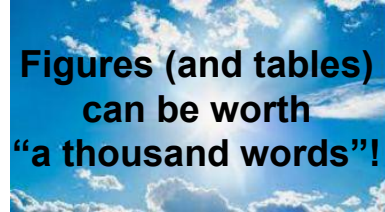


Team A5 – Follow Me

Jeffery Cao, George Chang, Ging Luo

Add your 12 slides after this slide... [remember, 12 min talk + 3 min Q/A]



**Figures (and tables)
can be worth
“a thousand words”!**

See <https://gsuite.google.com/learning-center/products/slides/get-started/> for how to import slides

Make sure to cover: (refer to the Final Presentation Guidance):

- Use Case / Application and Primary (Quantitative) Requirements (i.e. a reminder from prior presentations)
- Solution Approach – a reminder (include updates from Design Review presentation if changed)
 - E.g. block diagram(s), flow chart(s), schematic(s)
- System Implementation – your complete solution
 - E.g., pictures, screenshots, video (make sure that there is CMU access to play any media)
- Testing, Verification and Validation – with quantitative metrics and target values to compare with experiment
 - What tests did you run ? How many tests ? What were the results ?
 - Graphs, tables, quantitative results (compare with the metric targets & ultimately use-case requirements)
- Project Management – tasks, division of labor, and schedule
- Lessons Learned

Consider that this slide already works as a introduction slide so use your first slide wisely (i.e. feel free to delete guidance text)

Use case requirement

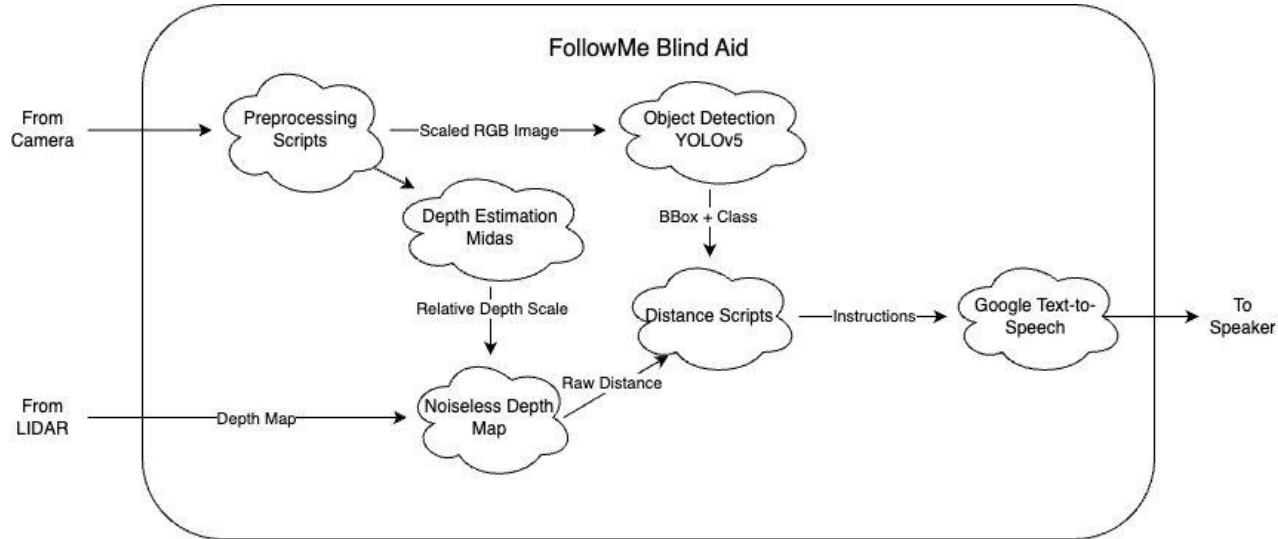
A blind person would benefit from a product which can give them more detail of the surroundings, thus boost their confidence, according to a paper in the British Journal of Visual Impairment.

- The product shall identify the closest obstacle accurately
- The product shall notify the user of obstacles in real-time
- The product shall have enough battery
- The product shall be light enough
- The product shall be economical

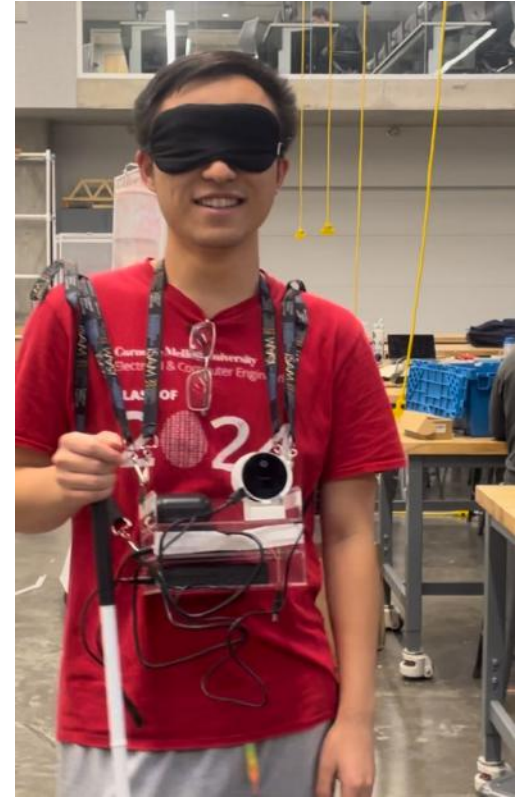
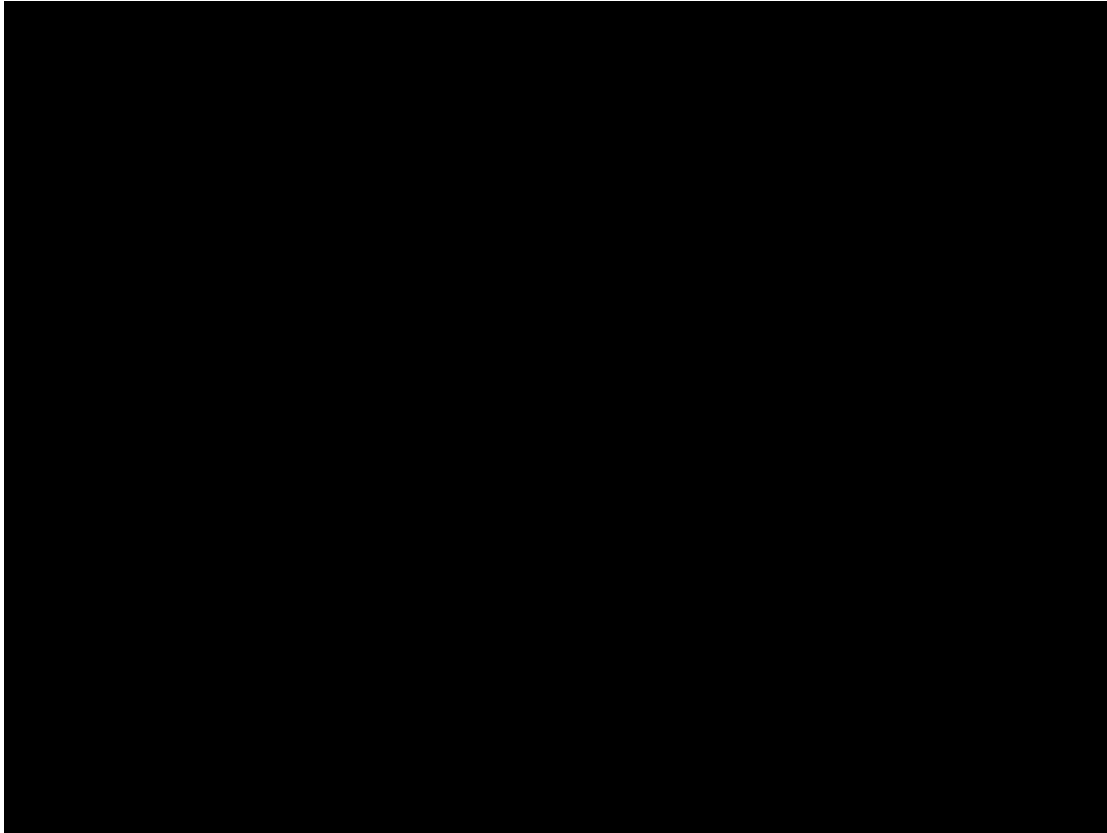


Solution Approach

A portable auditory warning system that uses camera and LIDAR to detect obstacle in front.

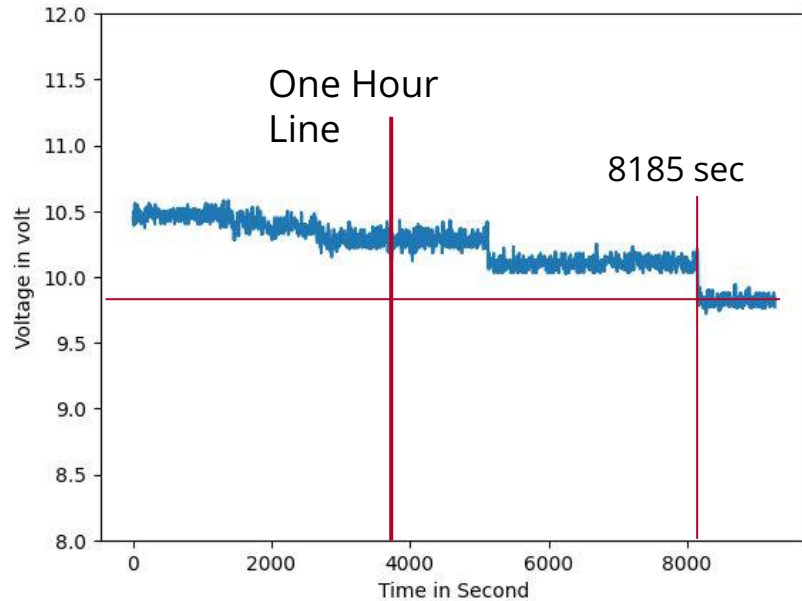


Solution Demo



Hardware Testing: Battery

Design Req: Battery should be above critical voltage of 9.8V for at least one hour



Procedure: Running FollowMe, connected battery watch through USB

Outcome: Battery dropped below 9.8 around 8200second (2.5hr)

Battery Watch: Tradeoff Analysis

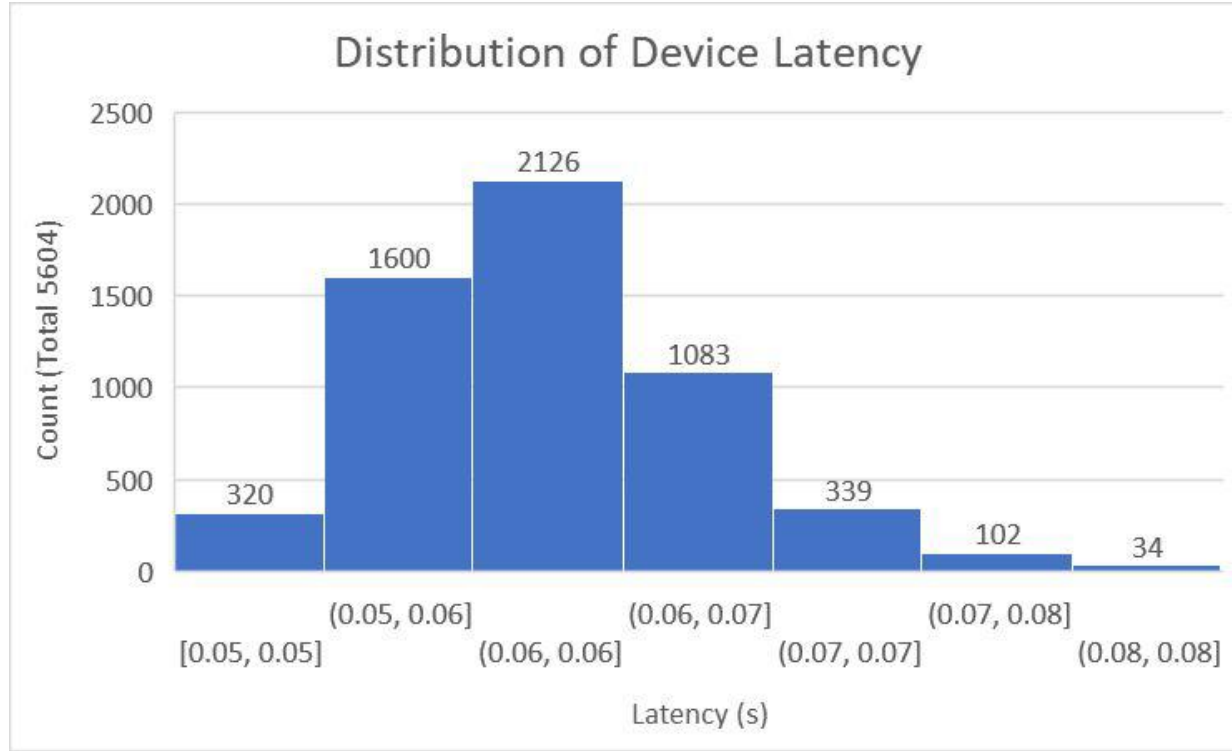
Best Better Good	Arduino	Voltage Comparator Circuit	Dedicated i2c maker's Battery Gauge
Ease of Development	Code Arduino Plug Pins	Built the analog circuit, solder it	Plug and Use
Ease of Data Collection	Built-in UART Conn.	External Voltmeter record by hand	Built-in I2C Connection
Cost and Availability	4\$	Free from 220 Lab	8\$
Support of 12V, 2.5A	Uses Voltage Divider	Op-Amp support high voltage	Only <200mA hard cap

Hardware Testing: Latency

Design Requirements: Maximum latency from camera intake to end of neural model is less than 500ms

Using Python time library to record latency while operating

Maximum Latency	84.2ms
Average Latency	62ms



Software Testing: Accuracy

Metric: Precision curve, Recall curve

Results:

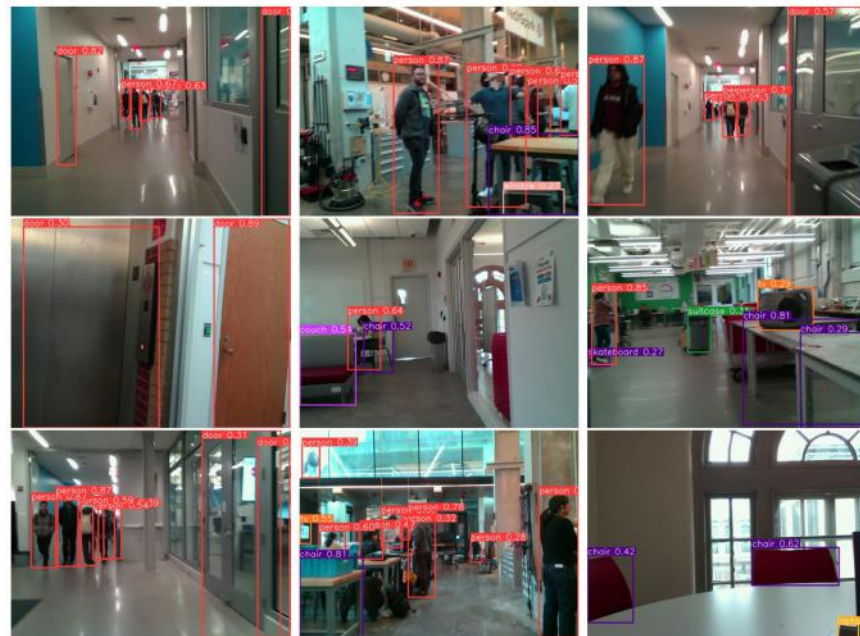
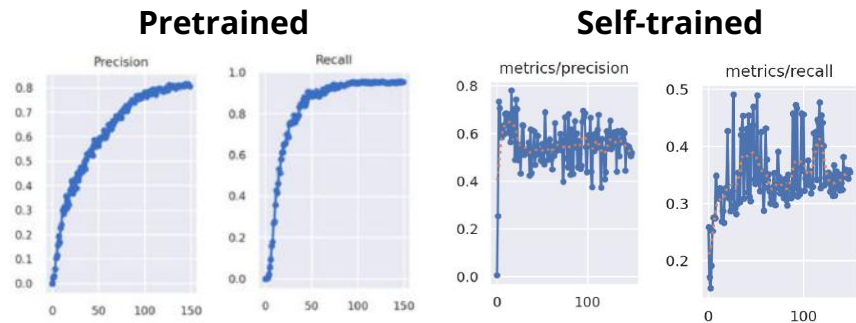
1. Model precision is 0.76, recall is 0.7
2. Real-world Testing in Techspark: Combined Accuracy 0.85

Visualization:

Good: Person

Medium: Door, Chair, Desk, Table

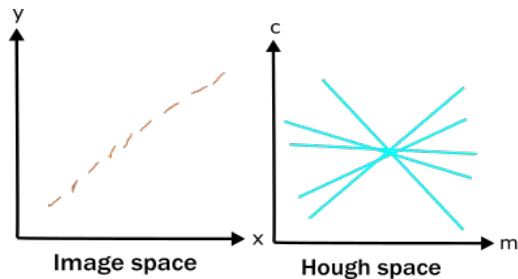
Bad: Window



Tradeoff Analysis: Recognize Objects

	Inference Speed	Accuracy
Hough Transform	CPU Memory Intensive	Bad for complex objects
Segmentation	~10s	Too Detailed
FRCNN	500ms	0.9
YOLO	200ms	0.76

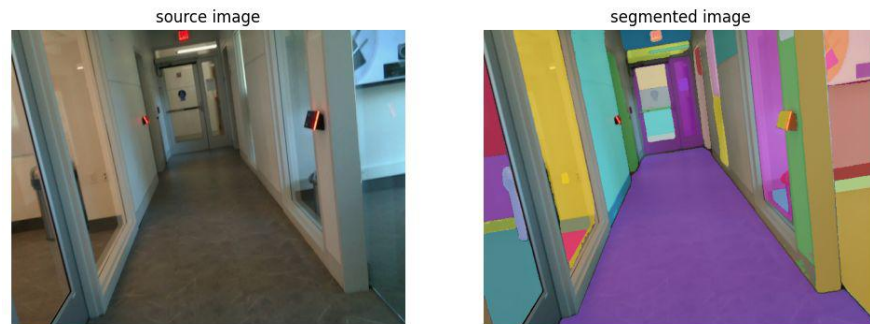
Non-DL approach: Hough Transform



Transform image space to Hough parameter space

Semantic Segmentation: Pixel-by-pixel object detection

Eg: Meta's SAM (Segment Anything Model)



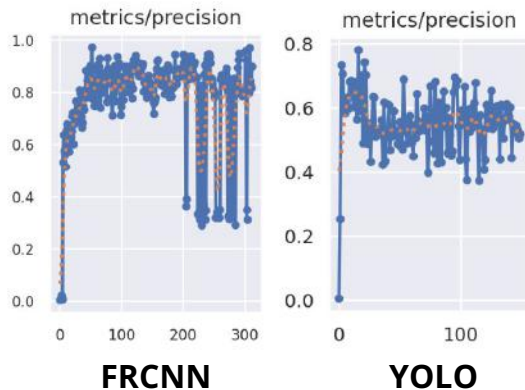
Tradeoff Analysis: Object Detection Architecture

FRCNN: Two-stage detector:

Propose Region of Interest; For each region, recognize.

YOLO: Single-stage detector (The one we choose)

Recognize extracted features directly. (right).



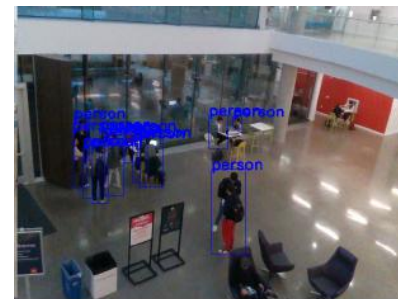
FRCNN

YOLO

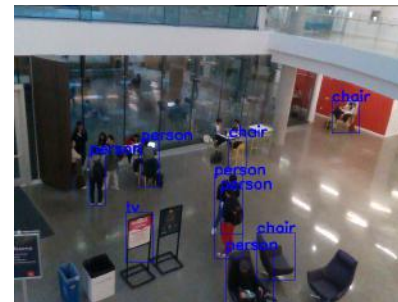
Loss Function:

Cross-entropy loss vs Focal loss

Class-imbalanced Dataset (80% vs 20%)



FRCNN



YOLO

Software Testing: Latency

To improve the latency of the system, we have implemented our scripts with the following strategy:

1. Use numpy functions to speed up distance calculation

=> **Depth Calculation Script < 7.2 ms**

2. Use multiprocessing/thread to spreadout speaker time cost.

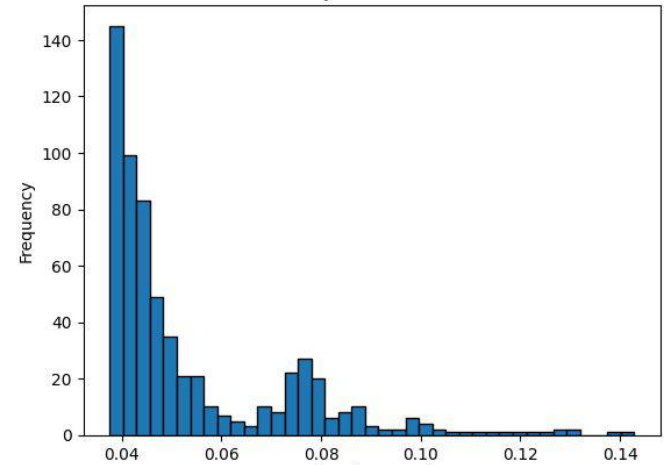
=> **Reach a per picture latency < 500 ms**

3. Use logging to track time spent for each process.

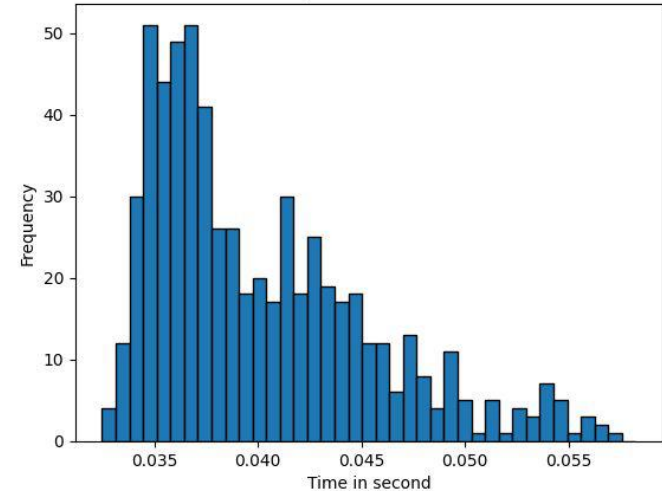
=> **Picture gathering + object label generation < 200ms**

=> **All of above + distance calculation + speaker start announcing < 500ms**

Latency for first model:



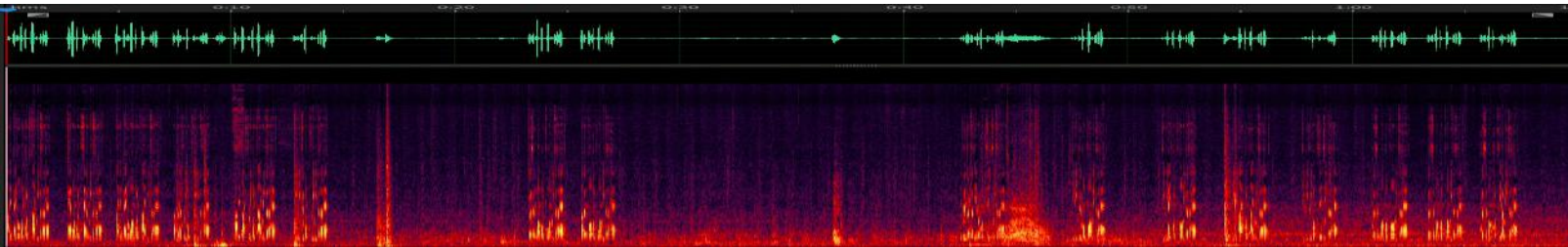
Latency for second model:



Other Testing



	Voice	Weight
Requirement	The voice prompt shall have no overlap	The core component shall weight less than 5 lb
Procedure	Spectral Graph to visually assess abnormal voice behavior; manually decide voice overlap	Put the whole setup on the scale and read its reading.
Verification	No overlap of audio	2lb 13.7oz, less than 5lb
Validation	The product can notify the user clear enough	The prodduct is light enough for user to wear



Schedule Examination

		Nov 5 - Nov 11	Nov 12 - 19	Thanksgiving	Nov 27 - Dec 3	Dec 4 - Dec 10	Exam Week
Jeffery	3D print the mount and mount the system onto it			Work Finalization			
Jeffery	[Hardware] Assemble Equipment Mounting			Work Finalization			
Ging	Refine object detection model + Explore more models for tradeoff			Work Finalization			
George	[Software] Improve on Text-to-Speech			Work Finalization			
George	[Software] Algorithm Instruction Optimization			Work Finalization			
All	[Software] Implement Emergency Protocol						
All	Test the system in Techspark and Scaife Hall + Final Video + Final Report						