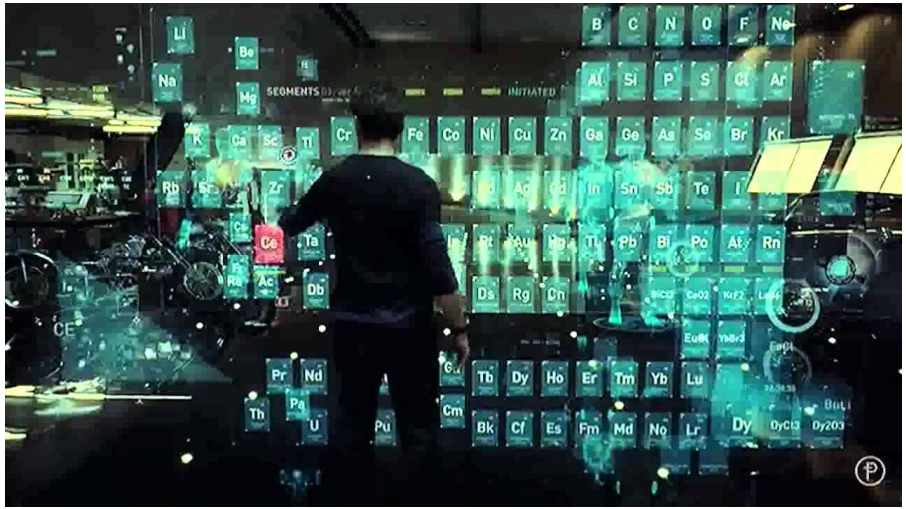
Virtual Whiteboard

Alan Song, Andrew Huang, Brian Lane

Introduction to Project

- Control a computer with touchless gestures
- Hand and finger movements tracked by sensors



Use cases

Interact with a desktop environment without direct contact.

Useful for:

- reducing eye strain by being further from a screen
- presentations or demonstrations
- interactivity with audience or environment
- sanitary

Use a limited language of gestures to perform different UX operations.

1. Movement of cursor
2. Right and left click
3. Holding and dragging
4. Scrolling

Software Systems + Signal Processing + Machine Learning

Use cases cont.

Closed fist (0 fingers) should not update cursor position, used to reposition hand

2 fingers for right clicking

1 finger for left clicking

Keep it held up for left click hold

Open palm (5 fingers) to move cursor around

Requirements

Smooth user experience

Recognize gestures from a distance of between 3 feet and 15 feet.

Recognize gestures with at most 33 ms latency.

Accuracy of clicking gesture recognition approaching 99%.

Accuracy of cursor motion within 30x30 pixel

Technical Challenges

- Integration of external camera data with UI/OS
- Accurate detection of user movement at (variable) distance.
- Accurate detection of small hand movements.
- Speed of calculation.

Risk mitigation

- Setting aside a lot of time in schedule for experimenting with different options for integration with hand position and gestures

Solution approaches comparison

IMU

- Low accuracy compounds over time[1]

Infrared

- Resolution: 0.2 mm^2 - 5.3 mm^2
- Only gets location of sensors, need more complex algorithms to determine gestures

Ultrasonic

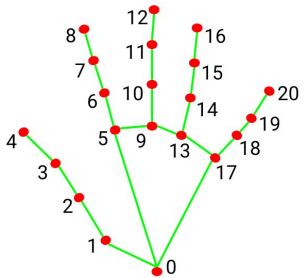
- Resolution: ~1cm^2
- No sensors needed, complex calculations
- 3D

Computer vision

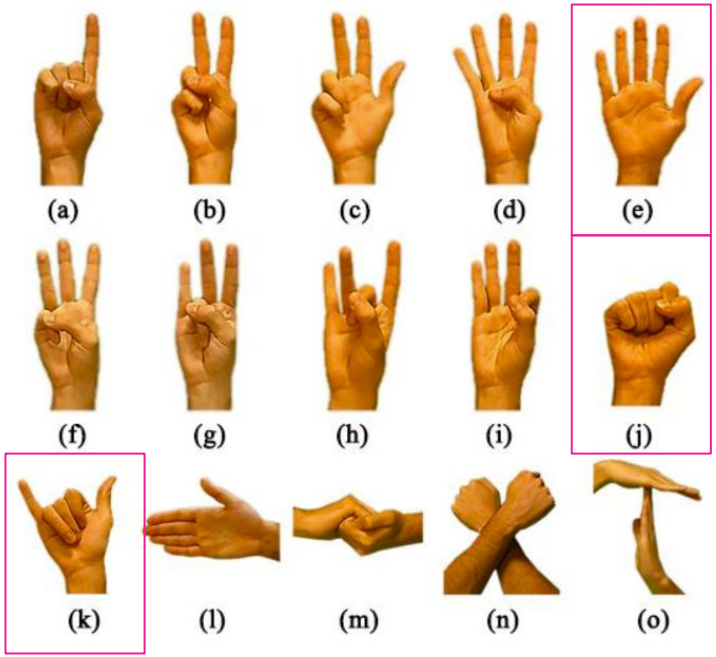
- Resolution scales with camera & distance
- Accuracy approaches >95%
- Speed dependant on model complexity
- Lots of previous work done, intersects most with our experience as a group

Computer Vision

- Hand location tracking in real time with pose estimation[1]
- Gesture recognition with Deep CNN
  - Dataset
- MediaPipe library for hand detection and labeling

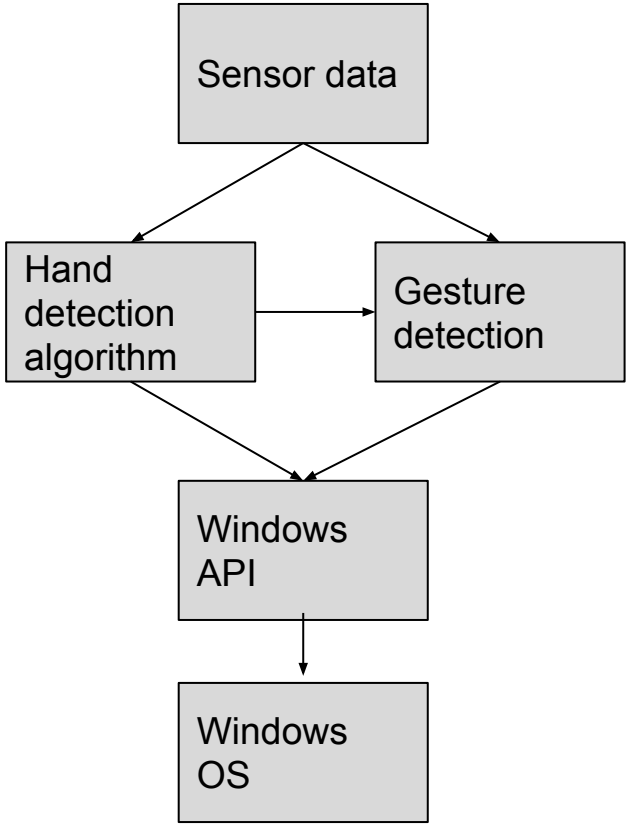


0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP



Implementation Specifics

- Use Windows features:
  - Middle mouse scrolling
  - Click detection
  - Mouse movement
- Python libraries
  - mouse
  - win32api
  - Win32con
- Range of motion and distance through bounding box



Testing/verification of requirements

Testing of device from many locations within range of distance (and outside)

A set pattern of motion and gestures, test for gesture accuracy and latency

Test on a page of shapes of different sizes to measure pixel accuracy

Division of Labor

Alan

Windows Cursor API

- Mouse library for mouse functions (positioning, clicking, etc.)
- Translating coordinates from hand detection to cursor movement in Windows OS
- Incorporating gesture data to program cursor functionality (clicking, scrolling)
- Helping others with the detection tasks

Andrew

Hand detection

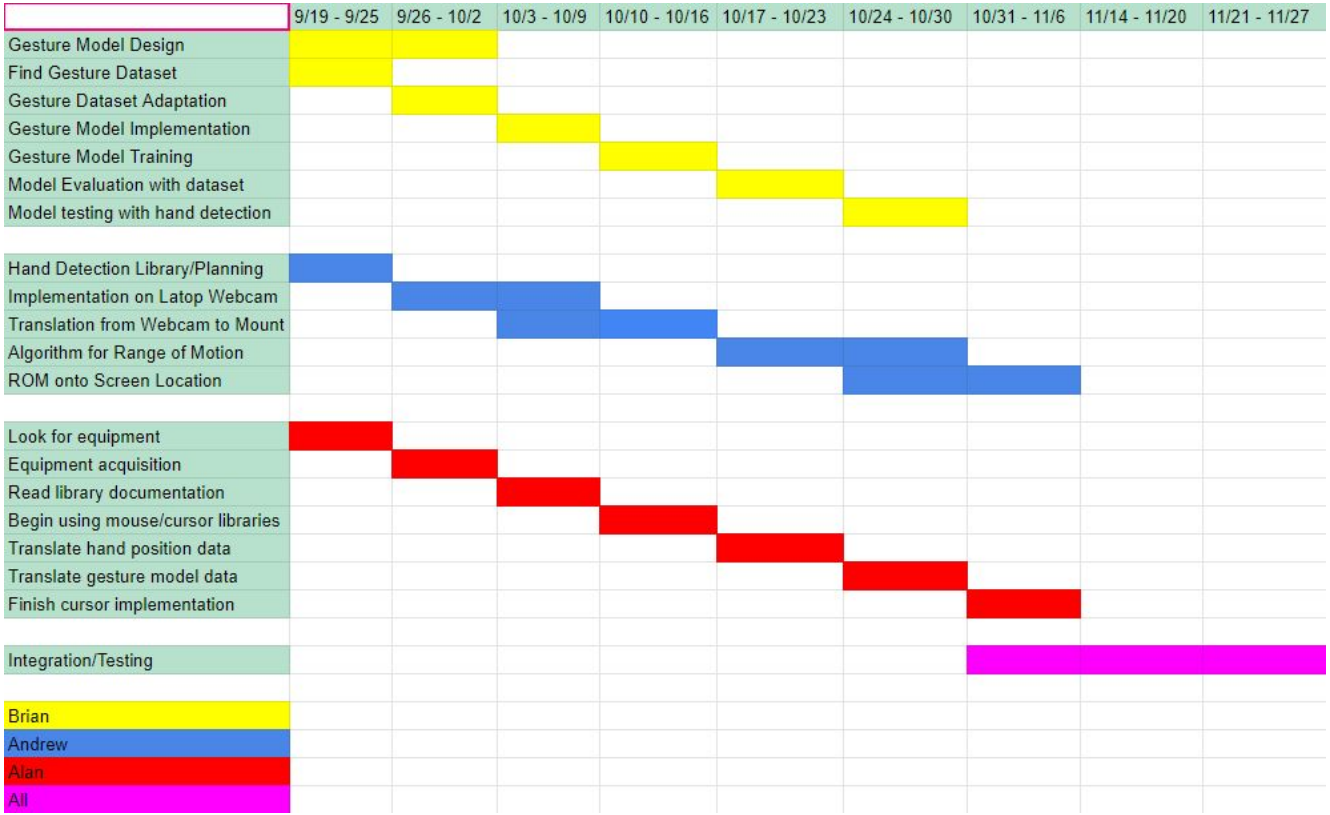
- OpenCV and MediaPipe libraries for detection
- Range of motion detection for operator and translation onto on screen coordinates
- Readjustment and calculation of range of motion bounding box given operator movement
- Filtering of important image data given operator

Brian

Gesture Detection

- Pytorch for model implementation.
- Design model architecture.
- Find and adapt existing gesture datasets
- Develop and train model.
- Implement use of hand location data to select image portions for gesture inference.

Schedule


9/19 - 9/25
9/26 - 10/2
10/3 - 10/9
10/10 - 10/16
10/17 - 10/23
10/24 - 10/30
10/31 - 11/6
11/14 - 11/20
11/21 - 11/27
Gesture Model Design
Find Gesture Dataset
Gesture Dataset Adaptation
Gesture Model Implementation
Gesture Model Training
Model Evaluation with dataset
Model testing with hand detection
Hand Detection Library/Planning
Implementation on Latop Webcam
Translation from Webcam to Mount
Algorithm for Range of Motion
ROM onto Screen Location
Look for equipment
Equipment acquisition
Read library documentation
Begin using mouse/cursor libraries
Translate hand position data
Translate gesture model data
Finish cursor implementation
Integration/Testing
Brian
Andrew
Alan
All