<u>ESE 421: Project Status (11/10/18)</u>

Adnan Jafferjee and Saumeel Desai

Hi there!

If you are reading this, you are probably trying to evaluate the progress Saumeel and I have made thus far on autonomously navigating Penn Park. We aren't quite there yet, but here are some highlights of what we got:

- We can reliably identify the desired edge of the road using computer vision. Sometimes we get multiple lines. We're working on only ever getting the right line. Check out **HSV_CV.py**
- We implemented a complementary-filter driven heading control algorithm. This was tested to be functional with GPS and inertial rate as inputs to the filter, but since then we changed the filter to swap GPS with the camera estimated heading. We haven't tested it with the camera heading yet, so there are likely bugs. We also need to tune our gain (K) and filter time constant (tau). Check out **headingControl.ino**
- We were able to set up i2c communications between the Pi and Arduino to use the joystick to control car speed, steering as well as control when the Pi takes a picture! Check out **joystickControl.ino** and **SendJoystickCommandsToArduino.py**
- **PathFinder.py** finds the closest node point to a given a GPS location from a dictionary of nodes on a map.

HSV_CV.py

HSV_CV.py is our most functional road detection Python script. It does the following:

- Takes in an image of a road
- Converts it to HSV color space
- Applies lower and upper thresholds that separate regions with colors (defined in HSV space) that are similar to the road into a mask[1]
- Performs Canny edge detection on a selected region of interest (bottom right corner of the original image) of the mask
- Uses Hough line detection to pick out desired lines from the image
- Returns calculated Ψ_R and X_0
- Plots detected line on image, as well as plots of intermediate detection steps

We still need to implement a heuristic to chose one line (as opposed to several) from the detected edges. Likely picking the longest line is a good enough "dumb" heuristic for right now. We also need to send the best estimated line Ψ_R and X_0 to the Arduino for heading control.

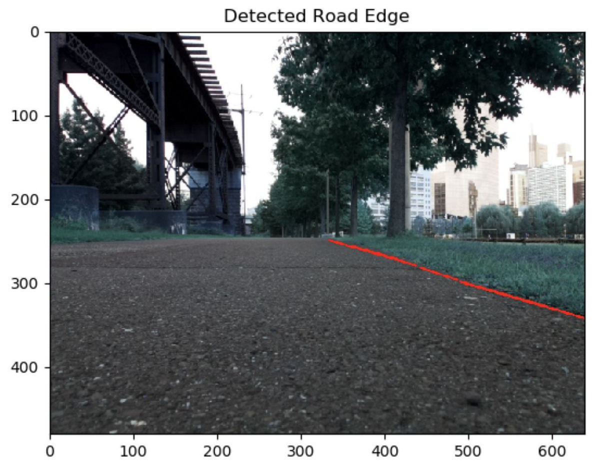
[1] The thresholding portion was adapted from code that Ryan Kortvelesy shared on Piazza. Thanks Ryan!

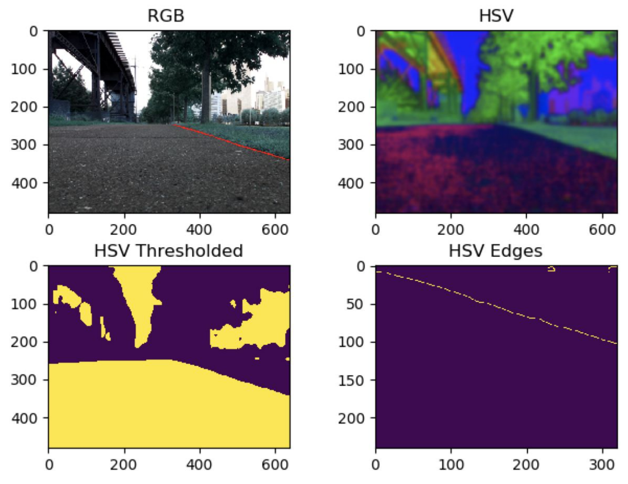
Some Results

Joystick Control Functioning

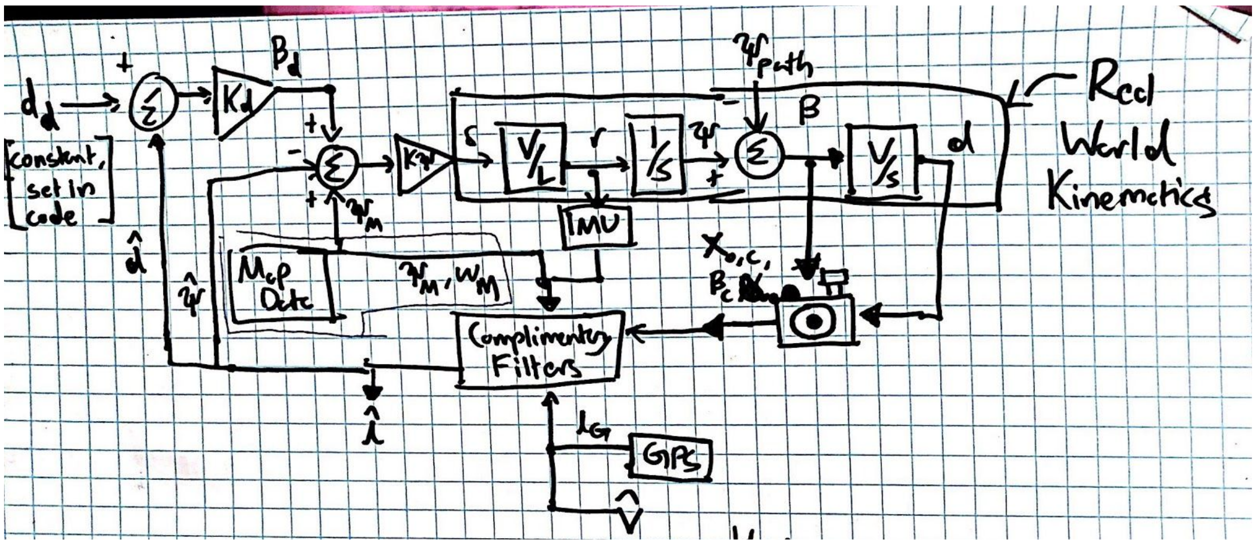
Heading Control Functioning (Defaults to Gyro when no GPS detected)

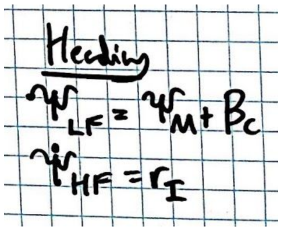
Some example outputs of our HSV_CV script





Estimated Offset: 45.07
Estimated Psi_r: 23.59

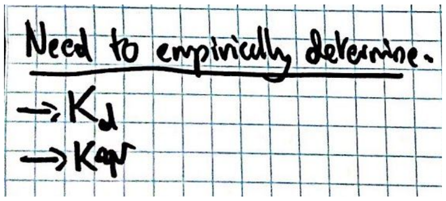




Heading

$\psi_{LF} = \psi_M + B_c$

$\psi_{HF} = r_I$



Need to empirically determine.

$\rightarrow K_d$

$\rightarrow K_\psi$

Control Diagram of our (desired) heading control. We still need to implement map data integration