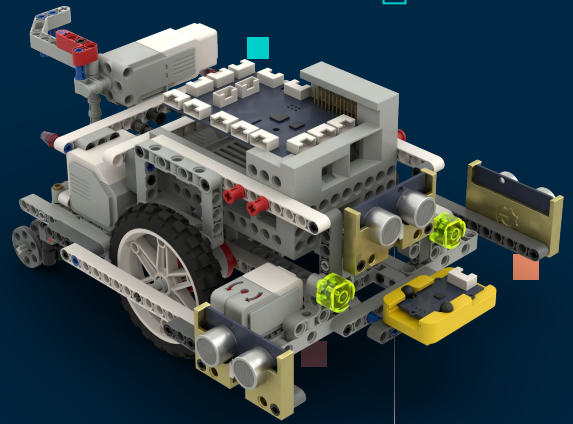
GEARS Design Proposal

Team #45
Sai Karthik
Gabe Kurfman
Alexander Brettnacher
Matthew Roxas

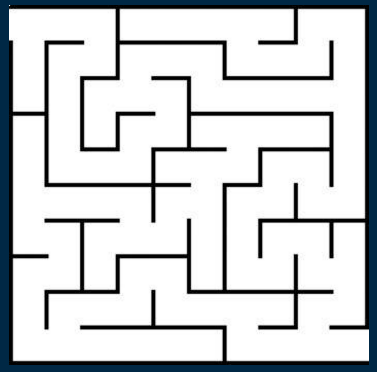


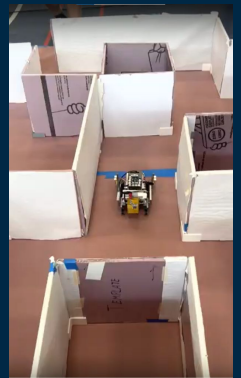
Project Objectives

Our Task: Design and build a Global Emergency Autonomous Response System (GEARS) for disasters zones across the globe

- "Precise navigation through damaged areas”
- “Characterize and avoid hazardous locations in the area”
- “Transmit a map of the path taken through the environment”
- “Communicate with community recipients in a non-hostile manner"

-Project 3 Description Spring 2023





Project Management

Metrics of Success

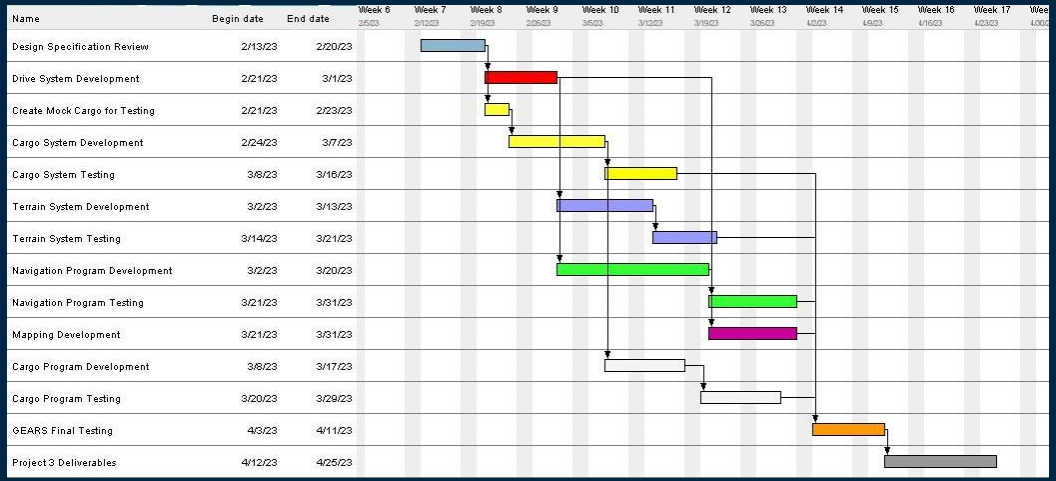
- PoC task execution
- Checklist of listed tasks in project description
- Consistency rates

Decision-Making

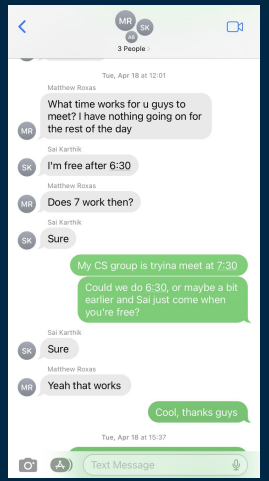
- Decisions were made as a group
- Execution was decided on task-by-task basis

Time Management

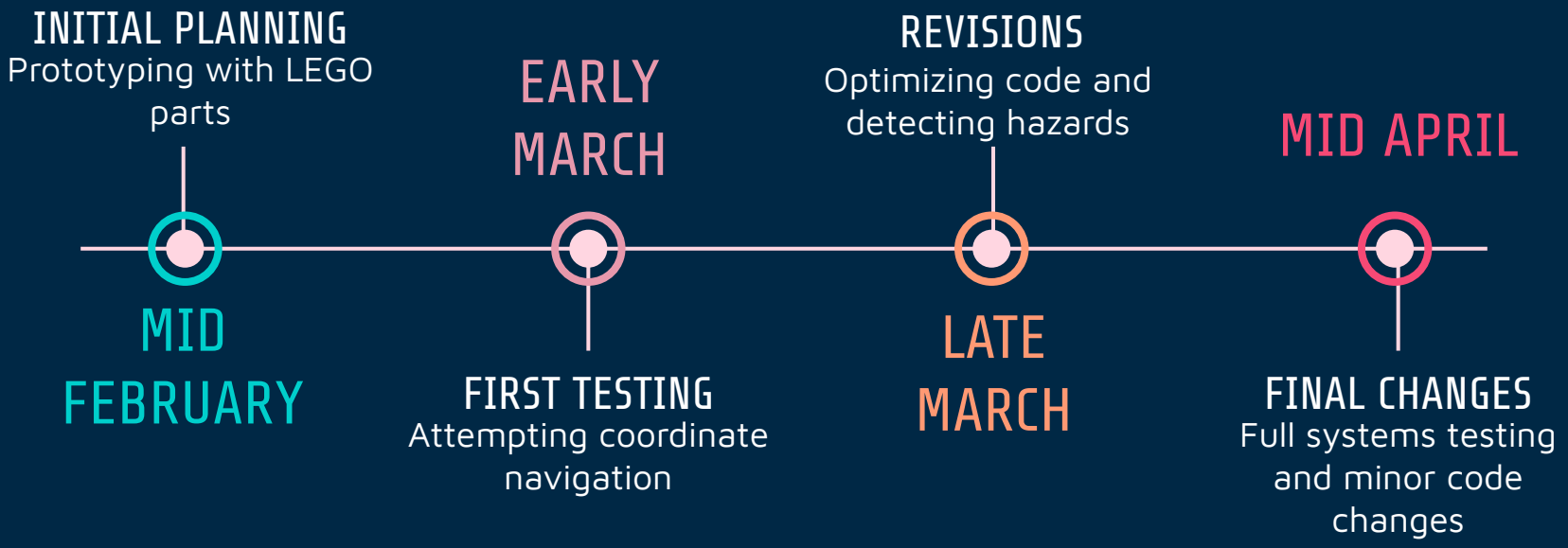
- Met at least once a week, tried to make meaningful decisions within each meeting
- Gantt Chart



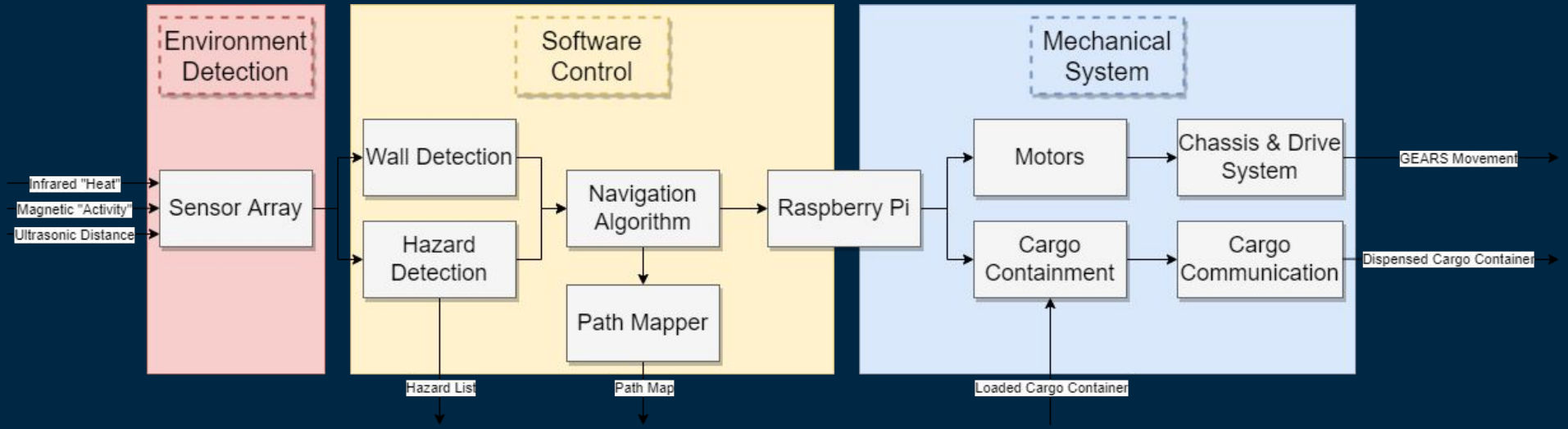
Name	Begin date	End date
Design Specification Review	2/13/23	2/20/23
Drive System Development	2/21/23	3/1/23
Create Mock Cargo for Testing	2/21/23	2/23/23
Cargo System Development	2/24/23	3/7/23
Cargo System Testing	3/8/23	3/16/23
Terrain System Development	3/2/23	3/13/23
Terrain System Testing	3/14/23	3/21/23
Navigation Program Development	3/2/23	3/20/23
Navigation Program Testing	3/21/23	3/31/23
Mapping Development	3/21/23	3/31/23
Cargo Program Development	3/8/23	3/17/23
Cargo Program Testing	3/20/23	3/29/23
GEARS Final Testing	4/3/23	4/11/23
Project 3 Deliverables	4/12/23	4/25/23



Our Process


INITIAL PLANNING
Prototyping with LEGO parts
MID FEBRUARY
EARLY MARCH
FIRST TESTING
Attempting coordinate navigation
REVISIONS
Optimizing code and detecting hazards
LATE MARCH
MID APRIL
FINAL CHANGES
Full systems testing and minor code changes

Functional Block Diagram

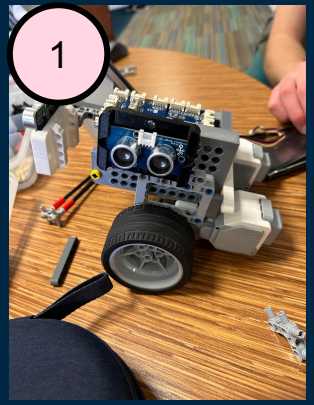


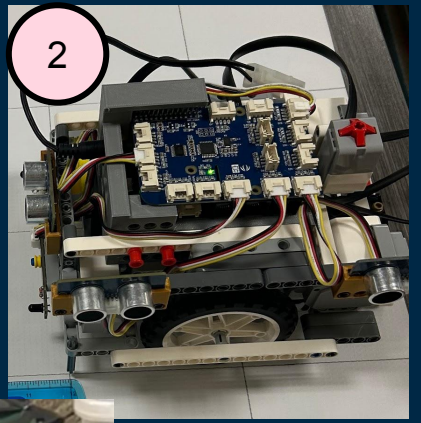
Engineering Specification

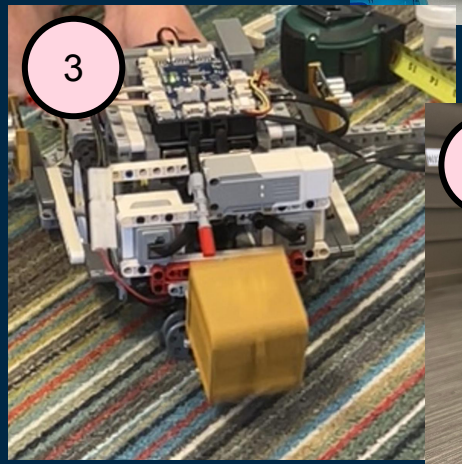
Customer Needs	Technical Needs	Technical Requirement	Target Value
Characterize and avoid hazardous locations	Percent of obstacles identified	>90% identified	100% identified
Precise navigation through damaged areas	Percent of obstacles avoided	>75% avoided	100% avoided
Transport cargo through unknown terrain	Percent of sucessful cargo drops	>75% successful	100% successful
Transmit a map of the path taken	Resolution of map transmitted	Accurate to <0.5 m^2	Accurate to <0.2 m^2
Indicate contents of cargo without specific language	Percent of recipients that recognized cargo	>80% recognized	100% recognized
Communicate with recipients without hostility	Percent of missions resulting in antagonistic actions	<15% antagonistic	0% antagonistic

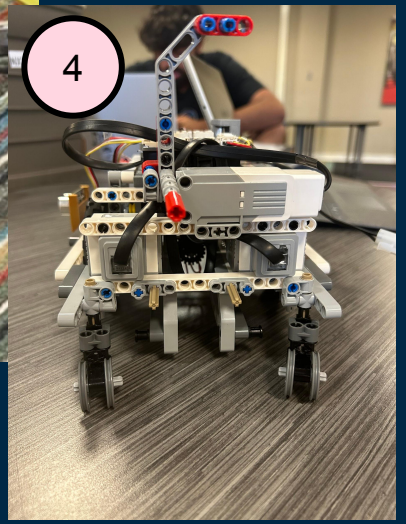
Prototypes

1. Made barebones visualization of GEARS system
2. Created first design with coordinate navigation as a priority
   a. 2 ultrasonic sensors on 1 side
   b. 1 centered caster wheel
   c. Touch sensor added
3. Added a structure to deposit cargo container consistently, and chose to use one ultrasonic sensor on each side
4. Switch rear navigation to include 2 casters rather than 1





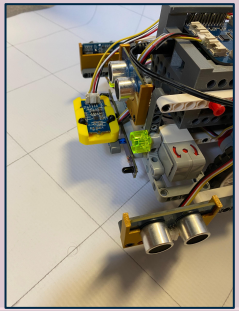




Other Design Features

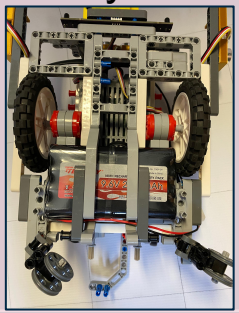
Sensor Array

- Compact
- Easy to modify
- Gyro, IMU, & 3 Ultrasonic



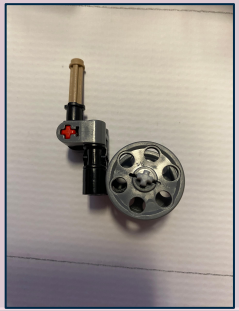
Battery Compartment

- Easy to remove for charging
- Secures battery



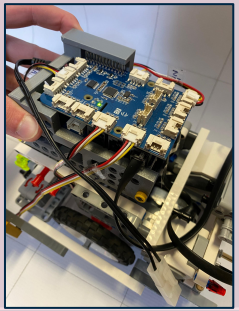
Caster Wheel

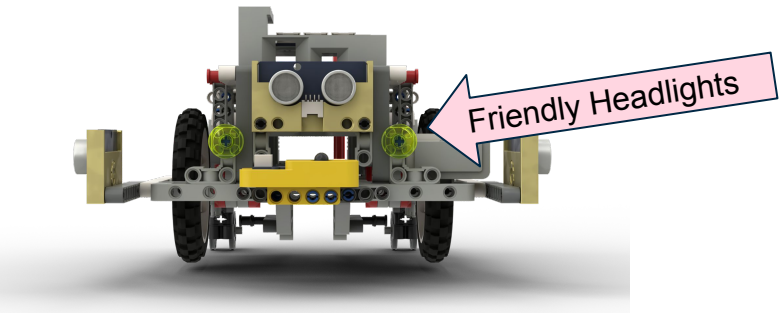
- Easy to spin
- Compact and low friction

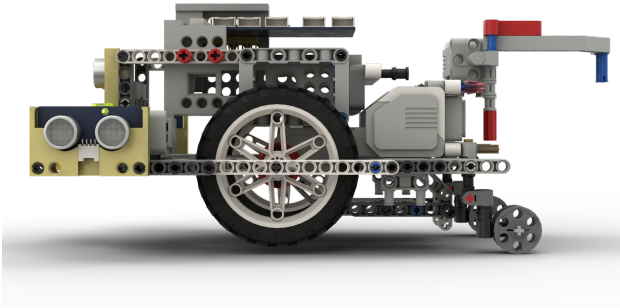


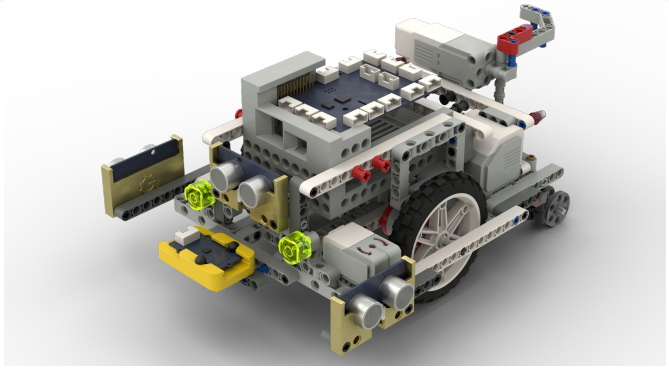
Raspberry Pi Mount

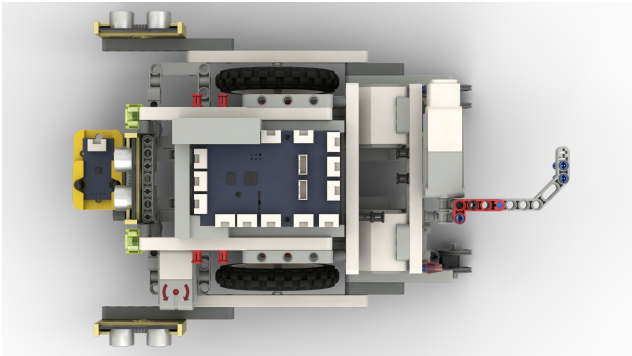
- Secures RPi and GrovePi
- Removable using 4 pins



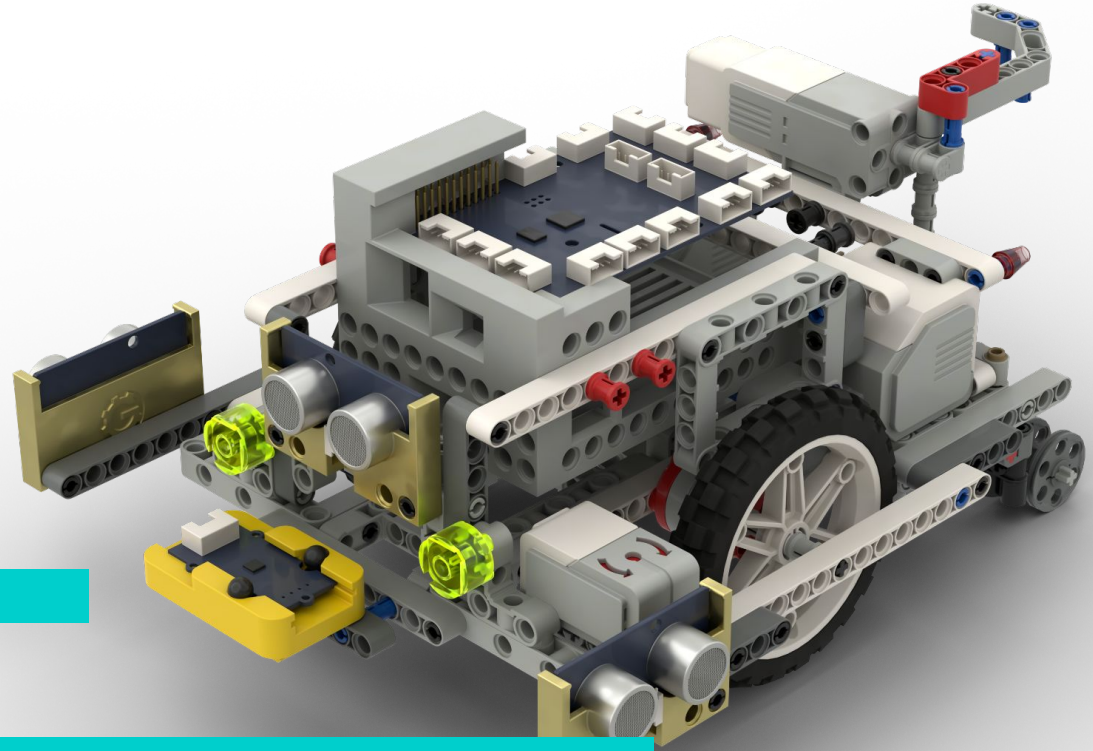








Final Robot Design

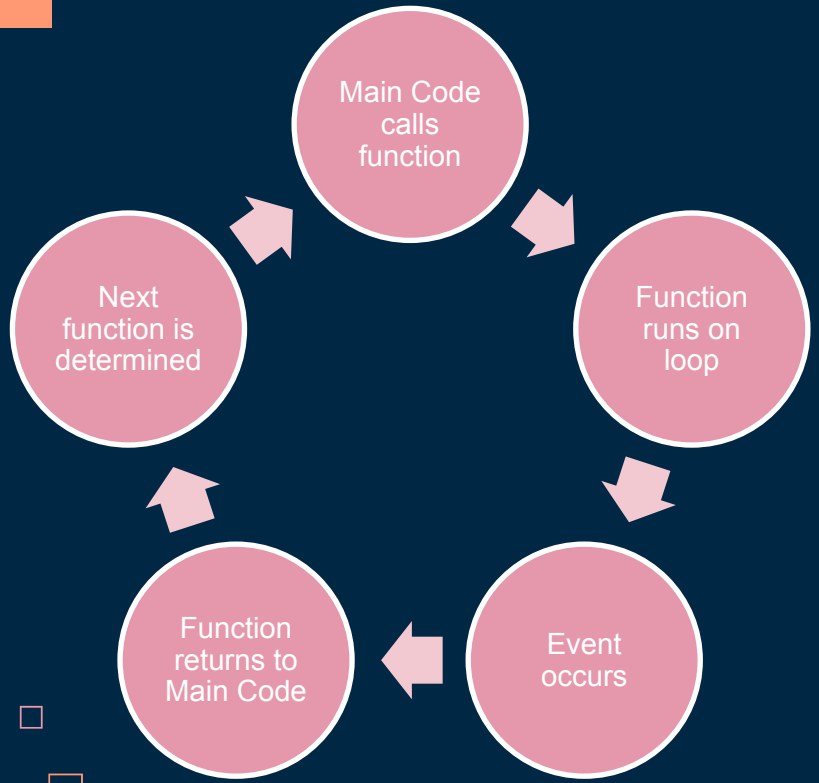


Final Robot Design

Design Specs

- Mass: 985.2 g
- Size: 31 x 21 x 13 cm
- Total Parts: 220
- Most Common Part: Black Technic Pin (x43)

The Build Process – Software and Logic




```
print(f"Moving to ({newX}, {newY}) at {angle}" + chr(176))
turn_absolute(angle)
hazard = drive_distance_gyro_assist(40, initialAngle = angle

if not hazard:
    mazeMap.append([newX, newY, 1])
    currentX = newX
    currentY = newY
```

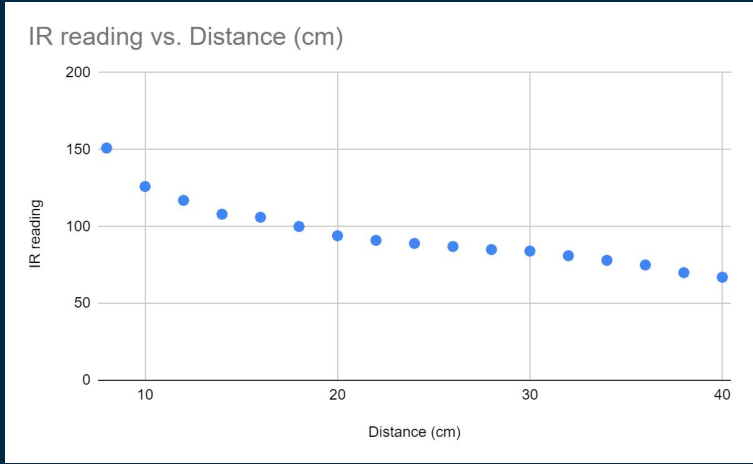
```
def drive_distance_gyro_assist(targetDistance, initialAngle = GYRO.value(), speed = 15, hazardDetect = True):
    MOTOR_RIGHT.reset_distance()
    distanceTraveled = MOTOR_RIGHT.get_distance()

    while (abs(distanceTraveled) < abs(targetDistance)):
        distanceTraveled = MOTOR_RIGHT.get_distance()
        val = IR_SENSOR.value()
        if (hazardDetect and (255 > val > 120)):
            print("IR Detected")
            drive_distance_gyro_assist(-distanceTraveled, -speed, hazardDetect = False)
            turn_angle(90)
            return [2, "IR Radiated Power (W)", val]

        val = IMU.get_mpu9250_mag()['z']
        if (hazardDetect and (abs(val) > 250)):
            print("Magnet Detected")
            drive_distance_gyro_assist(-distanceTraveled, -speed, hazardDetect = False)
            turn_angle(90)
            return [3, "Magnetic Field Strength (uT)", val]

        adjSpeed = ramp_speed(abs(speed), abs(distanceTraveled), abs(targetDistance-distanceTraveled)) * sign(speed)
        drive_gyro_assist(initialAngle, adjSpeed)

    brake_motors()
    return ""
```

The Build Process – Software and Logic

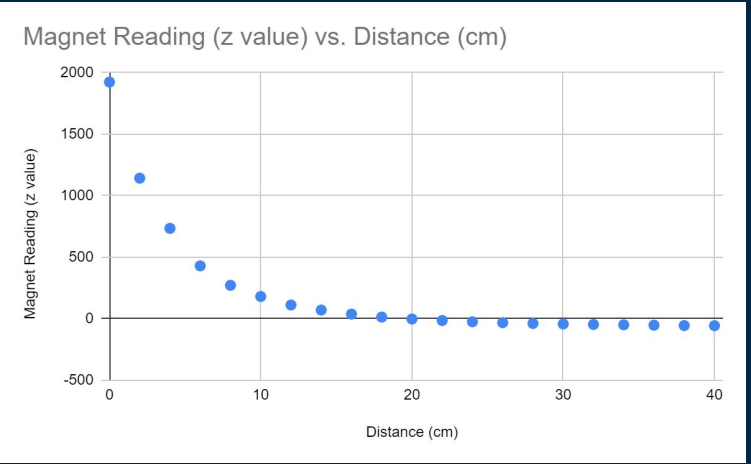
Sensor Reading Functions	Drive Motor Functions	Maze Mapping Functions	Cargo Motor Functions	Main Code
• wait_for_touch() • avg_ultrasonics()	• tank_steer() • drive_dist_gyro() • align_front_wall()	• coord_move()	• drop_cargo()	• Move forward one square • Turn based on ultrasonic reading • Repeat until end detected • Drop cargo

Testing Data

IR Sensor Calibration



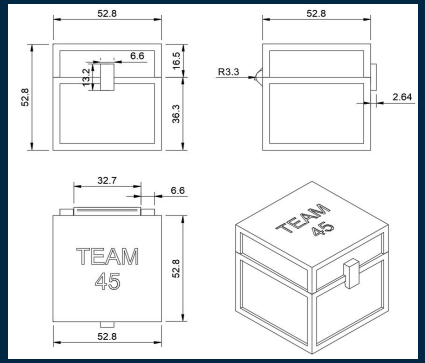
Magnetic Sensor Calibration

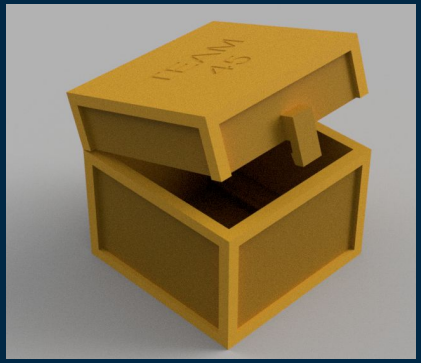


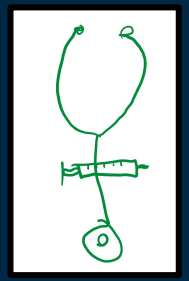
Cargo & Decals

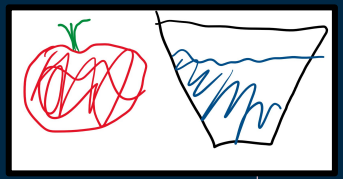
- Simple, effective container
- Aesthetic and easy to open

- Decal design philosophy: simple & child-like



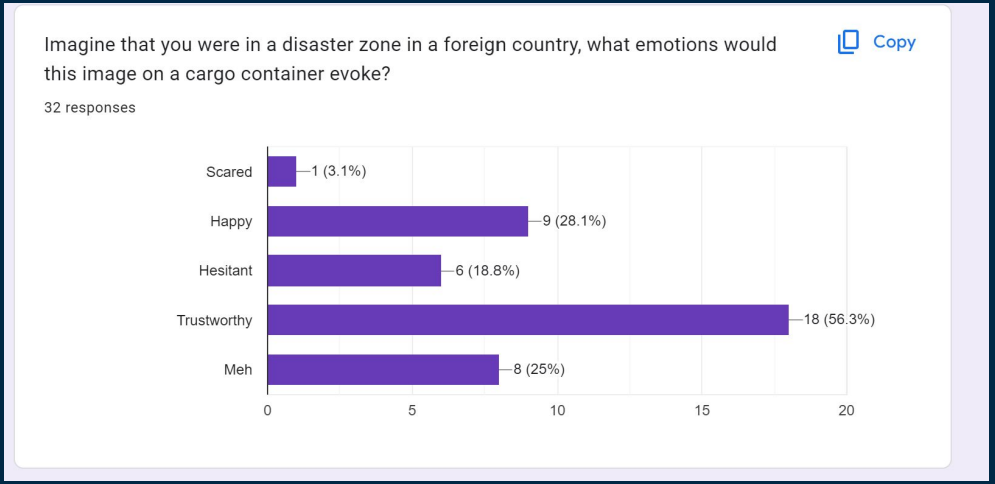


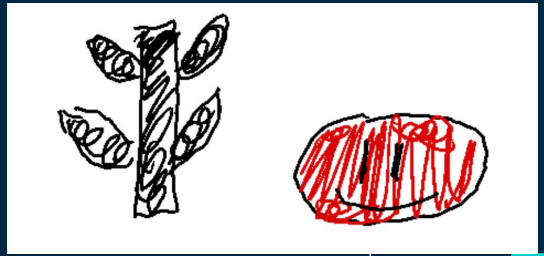


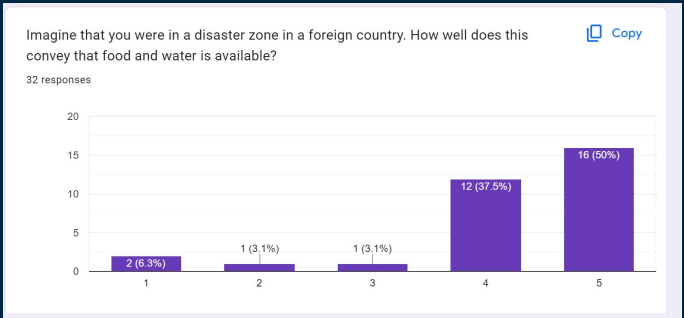


Decal Data Collection

- Produced a survey
  - Survey showed decal and asked two questions
  - Scores were similar for all decals chosen
- Repeated for all decals







Demo Performance

- Overall, performed well
- Achieved all tasks, but required checkpoints
- Successful mapping and hazard output
- Occasional gyro drifting, resulting in wall contact

Map Config No.	Pre-Run		Checkpoints					Data Output		Cargo	
	Symbols?	Cargo Check?	1	2	3	4	5	Resource Data CSV Generated?	Map CSV Generated?	Deposited Upright?	Signal After Deposit?
45	6.2	✓	✓	1	N/A	1/2	N/A	1	1	✓	✓
				A.1	A.2	B	C	Exit	<- Checkpoints correspond to the ends of different maze modules. Refer to setup doc. for specifics		

Pass = 1

Areas for Improvement

- Reducing error buildup
- Alignment with surroundings
- IR detection when off center
- Reduced gyro drift

Thank You

Any Questions?