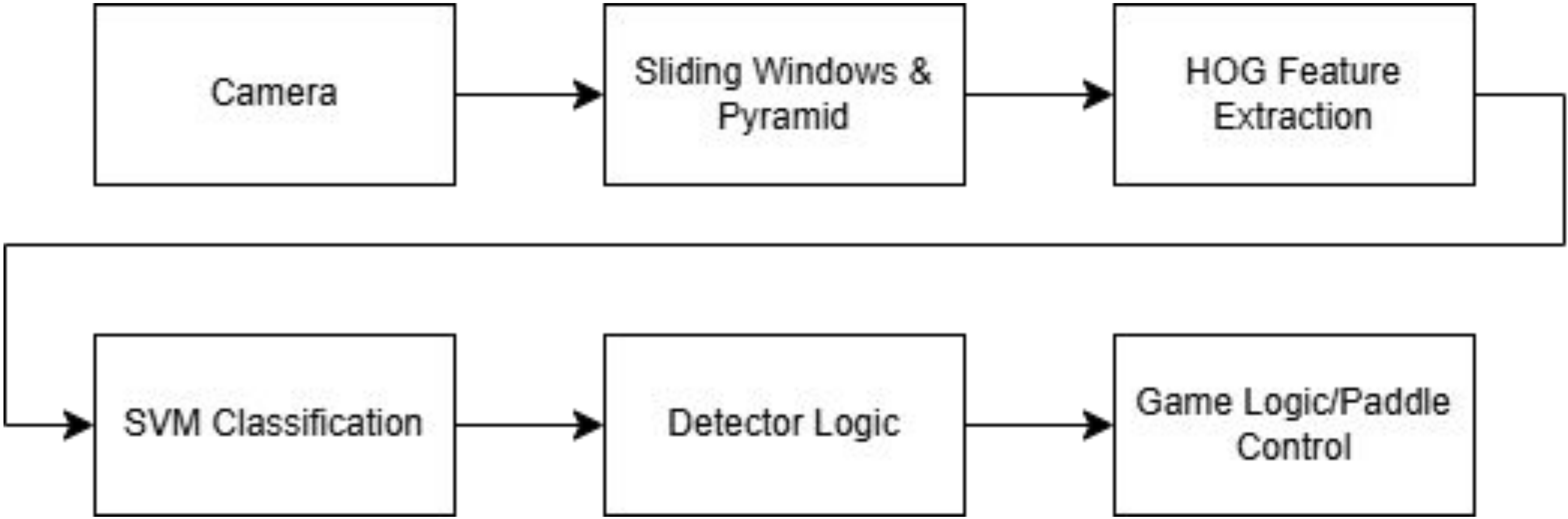
Hand-Controlled Ping Pong Game w/ HOG + SVM

Sharmarke Ahmed, Shin Thant Maung Maung
Robot Vision Fall 25
12/9/25

Reimagining Traditional Ping Pong Game

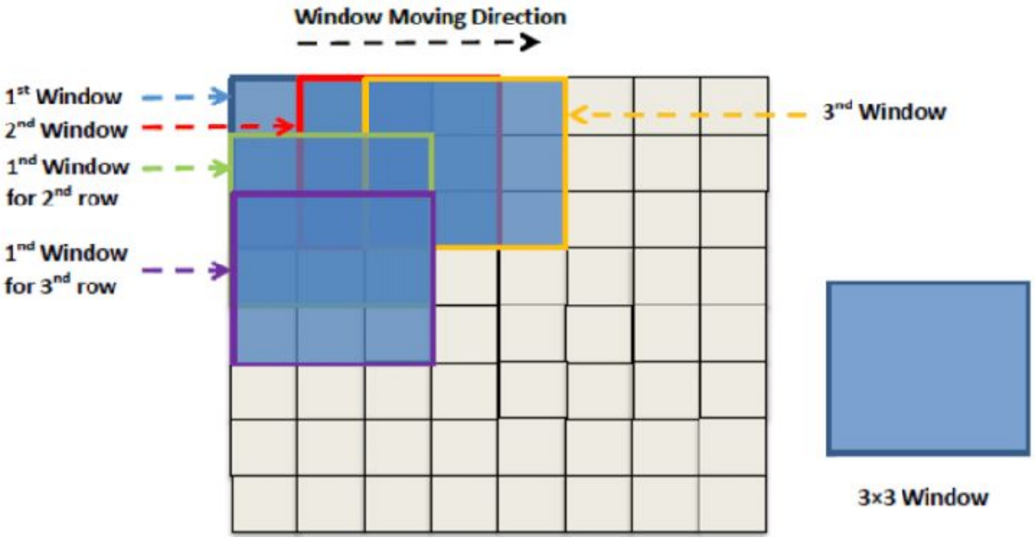
- **Motivation:**
  - Learn more about implementation of HOG + SVM object detection algorithms
  - Explore the efficiency and effectiveness of the algorithm
- **Goal:** Replace mouse/keyboard input with hand input
  - Detect finger/hand in live webcam feed
  - Map hand motion -> paddle motion
  - Maintain useable frame rate
- **Roles:**
  - Sharmarke -> GameUI & Design, Manual Hand/Finger Input Model
  - Shin -> 11k Hand Model

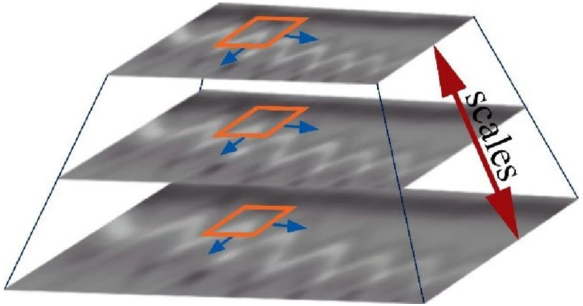
High Level Overview



Sliding Window

- **Sliding Window:** Move a 64x64 window across the image
- **Image Pyramid:** Scale image to three different sizes (small, medium, large)
- Extract HOG features from each window and get SVM confidence score
- High score -> Hand detected!

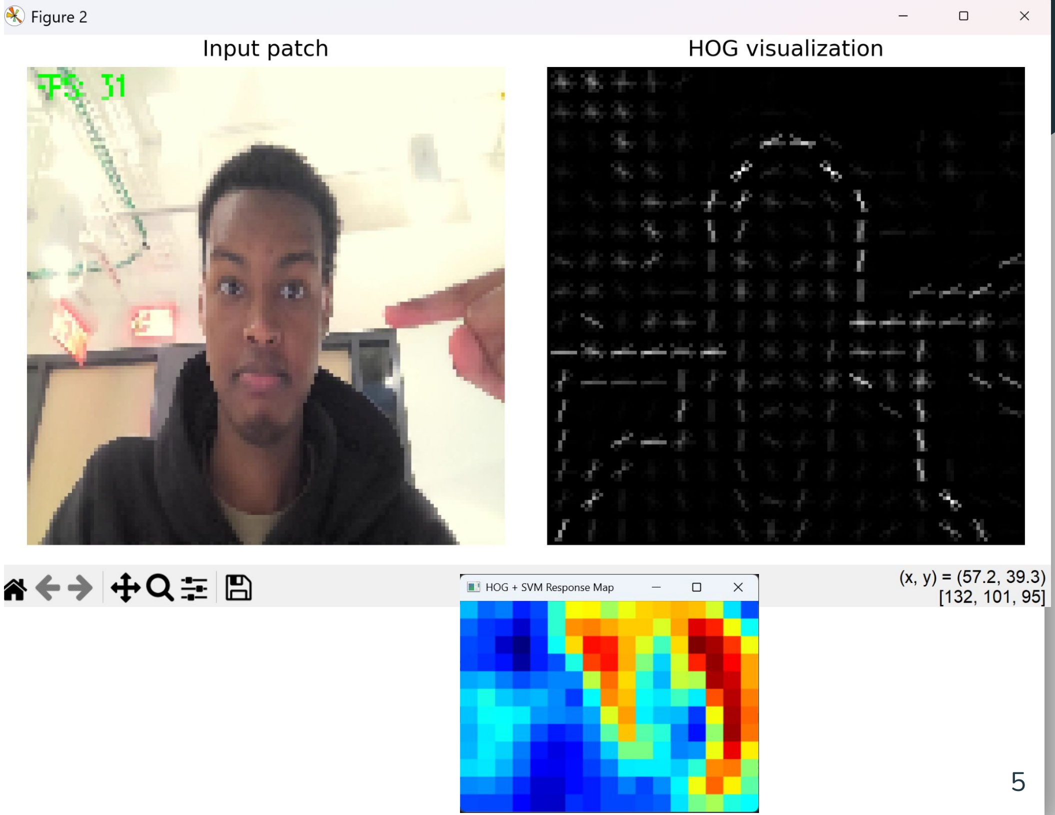




HOG Feature Extraction

HOG Parameters:

- Uses OpenCV HOG functions
  - Window: 64x64
  - Cells: 8x8
  - Blocks: 16x16 stride 8
  - 9 orientation bins
- Converts frame into features
- Good for capturing shapes

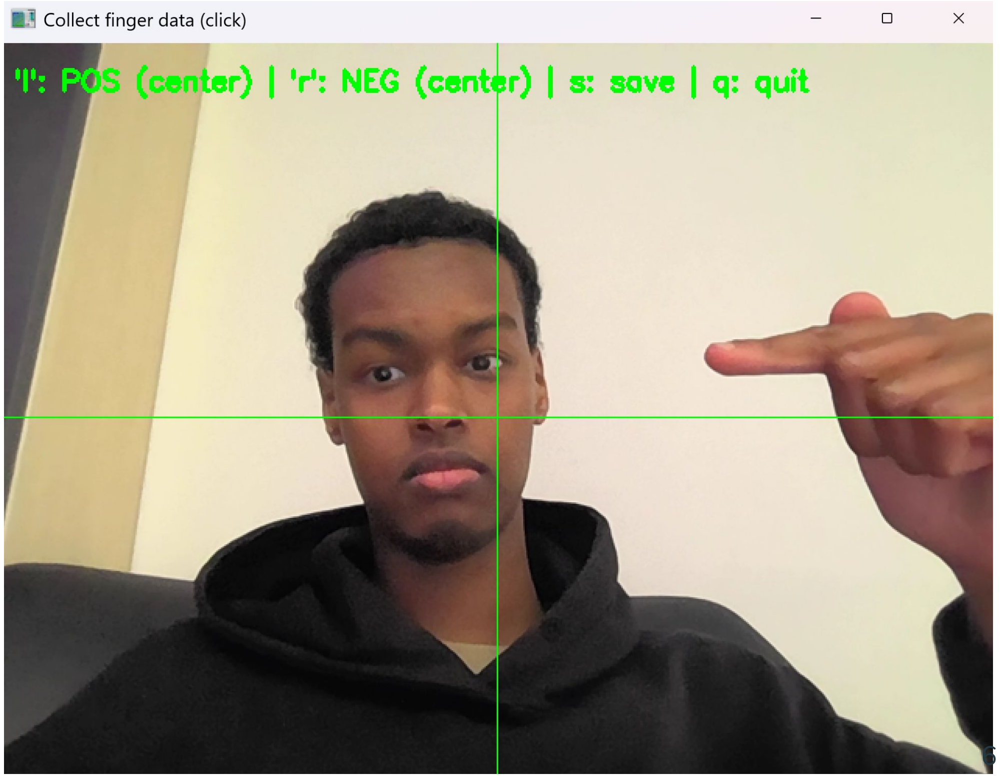


```
(x, y) = (57.2, 39.3)
[132, 101, 95]
```

Training SVM: Data Collection (Manual Hand Input Model)

- **hog_finger_data.py**
  - Store positive & negative HOG finger samples in a joblib file

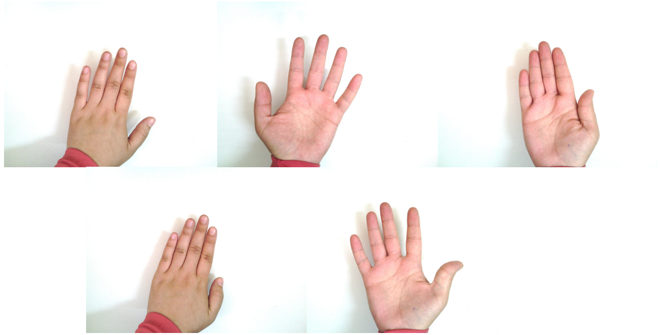
- **train_finger_svm.p**
  - Using LinearSVC() from sklearn library
  - Run SVM on positive & negative samples to be used for finger detection



Collect finger data (click)

'l': POS (center) | 'r': NEG (center) | s: save | q: quit

Training with External Datasets



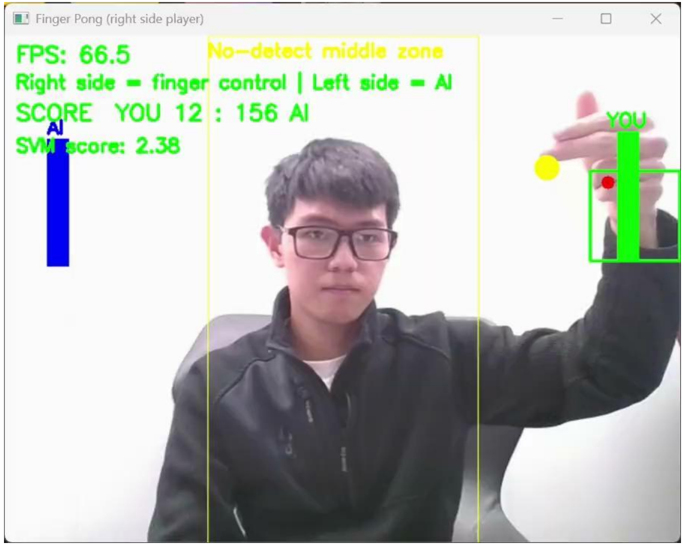
Positive Images

(100 images from 11k Hands dataset)

Negative Images

(50 images from Imagenet dataset)

Demo Videos



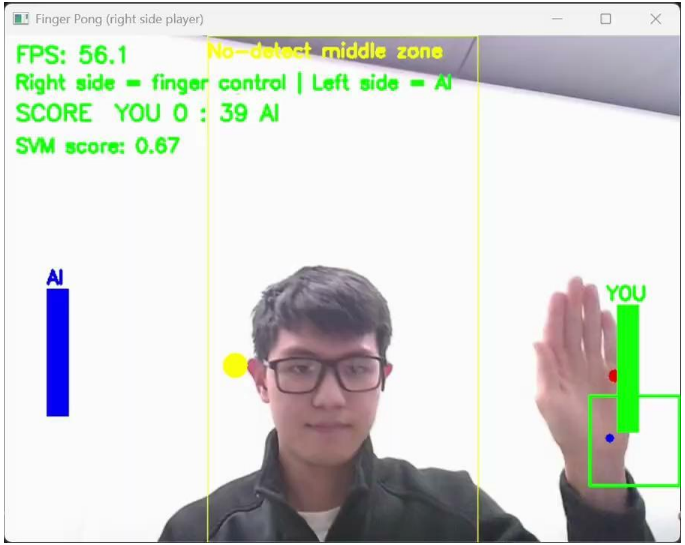
FPS: 66.5

Right side = finger control | Left side = AI

SCORE YOU 12 : 156 AI

SVM score: 2.38

Trained with Manual Hand Data



FPS: 56.1

Right side = finger control | Left side = AI

SCORE YOU 0 : 39 AI

SVM score: 0.67

Trained with 11k Hands Dataset

Main Challenges

- Finding hand datasets with complex background
- Highly dependent on a clear background
- Poor detection with different hand gestures and orientation

Further Improvements

- Background robustness
- Orientation invariance
- Game UI and Complexity

Thank you!
Q&A