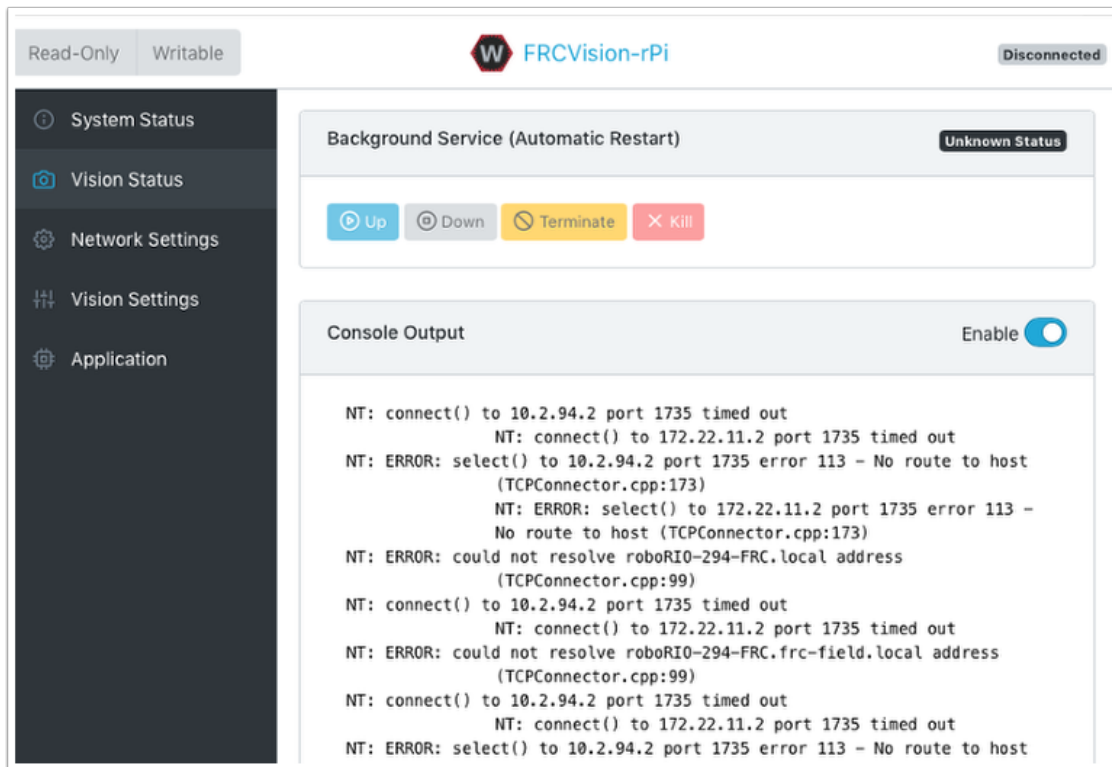
Allows monitoring of the task which is running the camera code in the rPI, either one of the default programs or your own program in Java, C++, or Python. You can also enable and view the console output to see messages coming from the background camera service. In this case there are number of messages about being unable to connect to network tables (NT: connect()) because in this example the rPI is simply connected to a laptop with no Network Tables server running (usually the roboRIO.)

Network Settings

The rPI network settings have options to connect to the PI:

- DHCP - the default name resolution usually used by the roboRIO. The default name is raspberry.local.
- Static - where a fixed IP address, network mask, and router settings are filled in explicitly
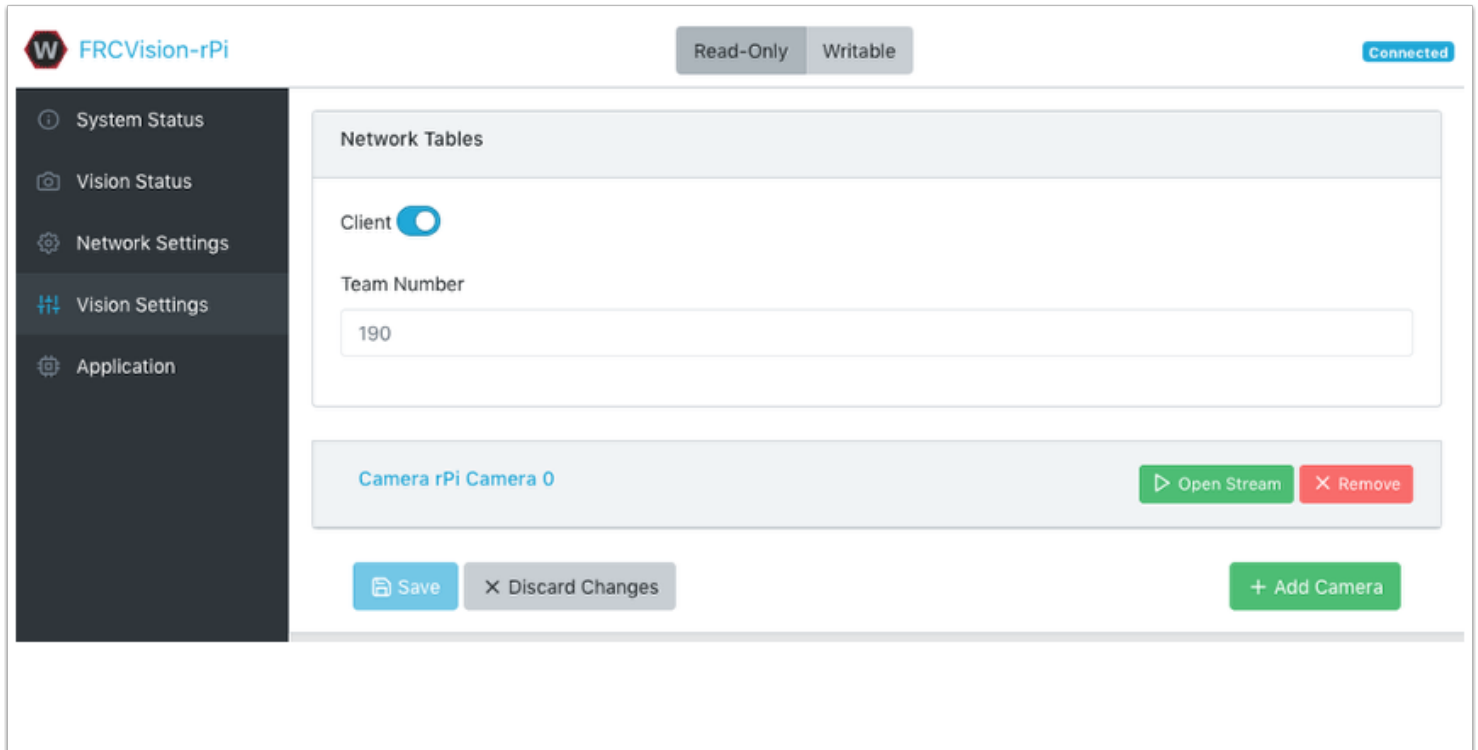
Using a Raspberry PI as a video coprocessor

- DHCP with Static Fallback - DHCP with Static Fallback - the PI will try to get an IP address via DHCP, but if it can't find a DHCP server, it will use the provided static IP address and parameters
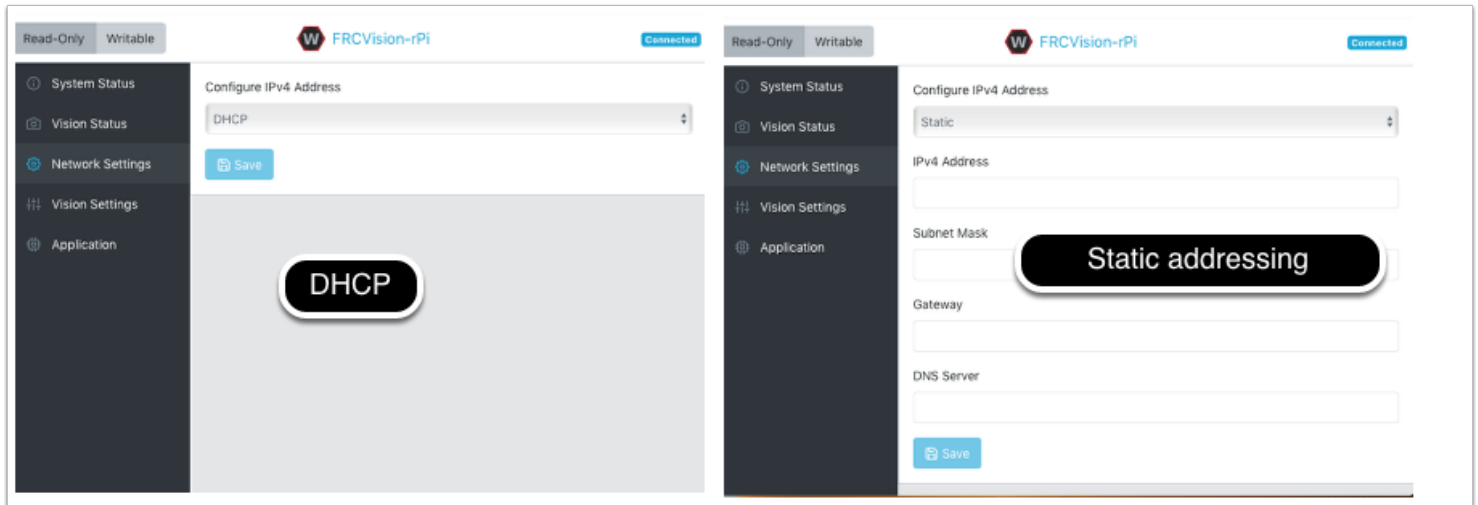
The picture above is showing the settings for both DHCP and Static IP Addressing. The mDNS name for the rPi should always work regardless of the options selected above.

# Using a Raspberry PI as a video coprocessor

Vision Settings

The Vision Settings are to set the parameters for each camera and whether the rPI should be a NetworkTables client or server. There can only be one server on the network and the roboRIO is always a server. Therefor when connected to a roboRIO, the rPI should always be in client mode with the team number filled in. If testing on a desktop setup with no roboRIO or anything acting as a server then it should be set to Server (Client switch is off).

To view and manipulate all the camera settings click on the camera in question. In this case the camera is called "Camera rPi Camera 0" and clicking on the name reveals the current camera view and the associated settings.

Using a Raspberry PI as a video coprocessor
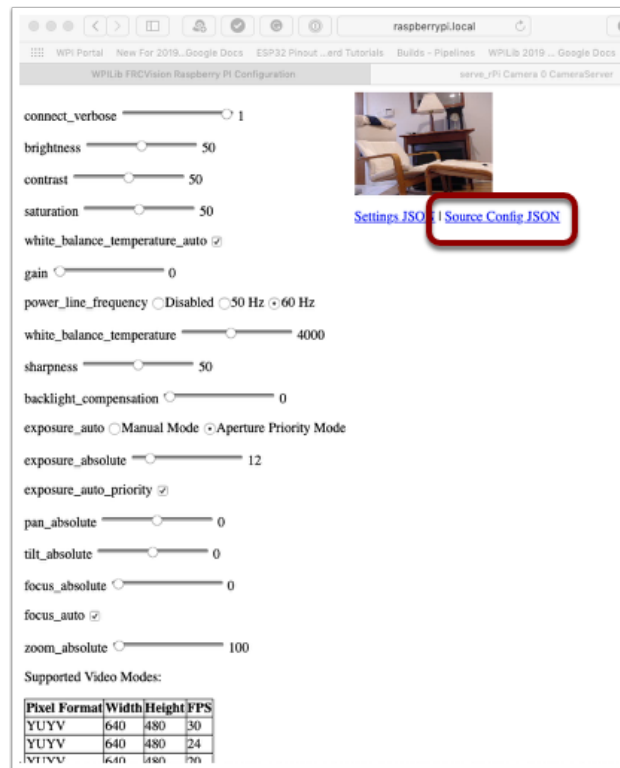
Manipulating the camera settings is reflected in the current camera view. The bottom of the page shows all the possible camera modes (combinations of Width, Height, and frame rates) that are supported by this camera.

Getting the current settings to persist over reboots

The rPi will load all the camera settings on startup. Editing the camera configuration in the above screen is temporary. To make the values persist click on the "Load Source Config From Camera" button and the current settings will be filled in on the camera settings fields. Then click "Save" at the bottom of the page. *Note: you must set the file system Writeable in order to save the settings. The Writeable button is at the top of the page.*

Using a Raspberry PI as a video coprocessor

There are some commonly used camera settings values shown in the camera settings (above). These values Brightness, White Balance, and Exposure are loaded into the camera before the user JSON file is applied. So if a user JSON file contains those settings they will overwrite the ones from the text field.

Application

The Application tab shows the application that is currently running on the rPi.

Vision workflows

There is a sample vision program using OpenCV in each of the supported languages, C++, Java, or Python. Each sample program can capture and stream video from the rPi. In addition, the samples have some minimal OpenCV. They are all set up to be extended to replace the provided OpenCV sample code with the code needed for the robot application. The rPi Application tab supports a number of programming workflows:

- Stream one or more cameras from the rPi for consumption on the driver station computer and displayed using ShuffleBoard

Using a Raspberry PI as a video coprocessor

- Edit and build one of the sample programs (one for each language: Java, C++ or Python) **on the rPi** using the included toolchains
- Download a sample program for the chosen language and edit and build it on your **development computer**. Then upload that built program back to the rPi
- Do everything yourself using completely custom applications and scripts (probably based on one of the samples)
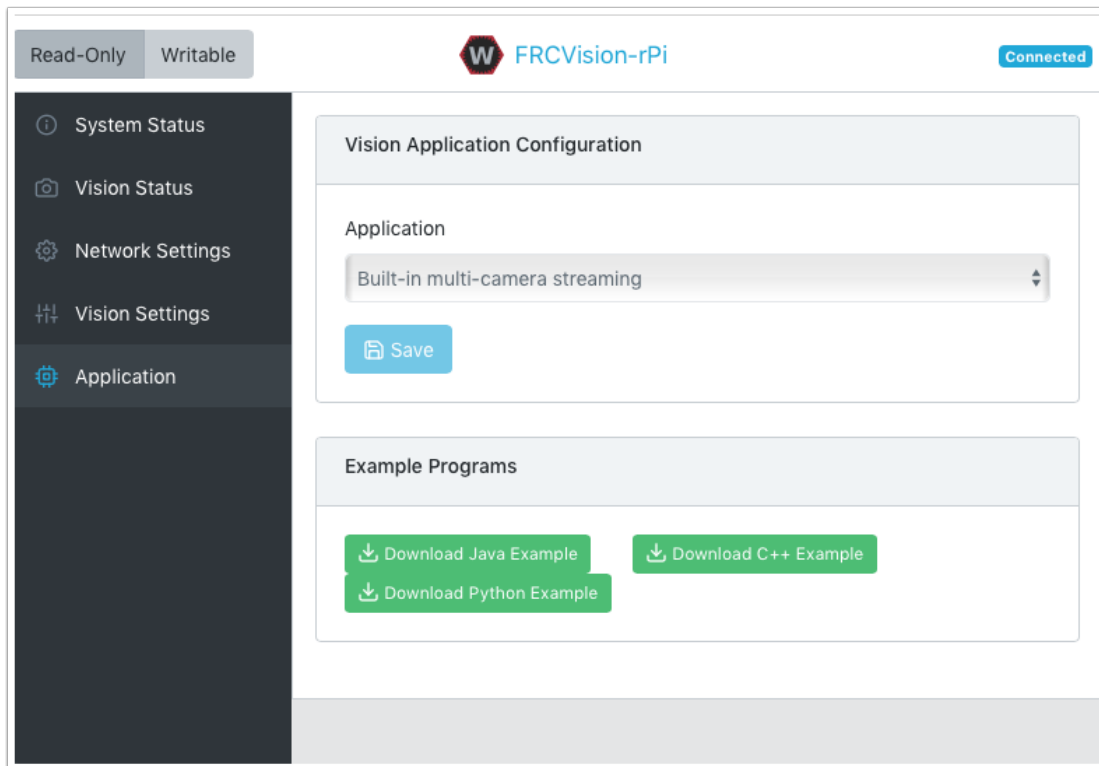
The running application can be changed by selecting one of the choices in the drop-down menu. The choices are:

- Built-in multi camera streaming which steams whatever cameras are plugged into the rPi. The camera configuration including number of cameras can be set on the "Vision Settings" tab.
- Custom application which doesn't upload anything to the rPi and assumes that the developer wants to have a custom program and script.
- Java, C++ or Python pre-installed sample programs that can be edited into your own application.
- Java, C++, or Python uploaded program. Java programs require a .jar file with the compiled program and C++ programs require an rPi executable to be uploaded to the rPI.

Using a Raspberry PI as a video coprocessor
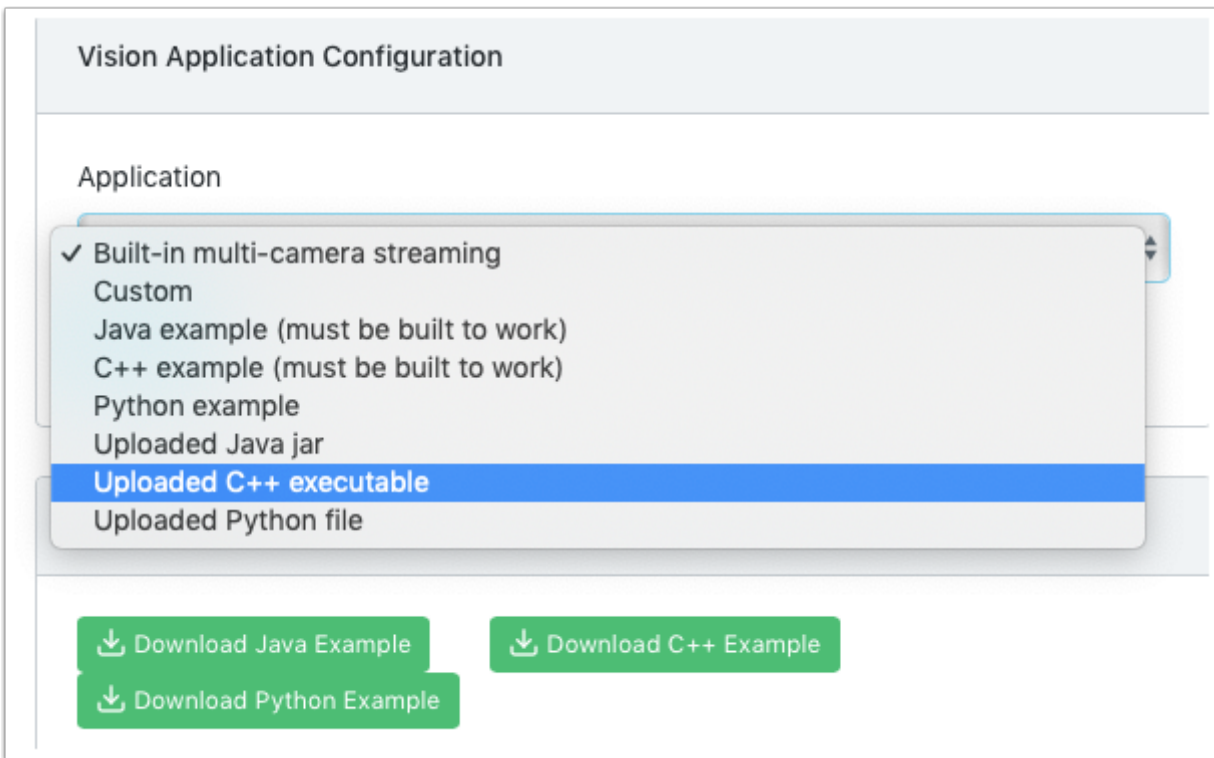
When selecting one of the Upload options, a file chooser is presented where the jar, executable or Python program can be selected and uploaded to the rPi. In the following picture an Uploaded Java jar is chosen and the "Choose File" button will select a file and clicking on the "Save" button will upload the selected file.

Note: in order to Save a new file onto the rPi, the file system has to be set writeable using the "Writable" button at the top left of the web page. After saving the new file, set the file system back to "Read-Only" so that it is protected against accidental changes.

Using a Raspberry PI as a video coprocessor

Application

Uploaded Java jar

Upload executable file

Choose File | no file selected

🖫 Save

Example Programs

⬇ Download Java Example   ⬇ Download C++ Example

⬇ Download Python Example