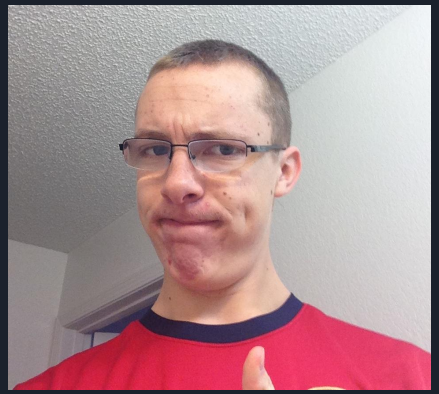
Pong Innovation

Better Ping Pong Through Science

Team



Brian Foley

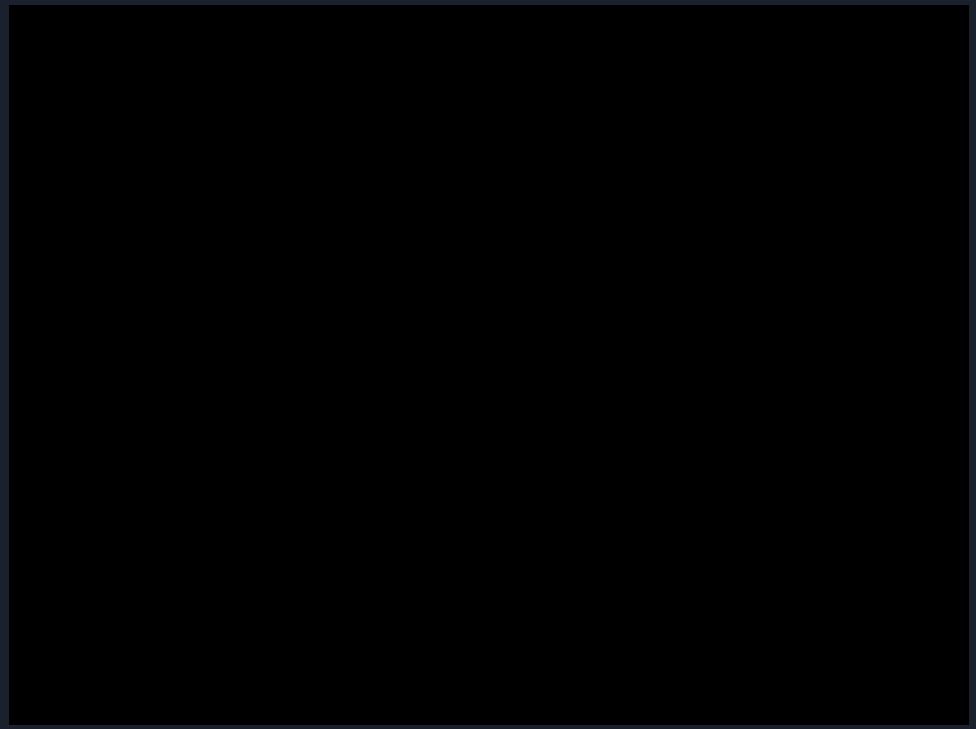
Austin Kauble

Karl Nuetzel

Project Description

Utilizing computer vision and machine learning to handle the least fun part of ping pong, keeping score, while providing statistical insights

Demo



Implementation - Stack

Match Referee:

- YoloV3
- Darknet
- Python

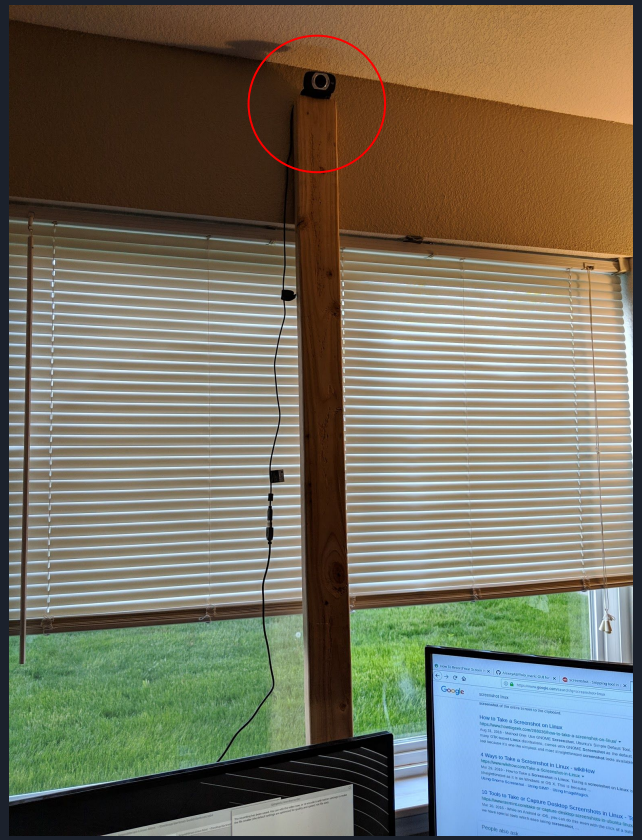
Stats Server:

- Bootstrap
- BoltDB
- Go

Project Management

- Agile... or not
- Simpler task tracking with schedule
- Open communication with group messaging
- Pair programming

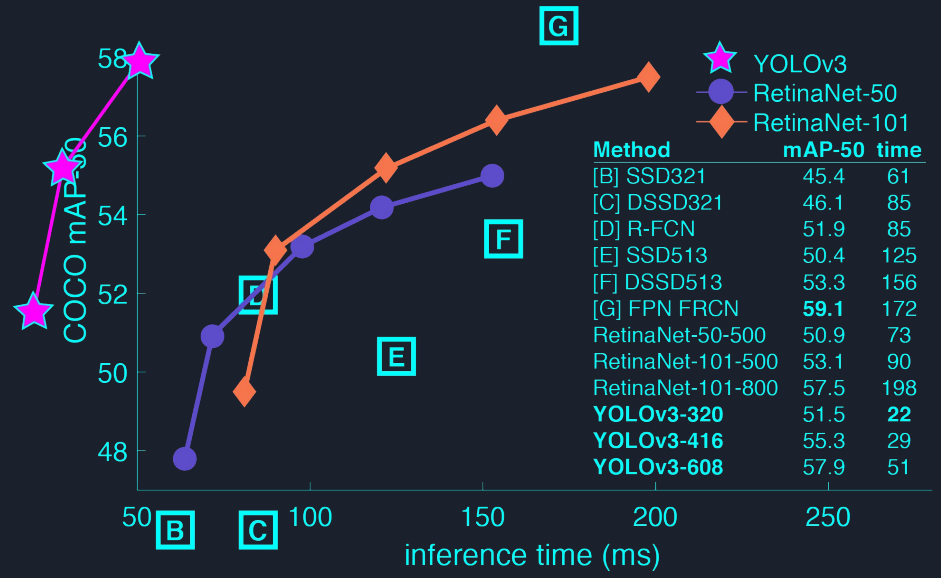
Implementation - Hardware



- Logitech - Full HD Portable Webcam
- Geforce GTX 1070ti
- 9 foot 2x4 (camera mounting apparatus)
- 2 Linen Sheets
- 5000 Lumen Light

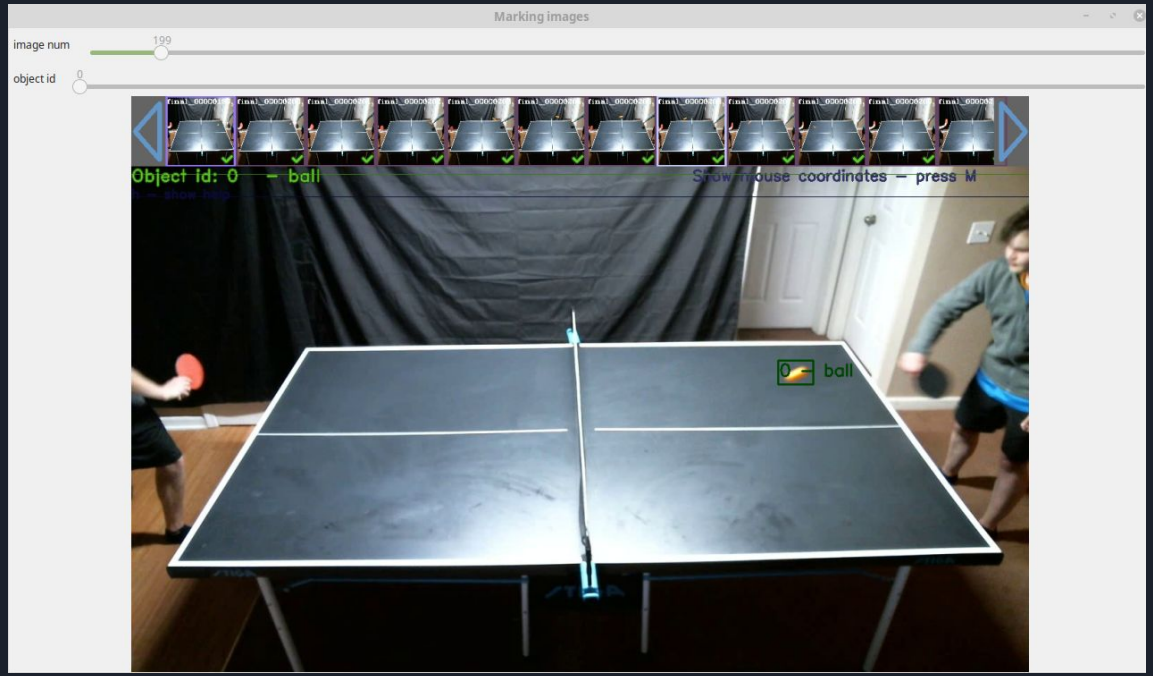
Implementation - Tracking the Ball

- YoloV3
  - You only look once (YOLO) is a state-of-the-art, real-time object detection system
  - Large Dataset (2000+ images per object for training)



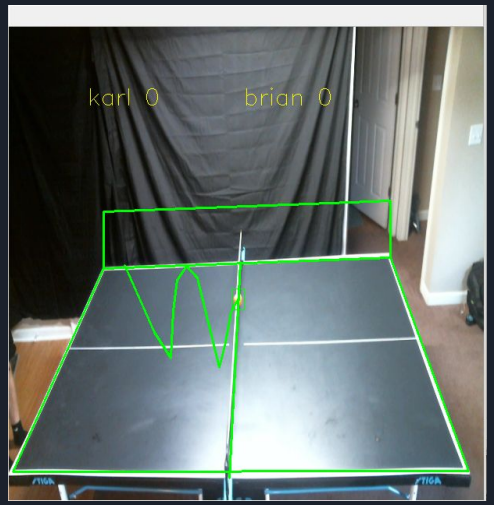
Implementation - Building Dataset

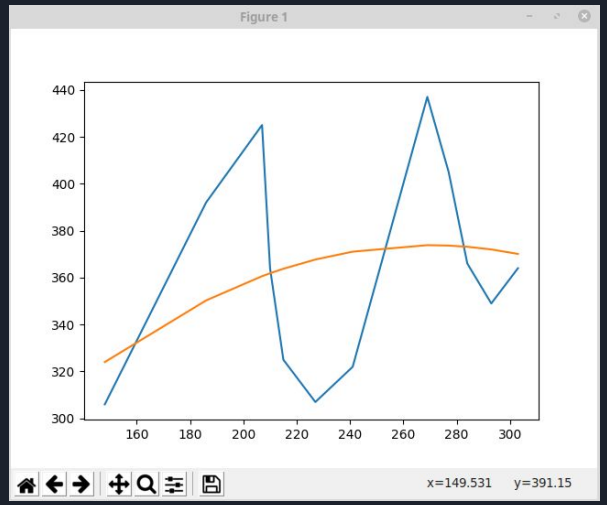
- Yolo Mark - Label bounding boxes
- Split up video into frames to draw the boundary boxes around object



Implementation - Side/Bounce Detection

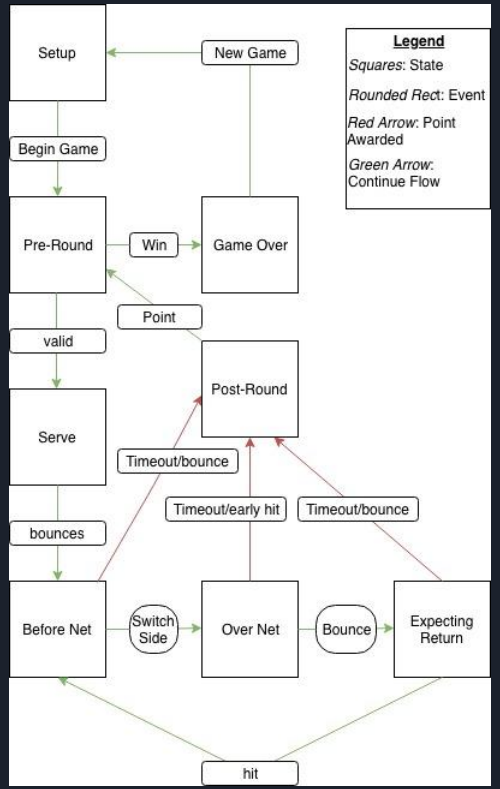
- Manually draw in boundaries of the table and the net
- Use the equation of the line that was drawn to determine what side the ball is on and if it is within the table boundary
- Bounce Detection is handled by storing the previous locations of the ball (x, y) in a list until the ball changes sides or timeout
    - Find change in direction by parsing the list





State System and Scoring

- Keep track of valid bounces to determine state of game
- Timer starts when the ball changes side (except on serves)
- Expected return when the timer starts
- Score points on timeouts



Stats Server

Pong Innovation Matches

Time	Player 1	Player 2	P1 Score	P2 Score
2019-05-05 16:35:21.801727 -0500 CDT	Brian	Alex	10	6
2019-04-28 21:09:12.354845 -0500 CDT	play1	play3	4	10
2019-04-28 20:53:22.536735 -0500 CDT	play1	play3	10	9

- Simple REST API
- Hands out match IDs to clients
- Stores all matches in a database (BoltDB)
- Updates the score of a match at the completion of the game
- Presents a match history interface

Testing

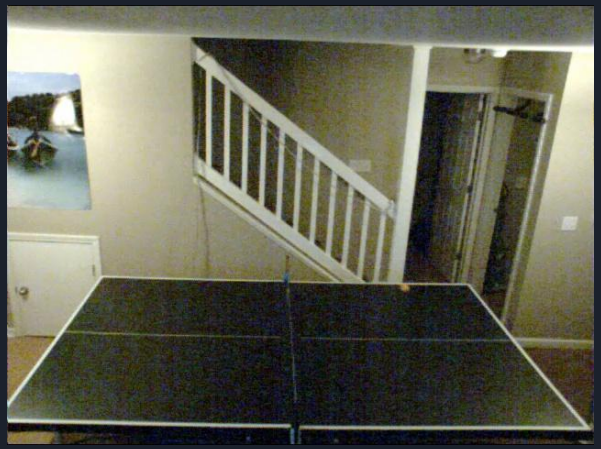
- Testing is challenging due to the highly variable nature of the game and tracking system
- Ability to feed captured video back into the system, which lets us test particular problematic cases
- A *lot* of manual iterative testing

Challenges - Tracking Method

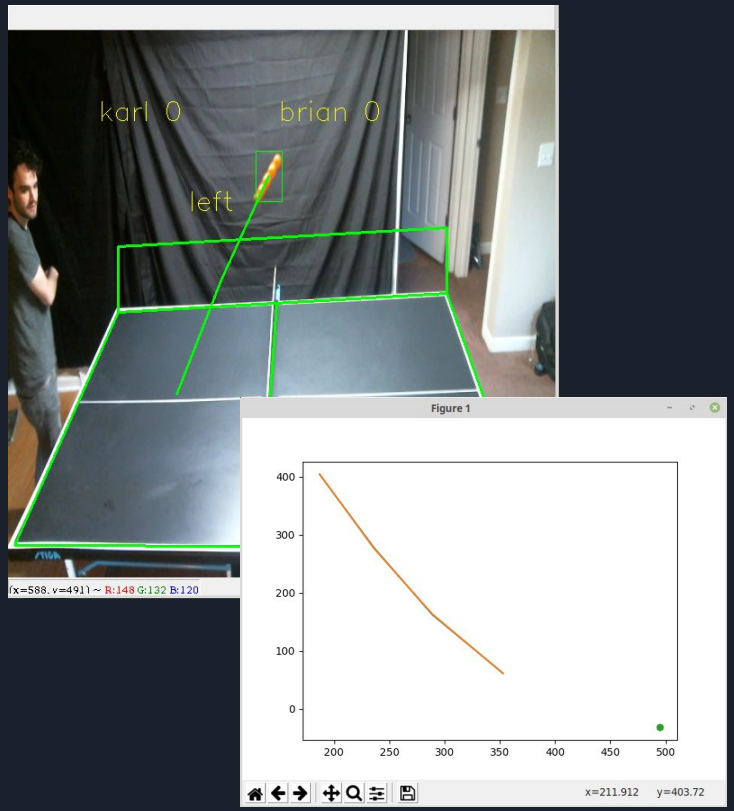
- Simple color detection inadequate at detecting ball due to lighting, camera quality, and the speed of the ball
- Building dataset for machine learning approach is time consuming
- Original datasets only consisted of 300-400 images and produced inaccurate detection of ball

Challenges - Camera and Environment

- Camera capability
  - PS3 eye camera boasts high frame rate, but inadequate fidelity
  - Single vs dual camera
- Lighting
  - bright, even lighting
  - consistency throughout the day



Challenges - Bounce Detection and Edge Cases



- 2D representation of 3D environment
- Parabola fitting (change in direction)
  - Always detects a min (location of bounce)
  - Too sensitive when bounce does occur
  - Occasionally only receive two data points before crossing sides (unable to verify if a bounce occured before crossing)
- Single low-budget camera
  - Limited GPU
- Alternate approach (sound)

Future Work

- Improving detection of edge cases and fine-grain ball tracking
- Opportunity for interesting statistics from historical match data
- Lowering the computational requirements

Questions?