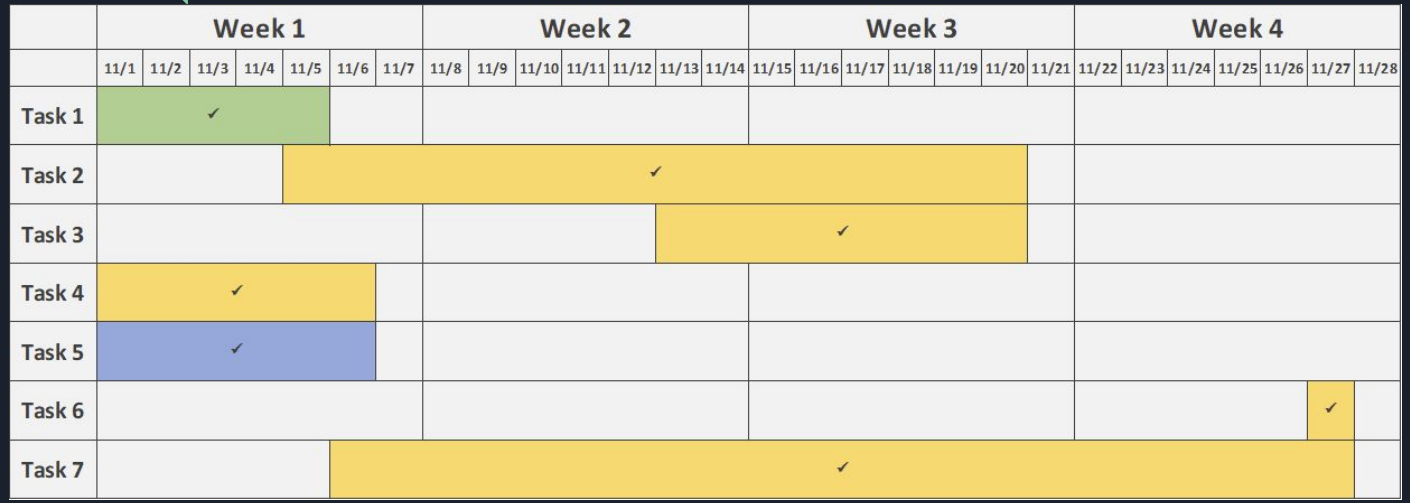
Project 3: Face Detection, Tracking, and Recognition Final Results

Group 6: Eric Watson & Hansi Zheng

Schedule & Roles



Roles

Blue - Hansi Green - Eric Yellow - Both

Tasks

1. Train YOLOv5 Model on Face
2. Implement a system for detection, tracking, recognition
3. Design live demo for Zoom
4. Determine camera FOV and angular resolution
5. Determine max distance from camera for system operation
6. Record system resource usage
7. Work on final presentation

Camera Details (Logitech C920/X)

- Max resolution of 1080p @ 30 FPS
- Lens focal length = 3.67 mm
- Lens diameter = 2 mm
- Angular resolution (Δθ)
  - White (880 nm) = 536.8 µrad
  - Red (660 nm) = 402.6 µrad
  - Green (520 nm) = 317.2 µrad
  - Blue (470 nm) = 286.7 µrad
- Horizontal FOV = 70.42°
- Vertical FOV = 43.3°

Logitech C920

Face Model

- For face detection, the YOLOv5 library [1] is used, with the V5S model being chosen as the face detector.
- For face tracking, the DeepSort library [2] is used with the YOLOv5 library.
  - DeepSort utilizes the SORT algorithm along with a CNN model for feature extraction, which is used to help reduce identity switching between bounding boxes being tracked.
- For face recognition, the Face Recognition library [3] was initially chosen.
  - The face recognition library impacts inference time when encoding detected faces and takes approximately 2 seconds to encode a detected face.
  - Usage of a second stage for face recognition was removed, and the face detector was trained with three classes to perform face recognition (‘Unknown Face’, ‘Eric’, ‘Hansi’) using a custom dataset.

Training Parameters & Face Detection Dataset

- The YOLO V5S model was trained with the following parameters:
  - Epochs: 100
  - Batch Size: 16
  - Image Size: 640
  - Initial Learning Rate: 0.0032
  - Momentum: 0.843
  - Weight Decay: 0.00036
  - IoU Threshold: 0.2
- The Open Images Dataset [4] was used train the YOLO V5S model.
  - 40,000 samples of the ‘Human Face’ class were used.
  - 30% of the samples were used as the validation set.
  - Annotations were converted to format used in YOLOv5.

Face Recognition Dataset

- Video Resolution = 1280 x 720 @ 30 FPS
- Use trained YOLO V5L face detection model to obtain labels from the recorded videos on a local desktop with an Nvidia RTX 2070 Super.
- Manually remove false labels.
- Use ‘Human face’ from Open Image Dataset [4] as ‘Unknown Face’.

	Training Labels	Training Images	Validation Labels	Validation Images
All Classes	26,882	20,615	6,653	5,154
Unknown Face	9,057	2,880	2,189	720
Eric	9,101	9,091	2,277	2,273
Hansi	8,724	8,644	2,187	2,161

Face Recognition Dataset Validation Results

	Precision	Recall	mAP(IoU = 0.5)
All Classes	0.964	0.94	0.96
Unknown Face	0.901	0.823	0.89
Eric	0.998	0.998	0.995
Hansi	0.993	0.998	0.995

Face Recognition & Tracking Results

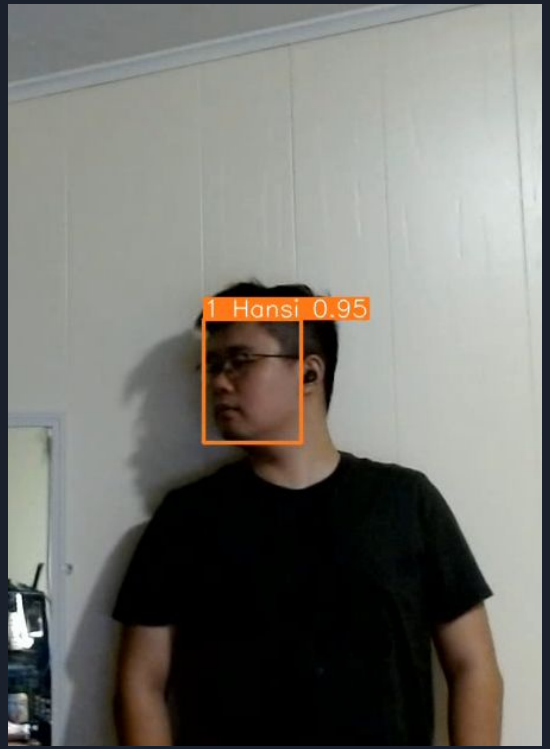
- Video Resolution = 1280 x 720 @ 30 FPS
- Detection/Tracking Resolution = 640 x 480
- Detection/Tracking Average Framerate = (11 ~ 14) FPS

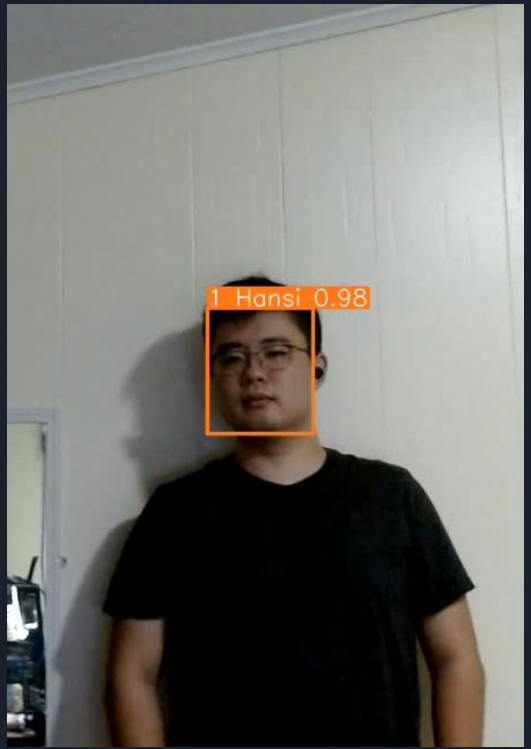
Confidence Based on Distance

	Close (2 FT)	Medium (6 FT)	Far (9 FT)
Front (Hansi)	0.92 ~ 0.98	0.95 ~ 0.97	0.85 ~ 0.96
Front (Eric)	0.51 ~ 0.96	0.87 ~ 0.95	0.88 ~ 0.97
Side (Hansi)	0.80 ~ 0.96	0.90 ~ 0.97	0.81 ~ 0.97
Side (Eric)	0.57 ~ 0.95	0.75 ~ 0.95	0.71 ~ 0.94

Face Detection & Tracking Results (Hansi)

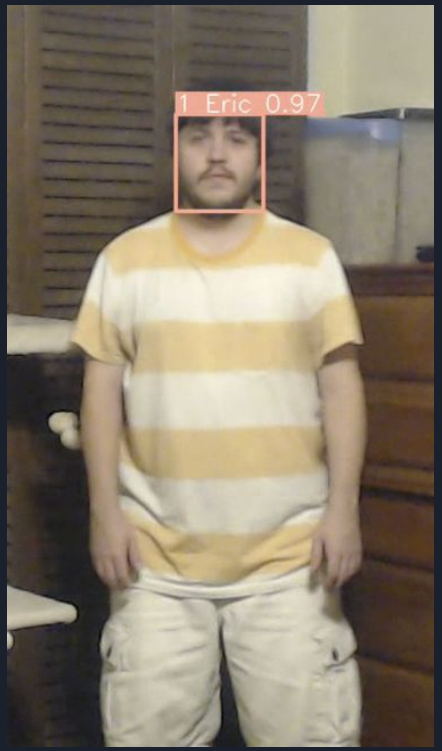
Bounding Box (Front Face) (~9 FT) ~= (61, 72) Pixels





Face Detection & Tracking Results (Eric)

Bounding Box (Front Face) (~9 FT) ~= (48, 53) Pixels



Resource Utilization Results

- The Jetson Stats library [5] was used to monitor resource utilization of the Xavier embedded system.
- A Python script was used to record the data when the system was running the face detector/tracker/recognition system.

	CPU Usage (%)	GPU Usage (%)	RAM Usage (MB)	CPU Temp. (° C)	GPU Temp. (° C)	Power (mW)
Mean	33.92	36.84	7417.5	44.88	43.78	8842.2
Max	53.16	86	7666.0	48	46.5	9937

Conclusion

- We were able to achieve a framerate above 10 FPS for the face recognition system.
- The face recognition system can simultaneously detect/track multiple faces from both the front and side.
- We designed a setup to run the demo for the presentation in the class using a capture card, which would reduce impact on resource utilization.
- Issues faced include having the detected face being the wrong class, but this is most likely due to the lighting conditions from the camera exposure and the projective distortion of the face shown from another device.

Demo Setup

- We use a capture card to capture video output from Xavier to desktop.
- Share desktop screen in Zoom.
- Show Eric’s face on iPad via Discord.

References

1) YOLOv5 Library
   https://github.com/ultralytics/yolov5
2) DeepSort PyTorch
   https://github.com/ZQPei/deep_sort_pytorch
3) Face Recognition
   https://github.com/ageitgey/face_recognition
4) Open Images Dataset
   https://storage.googleapis.com/openimages/web/index.html
5) Jetson Stats
   https://github.com/rbonghi/jetson_stats