Background and Motivation

- Safety and information of the environment are very important aspects of rescue missions
- Not fully understanding the environment and situation can lead to unnecessary risks and dangers

Examples:

Cave/mine rescue
Explorers trapped or lost

Urban search and rescue
Victims trapped in collapsed buildings
Goal

▪ Provide ability to remotely examine the situation and environment

▪ Reduce possible risks or dangers

▪ Improve efficiency of rescue teams in unknown environments
Our Product

- Night Vision Camera
- 180 Degree Gimbal
- Temperature/Humidity Sensor
- Non-Slip Tracks
- Ultrasonic Sensor
- USB Accelerator
- Shockproof Chassis
- Battery Pack
- Raspberry Pi
- 4WD Expansion Board
Requirements Analysis

- Be able to be remotely controlled via Wi-Fi
- Be able to work in dim lighting conditions with night vision
- Gathered sensor data can be viewed remotely
- Can traverse uneven/sloped ground
- Be able to detect obstacles and navigate accordingly
- Be able to detect and classify objects
Block Diagram

- Ultrasonic sensor
- USB Accelerator
- Camera
- GPIO Pins
- TF_lite
- OpenCV
- Stream.H264
- Flask Web Server
- Raspberry Pi board
- 4WD Expansion Board
- Motor Controller
- Wheel Motors
- Camera Controller
- Camera Servos
- Robot Chassis

Data flows from Ultrasonic sensor to GPIO Pins, from GPIO Pins to Flask Web Server, and from Flask Web Server to 4WD Expansion Board. Control signals are sent from Raspberry Pi board to Motor Controller and Camera Controller, and from Camera Controller to Camera Servos.

Sensor Data Visualization
- Camera Display
- Camera Controller
- Motor Controller

Data also flow from Flask Web Server to Camera Controller and Camera Servos, and from Camera Controller to Camera Servos.

Control Instructions flow from Raspberry Pi board to Motor Controller and Camera Controller.
# Requirements Analysis: Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>6 lb</td>
</tr>
<tr>
<td>Dimensions</td>
<td>256<em>183</em>213 mm</td>
</tr>
<tr>
<td>Battery Life</td>
<td>Board 5.8 hours</td>
</tr>
<tr>
<td>Camera</td>
<td>Night Vision 5MP</td>
</tr>
<tr>
<td>Speed Range</td>
<td>0.7 ~ 6.5 km/h</td>
</tr>
<tr>
<td>Obstacle Detection Range</td>
<td>3 ~ 450 cm</td>
</tr>
<tr>
<td>Video Stream w/ Object Detection Frame Rate</td>
<td>H.264 640x480 @ 30FPS</td>
</tr>
<tr>
<td>Object Detection Range</td>
<td>6 meters (best case scenario)</td>
</tr>
</tbody>
</table>
Battery Life Analysis

- Current peripherals consumes 800 mA in total
- Raspberry Pi 4 requires 3.7V, 3A* to operate stably
- Very few battery banks in market provide 3.7V, 3A output

### Main Board Power Consumption

<table>
<thead>
<tr>
<th>Components</th>
<th>Q’ty</th>
<th>Current (A)</th>
<th>Voltage (V)</th>
<th>Power (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi</td>
<td>1</td>
<td>1.1</td>
<td>3.7</td>
<td>4.1</td>
</tr>
<tr>
<td>Camera</td>
<td>1</td>
<td>0.16</td>
<td>3.7</td>
<td>0.59</td>
</tr>
<tr>
<td>UltraSonic</td>
<td>3</td>
<td>0.015</td>
<td>3.7</td>
<td>0.06</td>
</tr>
<tr>
<td>Camera Motors</td>
<td>2</td>
<td>0.3</td>
<td>3.7</td>
<td>1.1</td>
</tr>
<tr>
<td>USB Accelerator</td>
<td>1</td>
<td>0.5</td>
<td>3.7</td>
<td>1.85</td>
</tr>
<tr>
<td><strong>Sum</strong></td>
<td><strong>8</strong></td>
<td><strong>2.375</strong></td>
<td><strong>3.7</strong></td>
<td><strong>8.9</strong></td>
</tr>
</tbody>
</table>

### Battery Life Analysis

<table>
<thead>
<tr>
<th>Components</th>
<th>Q’ty</th>
<th>Capacity (Ah)</th>
<th>Current (A)</th>
<th>Battery Life (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Battery</td>
<td>1</td>
<td>11.1</td>
<td><strong>1.9</strong></td>
<td><strong>5.84</strong></td>
</tr>
</tbody>
</table>

### Driving Board Power Consumption

<table>
<thead>
<tr>
<th>Components</th>
<th>Q’ty</th>
<th>Current (A)</th>
<th>Voltage (V)</th>
<th>Power (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drive Board</td>
<td>1</td>
<td>0.1</td>
<td>12</td>
<td>1.2</td>
</tr>
<tr>
<td>Wheel Motors</td>
<td>6</td>
<td>0.35</td>
<td>12</td>
<td>12.6</td>
</tr>
<tr>
<td><strong>Sum</strong></td>
<td><strong>7</strong></td>
<td><strong>2.2</strong></td>
<td><strong>12</strong></td>
<td><strong>13.8</strong></td>
</tr>
</tbody>
</table>

* https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/*
CDR Deliverables

- Improve accuracy of object detection
- Improve speed of object detection
- Make semi-autonomous navigation more reliable
- Train model to be able to detect/classify certain objects

Responsibilities

- Derek Sun
  - Construct robot and restore functionality, compile training dataset, integrate USB accelerator, improve object detection, re-implement semi-autonomous navigation
- Arthur Zhu
  - Compile training dataset, improve object detection, data collection and analysis, battery analysis
CDR Deliverables: Robot

- Flask web application running off Raspberry Pi
  - Robot controller
  - Camera controller
  - Night vision video feed w/ object detection
  - Keyboard controls for better UX
  - Mobile-friendly
- Semi-autonomous navigation enabled
CDR Deliverables: Object Detection

- Implemented with Python, Tensorflow + TFLite, and OpenCV

Training
- Transfer learning with SSD MobileNetV2 model as basis
- Compiled our own image database (person, rock)
  - Used labelImg to label images
CDR Deliverables: Object Detection

Evaluation

- Tensorboard visualization tool
  - Provides training/eval metrics
- Detect overfitting/underfitting

<table>
<thead>
<tr>
<th>Detection Model Evaluation Metrics</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td>Value</td>
</tr>
<tr>
<td>mAP</td>
<td>0.4971</td>
</tr>
<tr>
<td>mAP (large)</td>
<td>0.5108</td>
</tr>
<tr>
<td>mAP (medium)</td>
<td>0.06634</td>
</tr>
<tr>
<td>mAP (small)</td>
<td>0.0016068</td>
</tr>
<tr>
<td><a href="mailto:mAP@.50IOU">mAP@.50IOU</a></td>
<td>0.8607</td>
</tr>
<tr>
<td><a href="mailto:mAP@.75IOU">mAP@.75IOU</a></td>
<td>0.5804</td>
</tr>
</tbody>
</table>
Demo
Proposed FPR Deliverables

- Further improve accuracy of object detection
- Improve robustness of robot

Responsibilities
- Derek Sun
  - Improve object detection accuracy, improve training dataset
- Arthur Zhu
  - Robustness enhancement, improve training dataset
# SCHEDULE OF INTELLISAR

<table>
<thead>
<tr>
<th>TASKS</th>
<th>JANUARY</th>
<th>FEBRUARY</th>
<th>MARCH</th>
<th>APRIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robot Reconstruction</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integrate USB Accelerator</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Re-implement Semi-Autonomous Navigation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Improve Semi-autonomous Navigation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add the Ability of Detecting Rock</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Improve accuracy of Object Detection</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Questions?
Appendix: Object Detection Metrics

Precision
- measures how accurate the model’s predictions are
- defined as: \( \text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} = \frac{\text{True Positives}}{\text{Total # of Predicted Positives}} \)

Recall
- measures how well the model finds all the positives
- defined as: \( \text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \)

Ex) in the context of a person detector:
\[
\text{Precision} = \frac{\# \text{People correctly predicted}}{\# \text{People correctly predicted} + \# \text{People incorrectly predicted}}
\]
\[
\text{Recall} = \frac{\# \text{People correctly predicted}}{\# \text{People correctly predicted} + \# \text{Objects incorrectly predicted as person}}
\]
Appendix: Object Detection Metrics

Intersection over Union (IoU)
- measures the overlap between the bounding box generated by the model and the ground truth bounding box and is what determines whether a prediction is a true positive, false positive, or false negative

Average precision (AP)
- defined as the area under the precision-recall curve (PR curve), with the recall on the x-axis and precision on the y-axis.

Mean average precision (mAP)
- calculated by taking the average of the AP for all the classes being predicted
Appendix: TensorBoard Metrics

mAP
- obtained by averaging the mAPs calculated using IoU thresholds ranging from .5 to .95 with increments of .05

mAP (large)
- calculated mAP for large objects
  (96^2 pixels < area < 10000^2 pixels)

mAP (medium)
- calculated mAP for medium-sized objects
  (32^2 pixels < area < 96^2 pixels)

mAP (small)
- calculated mAP for small objects
  (area < 32^2 pixels)
Appendix: TensorBoard Metrics

mAP@.50IOU
- mAP calculated using a IoU threshold of 50%

mAP@.75IOU
- mAP calculated using a IoU threshold of 75%