C8: Whiteboard Pal

Sebastien La Duca, Zacchaeus Williams, **Jenny Yu**

Application Area

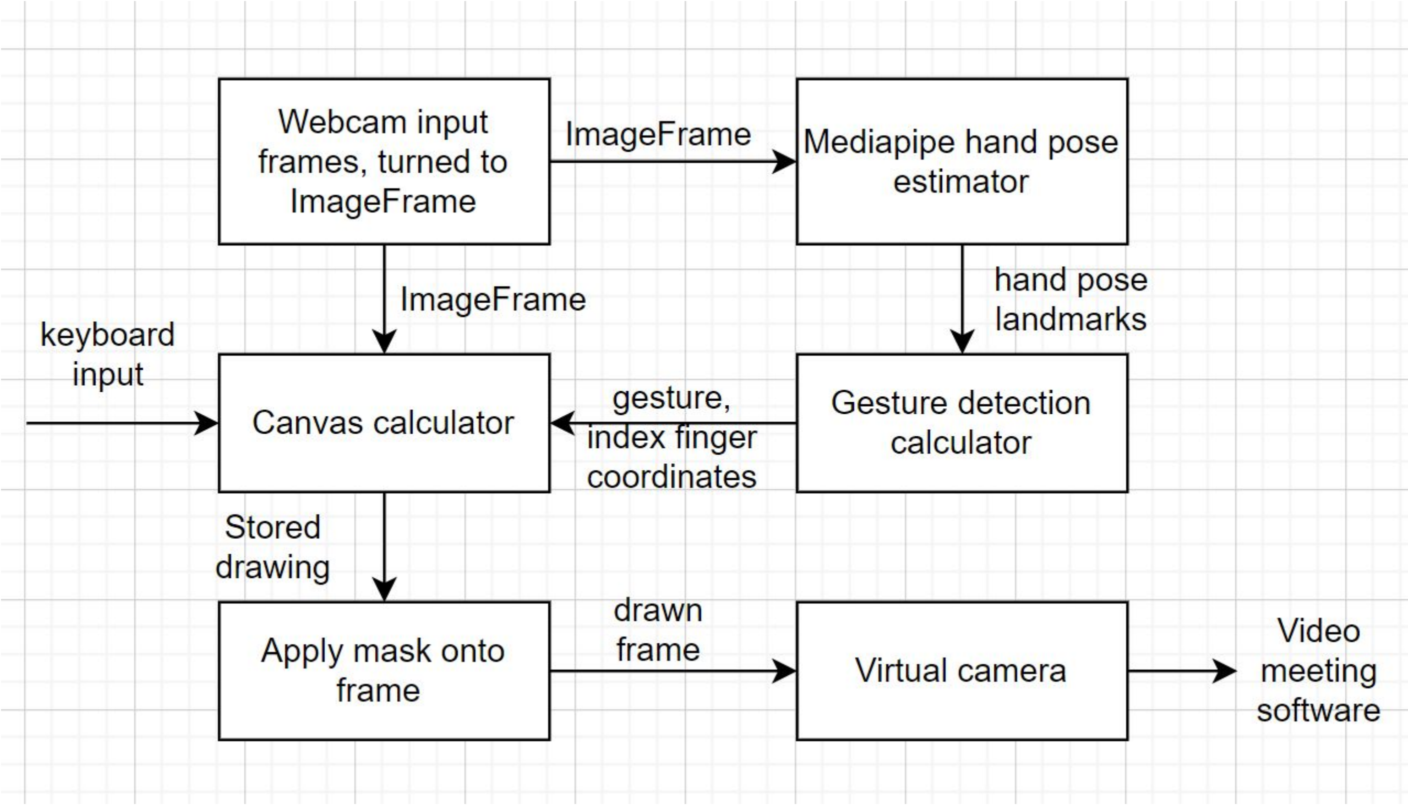
- We solved the problem of teaching remotely without having the hardware to draw diagrams easily.
    - Diagrams aid in explaining, but it is never straightforward setting them up on all systems.
- Tablets and other drawing devices can be expensive and are not consistently integrable for every situation
- Drawing with a mouse is a pain.

Solution Approach

Solution approach changed a lot...

	Before	After
Different	● Separate finger tracking & gesture detection models ● Custom C++ pipeline ● Custom trained gesture detection ● Run inference on GPU with CUDA	● Use Mediapipe for pipeline ● Use Google’s pre-trained hand pose estimation model ● Use hand pose as input to gesture detection + point mapping ● Run everything on CPU ● Add keyboard-based UI
Same	● Use v4l2-loopback kernel module for output ● Use opencv for input and image processing	

Complete Solution - design flow chart



Webcam input frames, turned to ImageFrame

ImageFrame

Mediapipe hand pose estimator

ImageFrame

hand pose landmarks

keyboard input

Canvas calculator

gesture, index finger coordinates

Gesture detection calculator

Stored drawing

Apply mask onto frame

drawn frame

Virtual camera

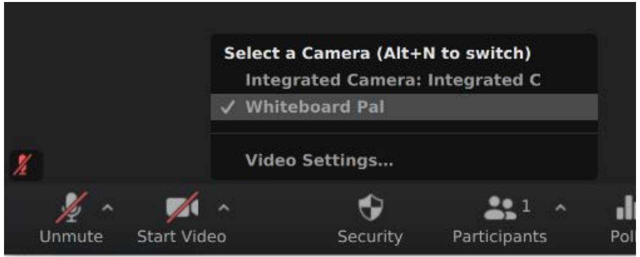
Video meeting software

Complete Solution - specs

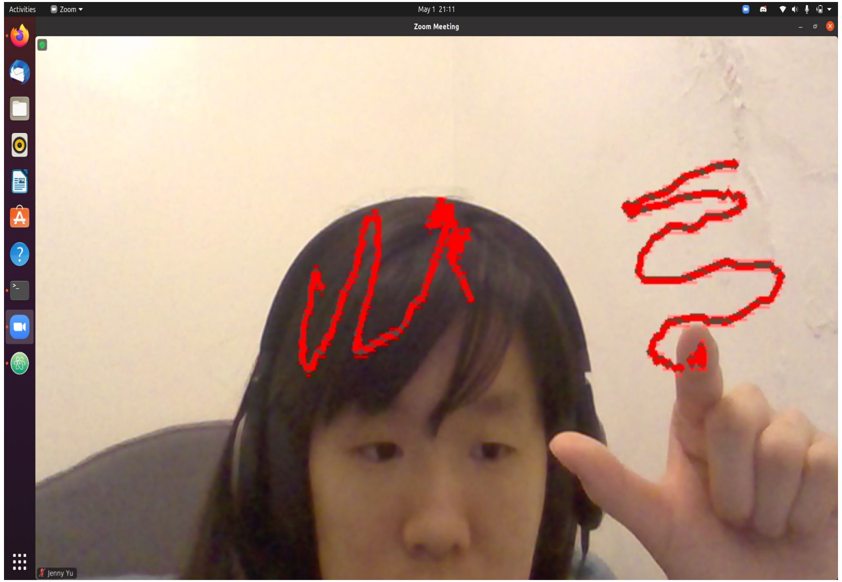
- clone c++ repo at: https://github.com/Sladuca/whiteboard-pal
- machine requirements:
  - Linux (Ubuntu, Mint, Arch, etc, etc)
  - Laptop needs to have a working webcam
  - no GPU needed!
- setup needed:
  - opencv (script to setup included)
  - bazel (we recommend bazelisk)
  - boost
  - ncurses
  - python venv for build
- build & run!
  - initial build can take ~15-20 mins

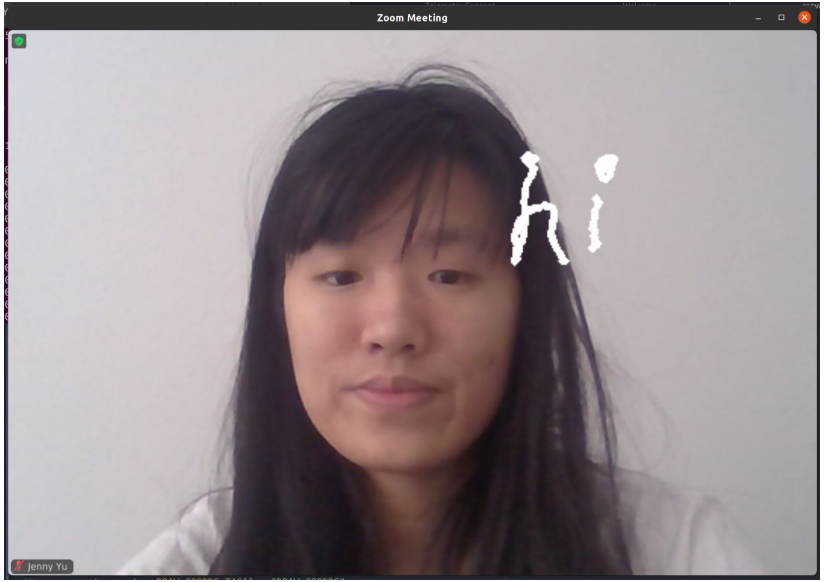
Complete Solution – features

- Tracks index finger tip and draws on webcam feed if presence of drawing gesture detected ->
- 3 modes -> toggle with “m”:
  - free drawing mode (default)
  - line mode
  - eraser mode
- change color of drawing -> toggle with “v”
- “Whiteboard Pal” will show up as camera option on video meeting platform (Zoom, FB Messenger, Google, etc) and can be selected



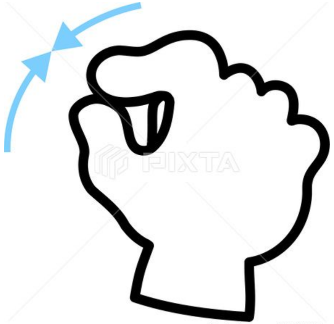
Complete Solutions - screenshots





Testing Plans

- Record instants at which every frame arrives at and leaves a particular pipeline stage
- Use sliding window for calculating averages
- Qualitative testing to determine best gesture out of candidates:
    - L-hand gesture
    - pinch
    - two finger point
    - rocker hand gesture



Testing Results

Performance

- average latency (30 frames, not including camera I/O) ~36-60 ms
  - Sebastien’s CPU (Arch, AMD Ryzen 9 3900X): ~33-43 ms
  - Jenny’s CPU (Ubuntu, intel core i7): ~50-58 ms
  - Zaccahaeus' CPU (Mint, AMD Ryzen 5 3600): ~42-51 ms
- consistently ~30 FPS

Qualitative

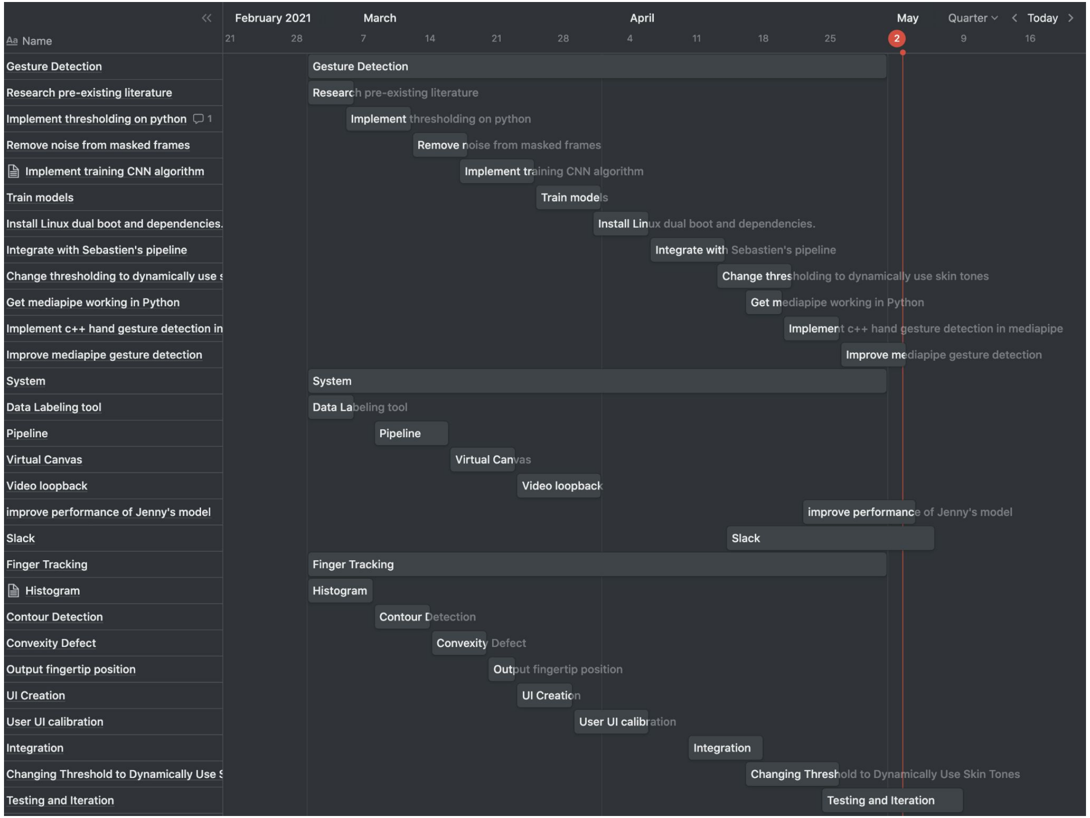
- L hand gesture was best in terms of ease to use and accuracy in detection

Design Tradeoffs

- latency >>> FPS
    - Reaching 30 FPS important but also easy - beyond that, don’t care about FPS
- Development time > UX
    - Getting everything together was harder than expected, had limited time to spend on increasing usability

Project Management

- Gantt chart jettisoned rather quickly
- Engineering tasks that require “figuring stuff out” very hard to schedule



Aa Name

Gesture Detection
Research pre-existing literature
Implement thresholding on python 1
Remove noise from masked frames
Implement training CNN algorithm
Train models
Install Linux dual boot and dependencies.
Integrate with Sebastien's pipeline
Change thresholding to dynamically use s
Get mediapipe working in Python
Implement c++ hand gesture detection in
Improve mediapipe gesture detection
System
Data Labeling tool
Pipeline
Virtual Canvas
Video loopback
improve performance of Jenny's model
Slack
Finger Tracking
Histogram
Contour Detection
Convexity Defect
Output fingertip position
UI Creation
User UI calibration
Integration
Changing Threshold to Dynamically Use S
Testing and Iteration

Conclusions: things we learned along the way

- Gantt Charts
- Application programming where you do not make every single component is incredibly tough
- Documentation for C++ MediaPipe was beyond limited
    - “The source code is the documentation”
- Latency was not as much of an issue as we assumed
- Q&A!