Raspberry Pi Tracker

Matthew Flaherty, Jackson Foster, Alexander Jiao

Agenda

Goal/Motivation

Design Overview

Object Identification

Object Tracking

Servo Control

Challenges/Limitations

Future Work

Goal/Motivation

Goal:

- To develop a mobile, low-power, low-cost system for human motion detection/tracking using OpenCV library.

Motivation:

- Achieve high performance with technology readily accessible to the public at low cost (~ $30 Raspberry Pi 3B, ~$25 8MP camera module, ~$2.50 servo x 2)
- Intersection of many fields→ Computer Vision, Control Systems, High Level Programming, Optimization, Embedded Hardware

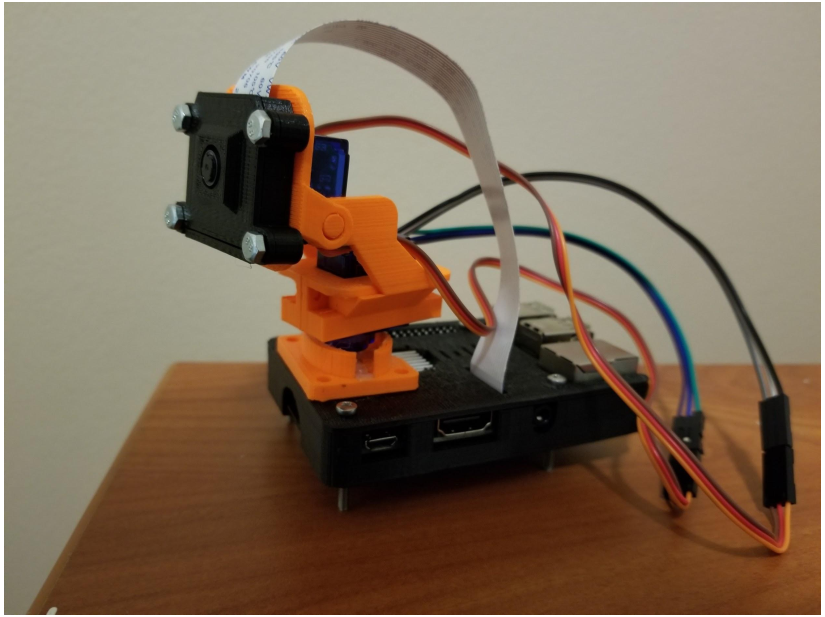
Project Overview

Project consists of several components

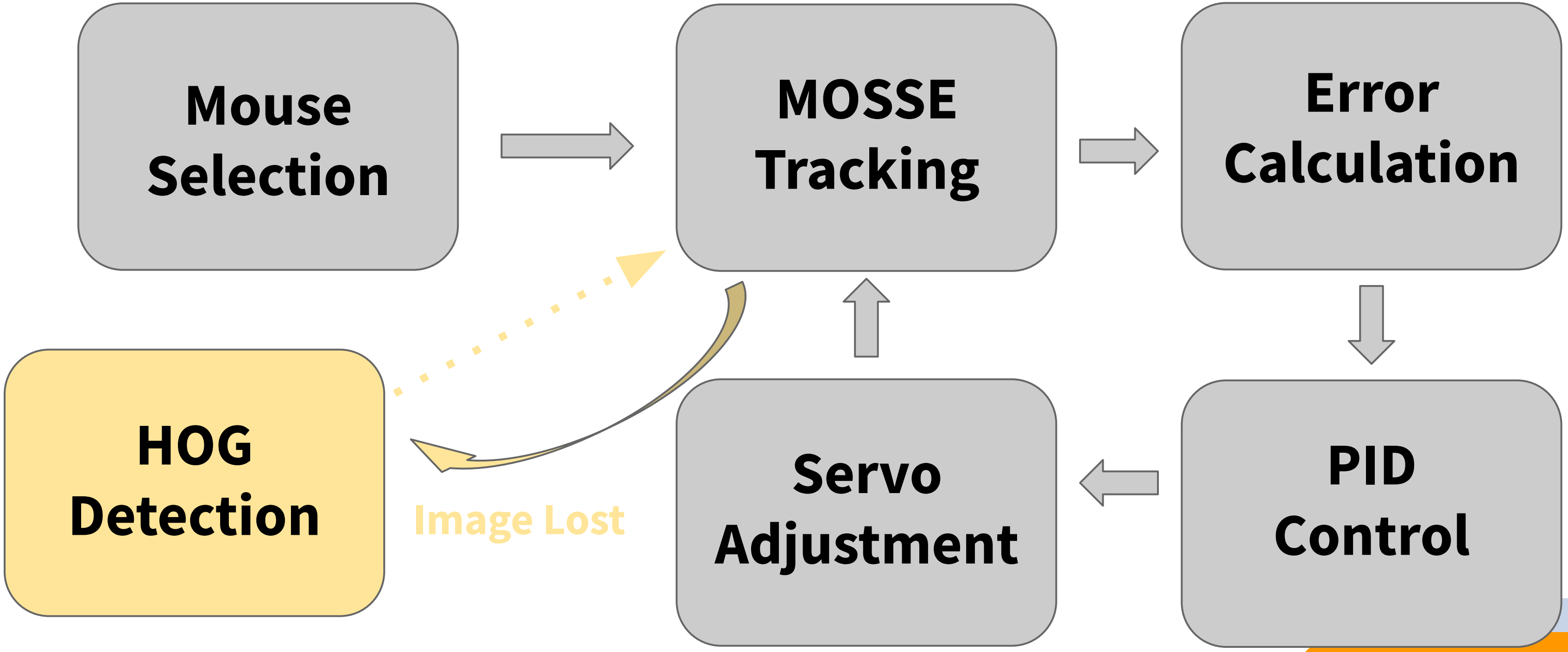
1. 3D printed gimbal and camera mount
2. Object Identification
3. Object Tracking
4. Servo controls

Servo Gimbal

- Designed for use with SG90 180° rotation servos
- Pi Camera Module V2
- Cutout for heatsink and cooling fans
- Modular to swap out servos quickly and easily



Software Flowchart



Object Identification

HOG: Histogram of Oriented Gradients- most promising method for detecting pedestrians on screen

- Poor speed (< 1 FPS) which made frame by frame computation unfeasible
- Idea: use HOG to find a bounding box and feed it to object tracking algorithm. Only recompute bounding box if it is lost or object becomes static for too long
- Did not work well with object trackers

Object Tracking

Tried object tracking with multiple OpenCV functions

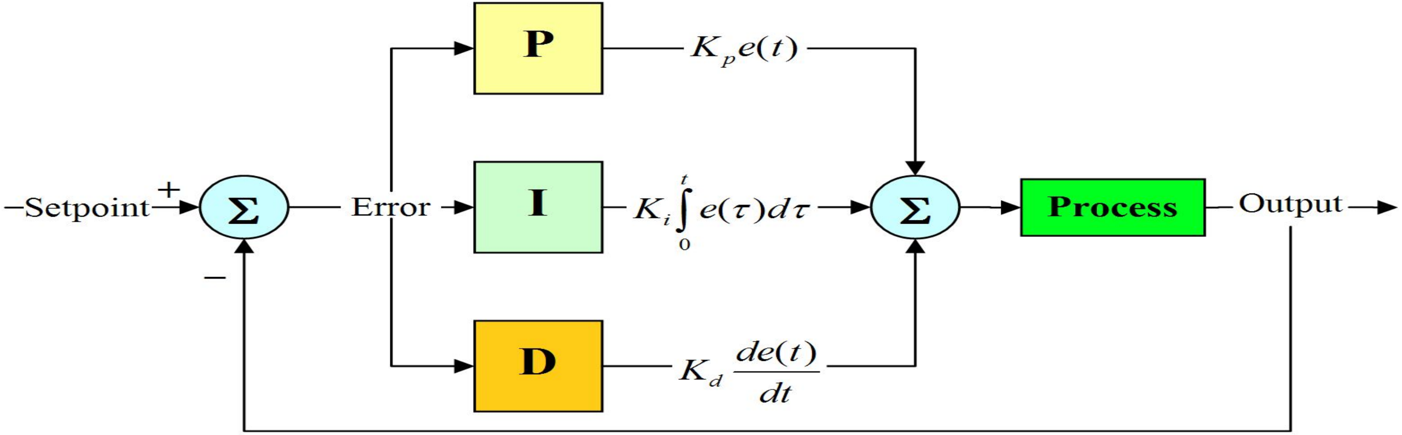
1. KCF - Kernelized Correlation Filters - initially thought this method would provide the best performance to speed trade off
   a. Poor speed (~ 1 FPS) which made object tracking impossible
2. MOSSE - Minimum Output Sum of Squared Errors - The method providing much better performance and speed
   a. Good speed (~ 23 FPS) allows for object tracking at reasonable distance
   b. Updates bounding box mask with every frame to allow for changes in light and object shape

Servo Controls

Control servo with PID (Proportional Integral Derivative) controller for smoother operation

- Proportional constant controls how far the servo moves based upon current error
- Integral constant control adds some damping based on sum of previous errors
- Derivative constant adds damping based on most recent error

Tune parameters empirically using Ziegler-Nichols method



Limitations/Challenges

- Several iterations required to find servos and 3D CAD design that mesh well together
- Small, cheap components to construct = HUGE time sink
- Creating the software environment→ package installation, lightweight text editors, system administration on the Pi
- Rapid servo adjustment = blurry images → poor MOSSE performance
- Overheating the Raspberry Pi
- Robustness to Occlusion

Room for Improvement

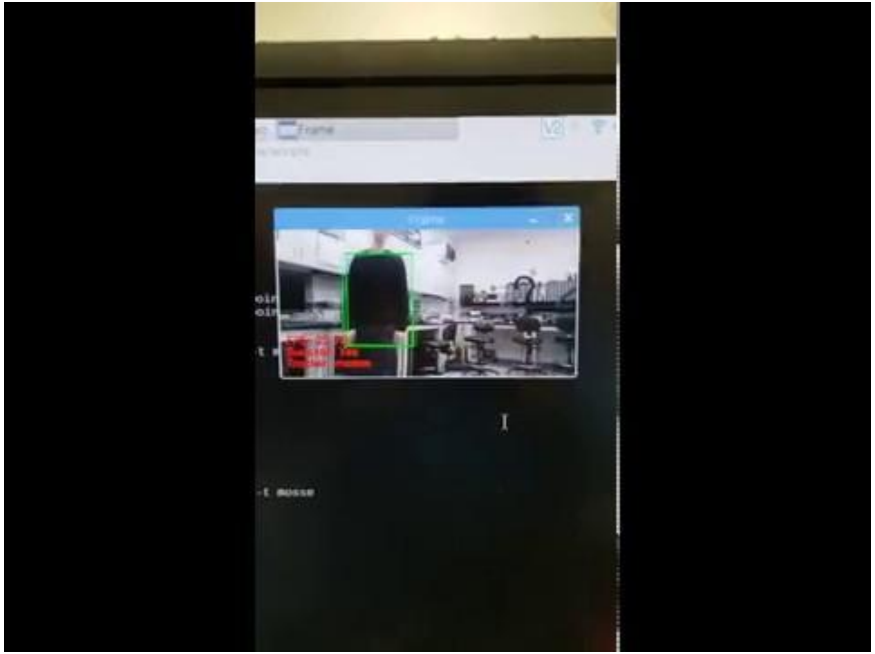
- HOG detection to MOSSE tracking transition
- Optimize controller coefficients
- More robust gimbal design
- Optimize Resolution vs Frame Rate
  - Multi-thread PID controller + MOSSE tracker

Future Work

Jetson Nano ($99)

- Increase image processing power
  - Use CNN’s for object/human specific tracking
- Applications:
  - line of sight beam steering, animal observation, security cameras, sports videography, spot light operation, monitoring shuttles docking on the I.S.S. autonomously

Video Demonstration



THANK YOU!

Questions?

Sources

[1]

D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2544–2550.

[2]

Thingiverse.com, "SG90 Servo 2-axis Gimbal by aanarcissus." [Online]. Available: https://www.thingiverse.com/thing:2892903. [Accessed: 13-May-2019].

[3]

Thingiverse.com, "Raspberry B+ housing, camera housing and servo gimbal for camera by Fido." [Online]. Available: https://www.thingiverse.com/thing:504196. [Accessed: 13-May-2019].

[4]

"pigpio library." [Online]. Available: http://abyz.me.uk/rpi/pigpio/python.html. [Accessed: 13-May-2019].

[5]

"Pedestrian Detection OpenCV - PyImageSearch." [Online]. Available: https://www.pyimagesearch.com/2015/11/09/pedestrian-detection-opencv/. [Accessed: 13-May-2019].

Sources

[6]

“Pan-Tilt Multi Servo Control,” *Hackster.io*. [Online]. Available: https://www.hackster.io/mjrobot/pan-tilt-multi-servo-control-b67791. [Accessed: 13-May-2019].

[7]

J. G. Ziegler and N. B. Nichols, “Optimum Settings for Automatic Controllers,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 115, no. 2B, p. 220, 1993.

[8]

A. Rosebrock, “OpenCV Object Tracking,” *PyImageSearch*, 30-Jul-2018. .

[9]

J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-Speed Tracking with Kernelized Correlation Filters,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[10]

“Going Straight with PID - Introduction | Raspberry Pi Projects.” [Online]. Available: https://projects.raspberrypi.org/en/projects/robotPID. [Accessed: 13-May-2019].