

Use Case/Solution Approach

Goals:

- Make search and rescue autonomous and distributed
 - No human input
 - Effective coordination between various search devices
- Deliver supplies to someone in need
- Should work in variety of environments, from open space to space with obstacles

Solution Approach:

- A SWARM with small rovers that segment and search the space for a target
- This has changed since our proposal presentation
 - Drones are much more expensive, leading to smaller SWARMS under the same budget
 - Purchases related to drones and drone parts are heavily restricted due global conflicts
 - Rovers can carry more weight with far more power efficiency

Design Requirements

Use Case Requirements	Associated Design Requirements
Fully autonomous with effective distributed algorithms	Raspberry Pi for algorithms and sensor data Index-based arbitration system Space distribution leads to <20% overlap
Successfully finds target 90% of the time	Efficient pathfinding algorithms: A* and exhaustive searching
Search time decreases with number of vehicles	Space separation continues until each rover has its own space Time to target identification in same environment scales with $1/n$
Deliver supplies to target	Can carry at least 1lb Convergence rate: $>\frac{2}{3}$ of all rovers get to target
Find target in 1000sq ft room in 10 minutes with 3 bots	Quick: at least 1 foot per second Good control: Sensors for localization to verify positions

Design Requirements

Use Case Requirements	Associated Design Requirements
Low cost per unit	Capstone budget: each rover must cost less than \$100
Localization	UWB for coordinate system Motor encoders, camera, IR sensor for position data.
Sufficiently long battery life	At least 30 minute battery life
No collisions between rovers or with obstacles	IR sensors
Communications	UWB and distributed communications algorithms

System Specification and Block Diagram

UWB:

- Allows for communication and ToF distance sensing between rovers

Camera:

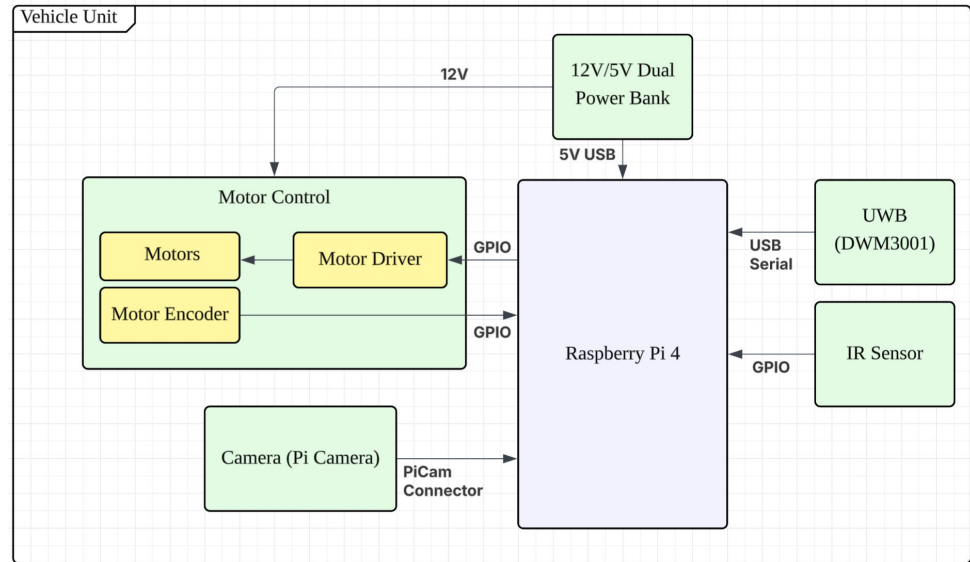
- Target detection
- Obstacle detection
- Movement

Motors:

- Encoders provide movement data

IR Sensor

- Can be used for obstacle detection and movement reinforcement



HW Systems

Localization:

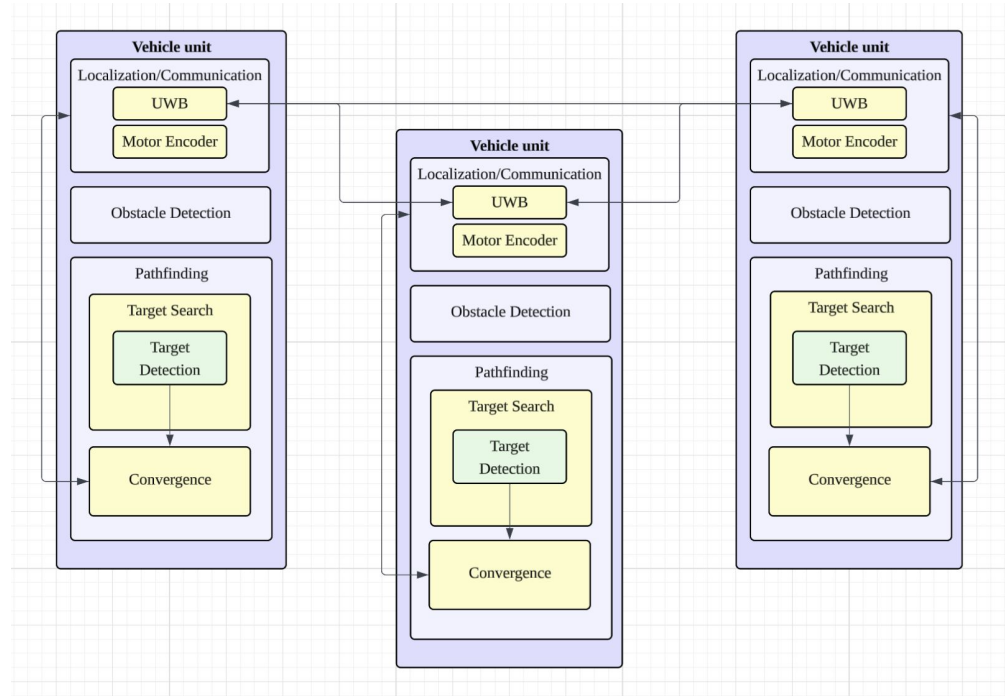
- Done using motor encoders with support from camera and IR
- Transmitted to others over UWB

Obstacle Detection

- Using camera and IR

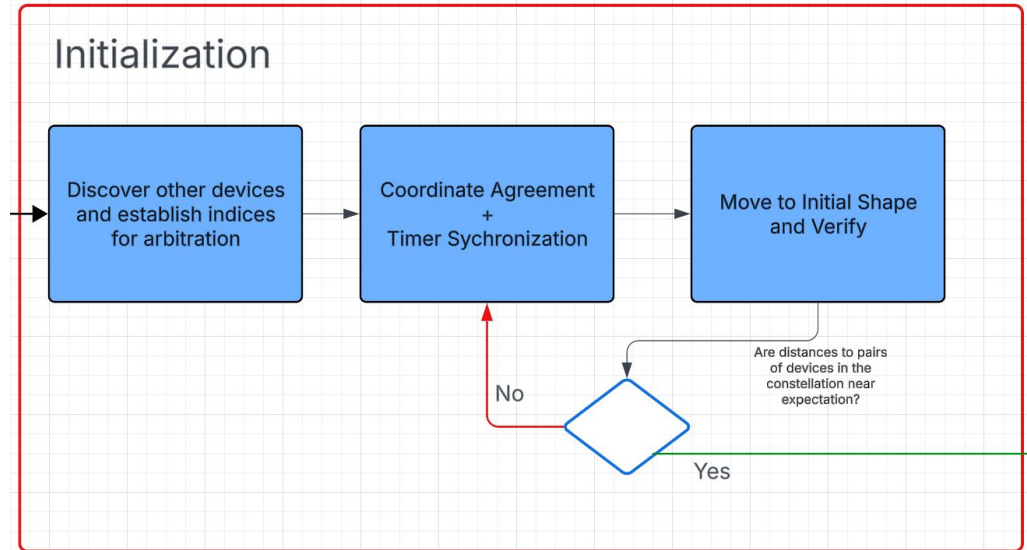
Pathfinding:

- Exhaustively search owned space, pathfinding around obstacles
- Uses camera to detect a target
- Transmit location of target over UWB so others can converge



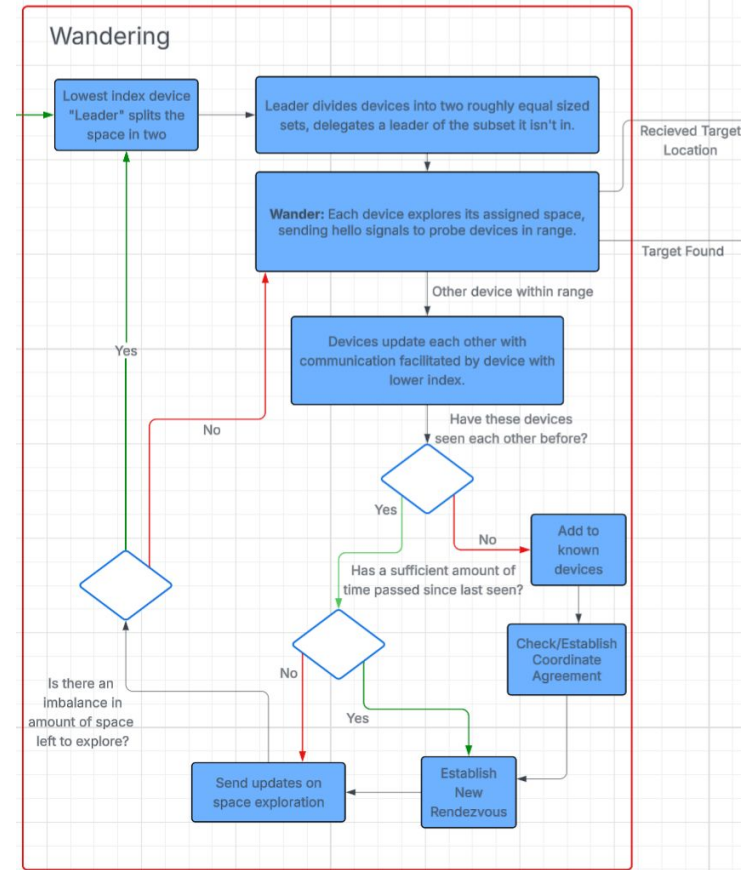
Software Approach: Initialization

- Rovers identify each other and assign unique indices.
- A designated rover leads the others through a coordinate calibration process. This establishes an agreed upon coordinate frame.
- All vehicles then move to an initial formation to organize and verify calibration.



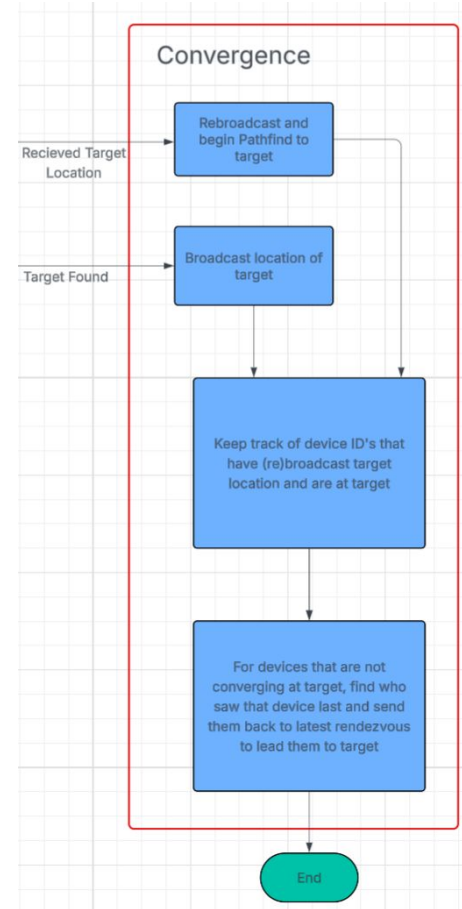
Software Approach: Wandering

- When multiple rovers operate in a space, the lowest-index rover splits the area and assigns rovers to one of two sections.
- Rovers wander their assigned regions, pathfinding toward unexplored areas.
- When encountering each other, rovers check coordinate alignment, record the contact, and update exploration state, re-dividing if necessary.



Software Approach: Convergence

- When a rover finds the target, it broadcasts the location so others can converge.
- If a rover is out of range, it will not receive the broadcast and must be informed another way.
- Using last-seen records, the rover with the most recent contact returns to that location to relay the target position.



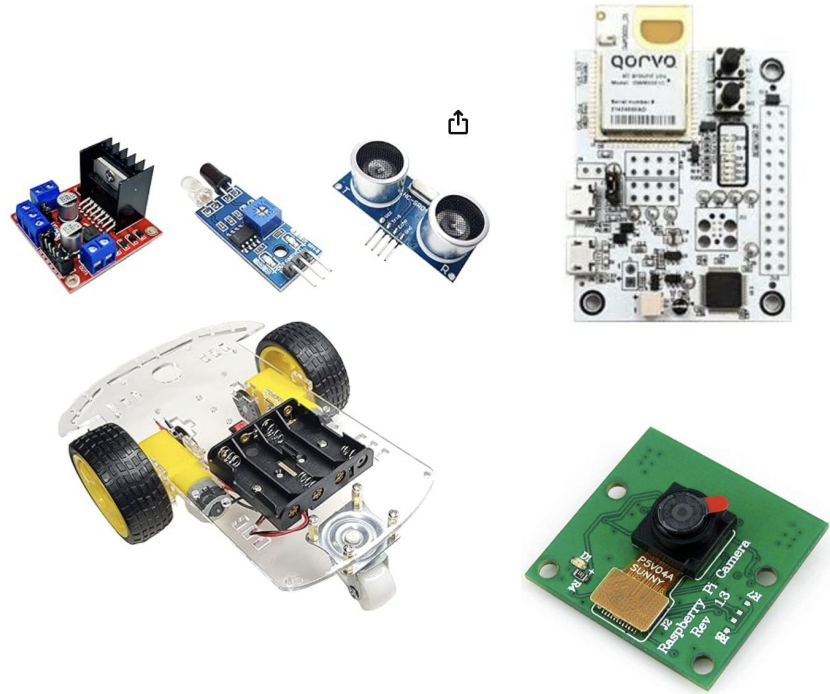
Implementation Plan

Hardware:

- **Raspberry Pi 4** (capstone inventory)
- Raspberry Pi **Camera** (purchased)
- 2-Wheel **Rover Kit** (purchased and enhanced)
 - **Frame, Wheels, Motors**
 - **LN298N Motor Drivers**
 - **IR Sensor**
- **DWM3001CDK UWBs** (purchased)

Software:

- Raspberry Pi OS
- Python libraries for interfacing with GPIO
 - RPi.GPIO
- Github repo for UWB interface
 - DW1000_Raspi_Python_library



Test, Verification and Validation

Design Requirement	Test Method	Quantified Metric
Message Consistency	Send a set of messages from one device to another	>95% reception rate per message, 100% after corrective retransmission
Distributed Message System	One device coordinates a “conversation” between 3+ devices	Messages sent in proper order
Autonomy	Specify target position to pathfind to	<30cm error from target
Convergence	Have rovers converge, measure space between rovers	<30cm per rover
Collision Avoidance	Send rover to a target with obstacles in path	stays >10cm from obstacles
Target Identification	Measure time (Wright room)	<5 minutes

Risk Mitigation

Risk Factors	Mitigations
Insufficient Power	Separate power supply
Motor Encoder Accuracy	Kalman filters for drift mitigation
Data Bandwidth > UWB Bandwidth	Revert to Wi-Fi
UWB inaccuracy	Artificial potential field for separation Marco-Polo/Hotter-Colder for convergence

