#4499

www.highlandersfrc.com

The Highlanders

Robotics Team

“It’s not about the game, it’s about the journey”

FIRST ROBOTICS COMPETITION

JeVois in FRC

7/28/2020

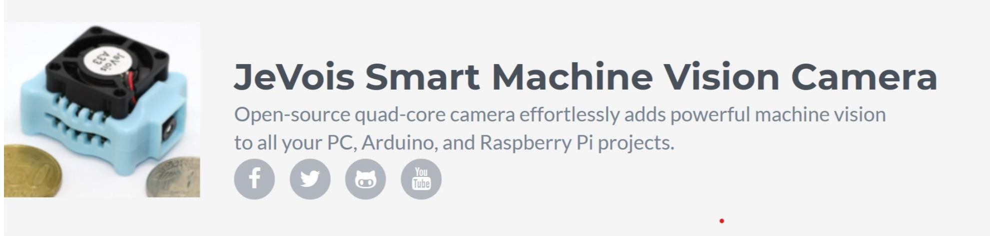
Contents

- Basics and Overview
- Getting Started
- Electrical Layout
- Basic Jevois Commands
- Computer Vision Algorithm
- Basic Code - object detection
- Tuning HSV
- Additional Tips
- Lessons Learned
- Pitfalls
- Questions

*We will be focusing on the 2020, Infinite Recharge Game and target

Basics and Overview

Jevois - What is it?



JeVois Smart Machine Vision Camera

Open-source quad-core camera effortlessly adds powerful machine vision to all your PC, Arduino, and Raspberry Pi projects.

- Camera System with onboard computer vision processor
- Fully self contained
- Flexible lens selection

Jevois - Why?

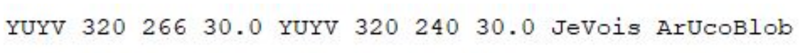
- Small
- Only $50
- Flexible and customizable solution

Why not LIMELIGHT?

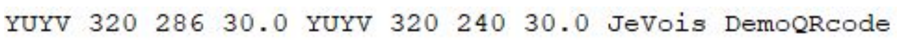
- Expensive
- Not as customizable

Camera Information

- Used as a driver vision camera and for computer vision
- Comes with 38 premade modules
  - Face Detection
  - QR code tracker
  - Etc
- Pre - made modules are a good intro to computer vision
- Custom modules can be built with OpenCV and Python, or C++



```
YUYV 320 266 30.0 YUYV 320 240 30.0 JeVois ArUcoBlob
```



```
YUYV 320 286 30.0 YUYV 320 240 30.0 JeVois DemoQRcode
```

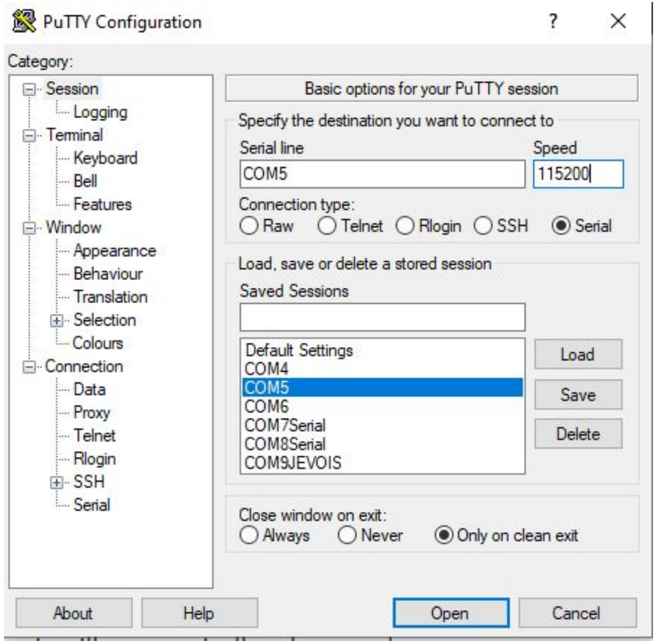
Required Software

- Jevois SD Card - Image that will be flashed onto the Jevois
- Notepad++ - Used to edit python programs on the Jevois
- Putty - Terminal used to connect to the Jevois
- OBS Studio - Used to see camera image
- Highlanders do not recommend Jevois Inventor - Headless mode, inconsistent operation, difficult to transition from computer to robot.

Getting Started

Connect to Jevois

- Connect through putty
- Each module is defined by the screen resolution and FPS
- To edit code, use putty to mount the drive
- Then navigate to the correct python program and edit with Notepad++
- Once the code is edited, eject the drive
- The Jevois will automatically reboot and you can continue to test



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM5

Speed: 115200

Connection type: ○ Raw ○ Telnet ○ Rlogin ○ SSH ● Serial

Load, save or delete a stored session

Saved Sessions

Default Settings
COM4
COM5
COM6
COM7Serial
COM8Serial
COM9JEVOIS

Close window on exit: ○ Always ○ Never ● Only on clean exit

Jevois Organization

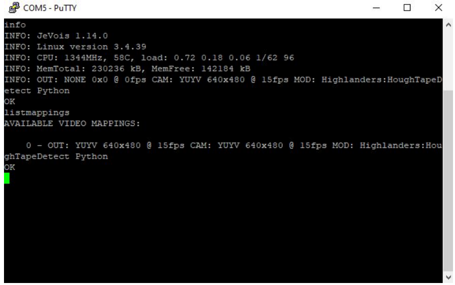
- Jevois is organized into a few different sections of code
  - Modules
  - Config
- Modules include the pre - built Jevois modules as well as custom modules
- Config is split into a few different programs
  - Videomappings - List of all the modules that the Jevois has
    - The custom modules must be written into the videomappings file using the image resolution and FPS
  - Initscript - Run on start up of the Jevois
    - Used to set exposure, gain and vision module

Basic Jevois Commands

setpar serout ...: Command used to tell the jevois where to send the serial outputs(USB, serial, or both)

usbsd: This command will allow you to access the Jevois files, and is the main way to edit your program.

Command Line Interface



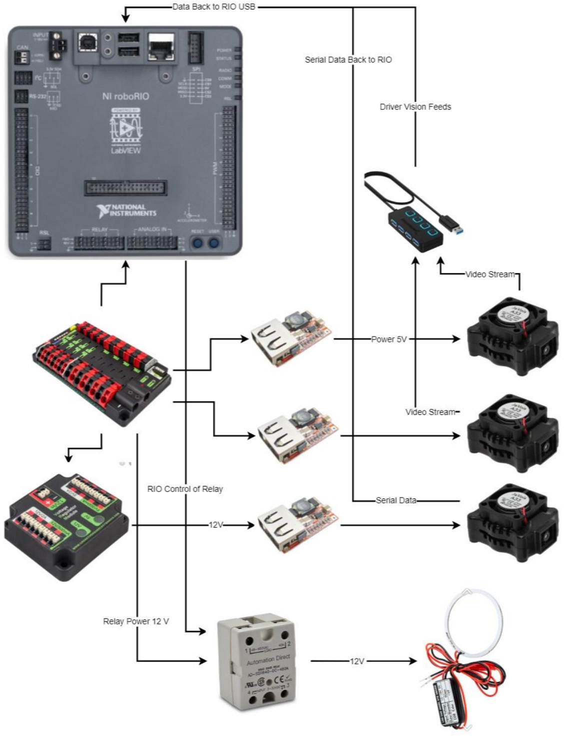
```
info
INFO: JeVois 1.14.0
INFO: Linux version 3.4.39
INFO: CPU: 1344MHz, 58C, load: 0.72 0.18 0.06 1/62 96
INFO: MemTotal: 230236 kB, MemFree: 142184 kB
INFO: OUT: NONE 0x0 @ 0fps CAM: YUYV 640x480 @ 15fps MOD: Highlanders:HoughTapeD
etect Python
OK
listmappings
AVAILABLE VIDEO MAPPINGS:

    0 - OUT: YUYV 640x480 @ 15fps CAM: YUYV 640x480 @ 15fps MOD: Highlanders:Hou
ghTapeDetect Python
OK
```

Electrical Layout

Electrical Layout

- Leave room for wires to be routed and room to unplug cables from jevois
- Driver Vision Cameras
  - Only need power cable(USB)
  - Uses a y split cable that is plugged into a USB hub and power buck
- Computer Vision Camera
  - Need both power and serial cables
  - Plug the power cable into a power buck
  - Plug the serial cable into the Roborio

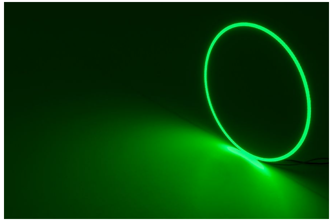
*The power bucks were incredibly important - otherwise during a round you will lose connection to the cameras.



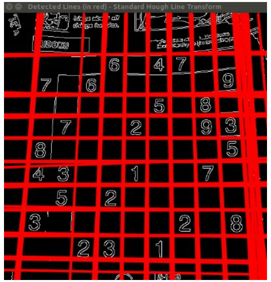
Computer Vision Algorithm

Successful - Object detection 2019 & 2020

The Computer Vision Algorithm



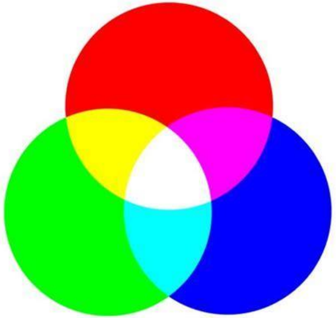
- Two Methods - Contours, and Hough Lines
- Turn down the exposure and gain on the Jevois
- Use bright green LED light ring to light up reflective tape target
- Mask the image received from the Jevois
- Contours - Look for contours in the frame
- Hough Lines - Use Hough Lines function to find tapes
- Find the center of the target and use the X and Y pixel values to find distance and angle

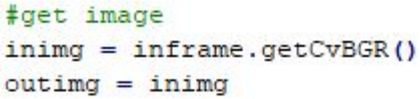


Basic Code - Object detection

Basic Code Pt. 1

- Let’s start by getting a frame from the Jevois
- We will declare the frame that we get from the Jevois as inimg and the frame that we will be sending out as outimg
- We get this image in the RGB (Red Green Blue) colorspace
    - RGB images are defined that each pixel has a 1 byte for Red, Green and Blue Components.
    - RGB values are typically represented between 0-255

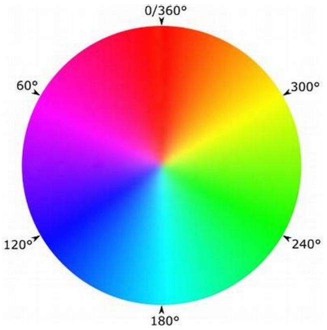


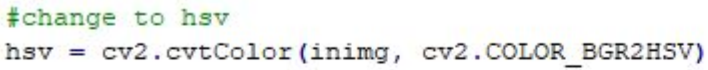


```
#get image
inimg = inframe.getCvBGR()
outimg = inimg
```

Basic Code Pt. 2

- We then convert the RGB image received from the camera into the HSV colorspace
- HSV = Hue, Saturation, Value
  - Hue - The color (red, green, blue etc.)
  - Saturation - How intense a color is
  - Value - How light or dark the color is
- It is faster to filter an image by color in HSV because the computer vision only needs to perform comparisons with one value (hue) as opposed to 3 for RGB

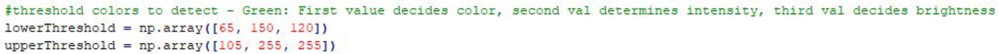




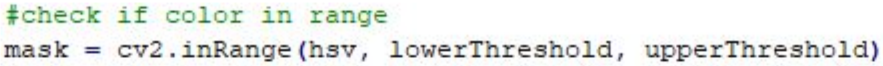
```
#change to hsv
hsv = cv2.cvtColor(inimg, cv2.COLOR_BGR2HSV)
```

Basic Code Pt. 3

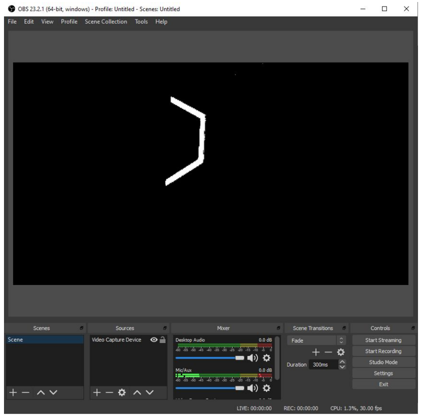
- Next we will create a threshold of colors that we can accept as our chosen color
- Using this threshold, we will create a mask over the HSV image
  - The mask will turn every pixel that is in that range white, and every other pixel black



```
#threshold colors to detect - Green: First value decides color, second val determines intensity, third val decides brightness
lowerThreshold = np.array([65, 150, 120])
upperThreshold = np.array([105, 255, 255])
```

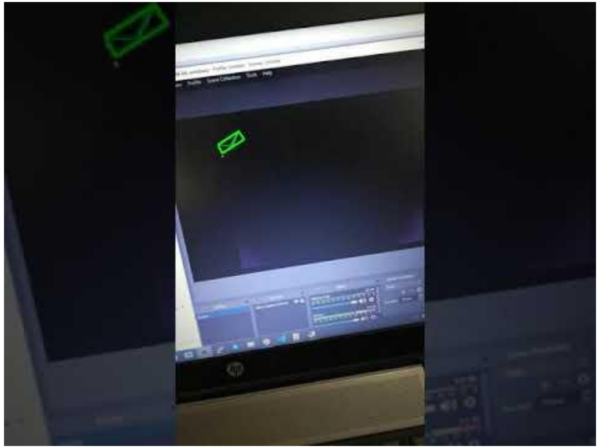


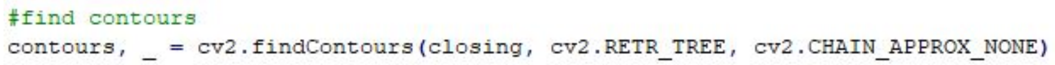
```
#check if color in range
mask = cv2.inRange(hsv, lowerThreshold, upperThreshold)
```



Basic Code Pt. 4

- Next, we look for contours in the image
  - Contours are contiguous regions within the masked image.
- These contours will only be the shapes that appear in white in the image

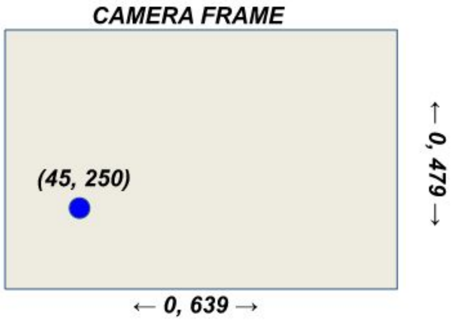


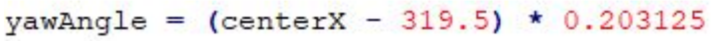


```
#find contours
contours, _ = cv2.findContours(closing, cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)
```

Basic Code Pt. 5

- Once we have found the contours, we have essentially found the vision tape in the frame
- Once we have found the tape, we need to run some calculations to return distance and angle
- The angle is found using the following formula
    - (X pos - centerFrame) * degrees per pixel on the Jevois

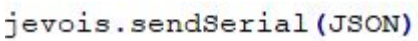




```
yawAngle = (centerX - 319.5) * 0.203125
```

Basic Code Pt. 6

- Once we have the distance and angle values, we need to send them to the robot. This is done using the Jevois serial channel.
- To send data over serial we convert the data to an ascii string using JSON
    - JSON = JavaScript Object Notation
    - {“Distance”: Distance, “Angle”: Angle}
- Then we can send the string over with Jevois.sendSerial()

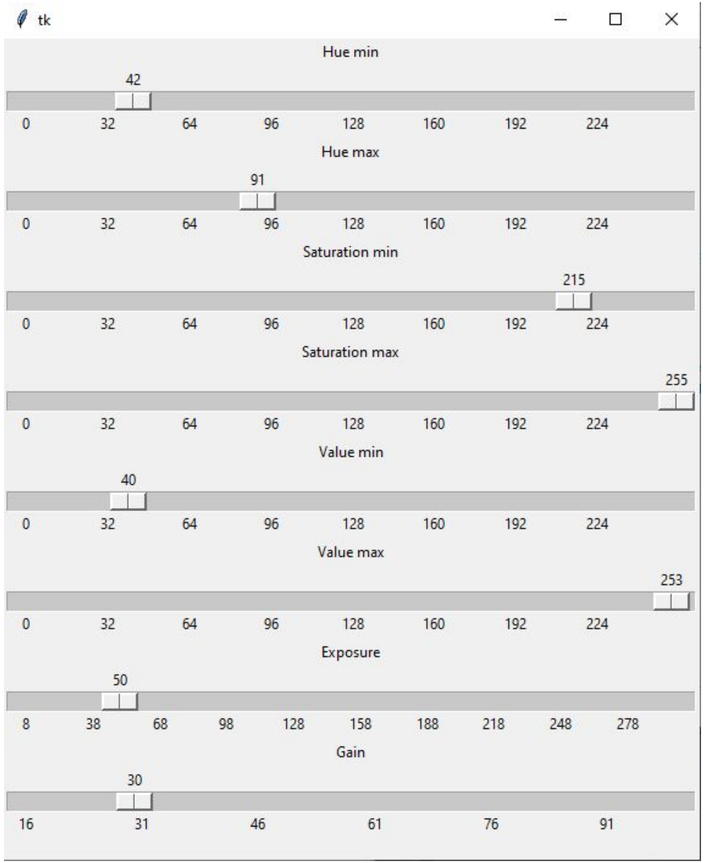


```
jevois.sendSerial(JSON)
```

Tuning HSV

Customized HSV Tuner

- Tuning HSV is one of the most important parts of this entire process
- Constantly changing the values that we have set in code and having to redeploy, only for the values not to work is time consuming and can often be counter - productive.
- We created a python script that will run and change the HSV values from the terminal
- Once the best HSV values are found, we input those values into the actual python program
- Extremely useful in the calibration time at a tournament



tk

Hue min
42

Hue max
91

Saturation min
215

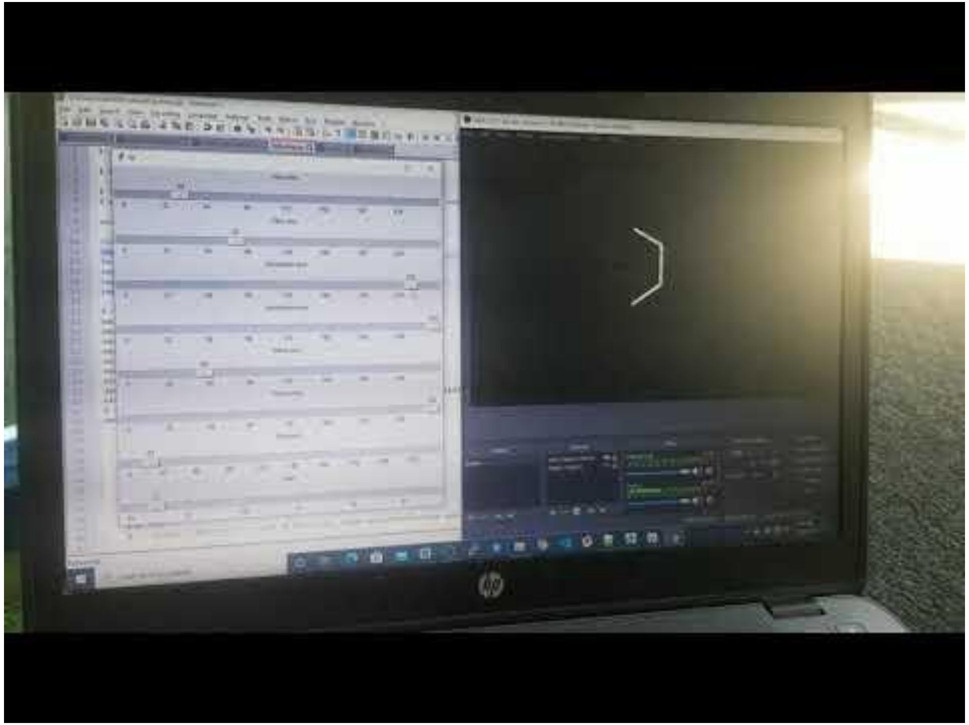
Saturation max
255

Value min
40

Value max
253

Exposure
50

Gain
30



Additional Tips

Helpful Information

- Wiring on the robot is extremely specific, so make sure it is perfect!
- The JeVois uses a few different methods to run code so set up three methods
  - ProcessNoUSB
  - Process
  - UniversalProcess
- Keep all code in UniversalProcess and call that into both Process and ProcessNoUSB
  - The Jevois will automatically call Process or ProcessNoUSB based on whether the Jevois is used on a robot or a computer
- When on the robot, make sure the serial is only sent to the serial port when on the robot
- White paper from FRC #2073 - Link to a great paper on using the Jevois in FRC

Running Multiple Driver Vision Cameras

- Running multiple cameras requires an additional step
- Plugging in the sd card into the computer, add a blank text file called multicam to the BOOT: portion of the drive
- This helps to lower the bandwidth used by the Jevois and lets multiple Jevois' run on the robot
- Plugged into the USB Hub and power buck using Y - split cable
- Driver Vision Camera Image displayed on Shuffleboard

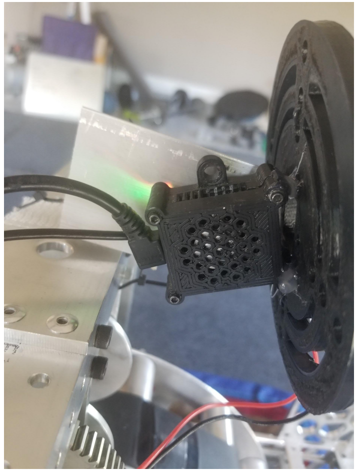
Potential Pitfalls

We made mistakes so you don't have to!

- Jevois Inventor
    - Programming the Jevois with Jevois inventor makes the Jevois dependent on being able to connect to Jevois inventor.
    - This causes problems when the Jevois is running without inventor
- Power bucks
    - Make sure to plug power bucks into 12 V rail on VRM, or directly into PDP.
    - When plugged into 5 V rail on VRM we found that the Jevois has multiple power failures
- Don't put computer vision Jevois on a USB hub
    - USB hubs sometimes mess with the port ID of the Jevois on the RIO. This sometimes prevents the rio from properly connecting to the Jevois
- The 2 USB ports on the roboRIO are enumerated differently. Always use the same port for the Jevois
- Make sure Driver vision cameras have serial out set to none, otherwise there will be issues
- Make sure Driver vision cameras are running at low resolutions, otherwise they will saturate the USB bus.

Lessons Learned

- Light Ring - Using relay, don’t leave on too long
- Fisheye Lens - Better lens for driver vision
- Booting Up Jevois with USB Hub - Computer Vision first, driver vision second
- Make sure Source Control is important!
- Keep the Jevois in one spot on the Robot (angle and distance)
- Hot glue the serial cable into the serial connector on Jevois
- Creating new cases - model on GrabCad
- Mount the Jevois horizontally and get a larger range of view



References

Help Documents

- http://jevois.org/doc/UserConnect.html
- http://jevois.org/doc/UserCli.html
- http://jevois.org/start/
- https://docs.opencv.org/master/d9/df8/tutorial_root.html

Jevios Products

- https://www.jevoisinc.com/collections/jevois-hardware/products/jevois-a33-smart-machine-vision-camera?variant=36249051658 - Jevois
- https://www.jevoisinc.com/products/jevois-1-3mp-sensor-with-120deg-fisheye-lens - Fisheye Lens

Code

- https://github.com/HighlandersFRC/2020-Chronostar/tree/master/VisionCode

Questions?

Feel free to email me at: adarsh.payyakkil@gmail.com