Performance Validation Review for Pac-Man

Version 1

16-450: Systems Engineering

May 8th, 2018

Group Members: Mehar Singh, Jae-Eun Lim, Nathalie Domingo, Angel Macias
Group Name: General Robotics

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to analyze the requirements and feasibility of a physical Pac-Man game that is used for both entertainment and educational purposes. The need for such a robot is not only to entertain people but also provide a platform for children and young adults to learn more about robotic technologies and facilitate an experience that helps them get the most out of the content. In this document, we will outline the system constraints, interface, interactions with its environment and other functional requirements.

1.2 Need / Motivation

Robotics is rapidly transforming the way we live and work – from self-driving cars to industrial robots to vacuuming robots in our houses. However, despite their vast impact, most people are baffled by how a robot even works or operates. Our goal is to provide a platform to teach key concepts in robotics focusing around the field of path-planning in an innovative and engaging way outside of a classroom setting.

By re-creating the retro arcade game: Pac-Man, we can build a platform to teach various people the concept of path-planning in an engaging and thoughtful way with a real-life example to go along with it. The goal and motivation of this system is to de-mystify the field of robotics to help people better understand how robots work and to encourage the younger generation into learning more about these technologies.

1.3 Intended Audience

This document is intended to be used by System Engineers, Hardware and Software Engineers, Project Managers, Marketing Managers and Testers. The document provides key information about the development and goals of the system. These outlines and descriptions are very useful for all these various groups working on the system as it allows them to benchmark various requirements and work towards a common goal.

1.4 System Scope

The Pac-Man is a physical system that will re-create several aspects of the traditional game. The robotic system will include 3 main components. However, the system will contain a twist on the original version where the player will use the ghost instead.

1. **Pac-Man**: Pac-Man will be autonomous and his goal will be to finish the maze and eat all the dots.
2. **Ghost(s)**: The ghosts will be tele-operated by humans in order to prevent Pac-Man from accomplishing its goal.
3. **Interface with Audience**: The user interface will contain a diagram of what Pac-Man is trying to do and presents the path planning processes to teach students about robotics in a more compelling way.

The system will follow the same game rules that are typically found in Pac-Man games. The objective is for the player to prevent the Pac-Man robot from accomplishing his goal and catching up to it before Pac-Man eats all the dots.

The objective of designing and creating this system is not only to teach students about the field of robotics but also to give us an opportunity to apply systems engineering concepts to real-world robotic systems to better facilitate our designing process.

2. Use Cases

3.1 Use Case Diagram

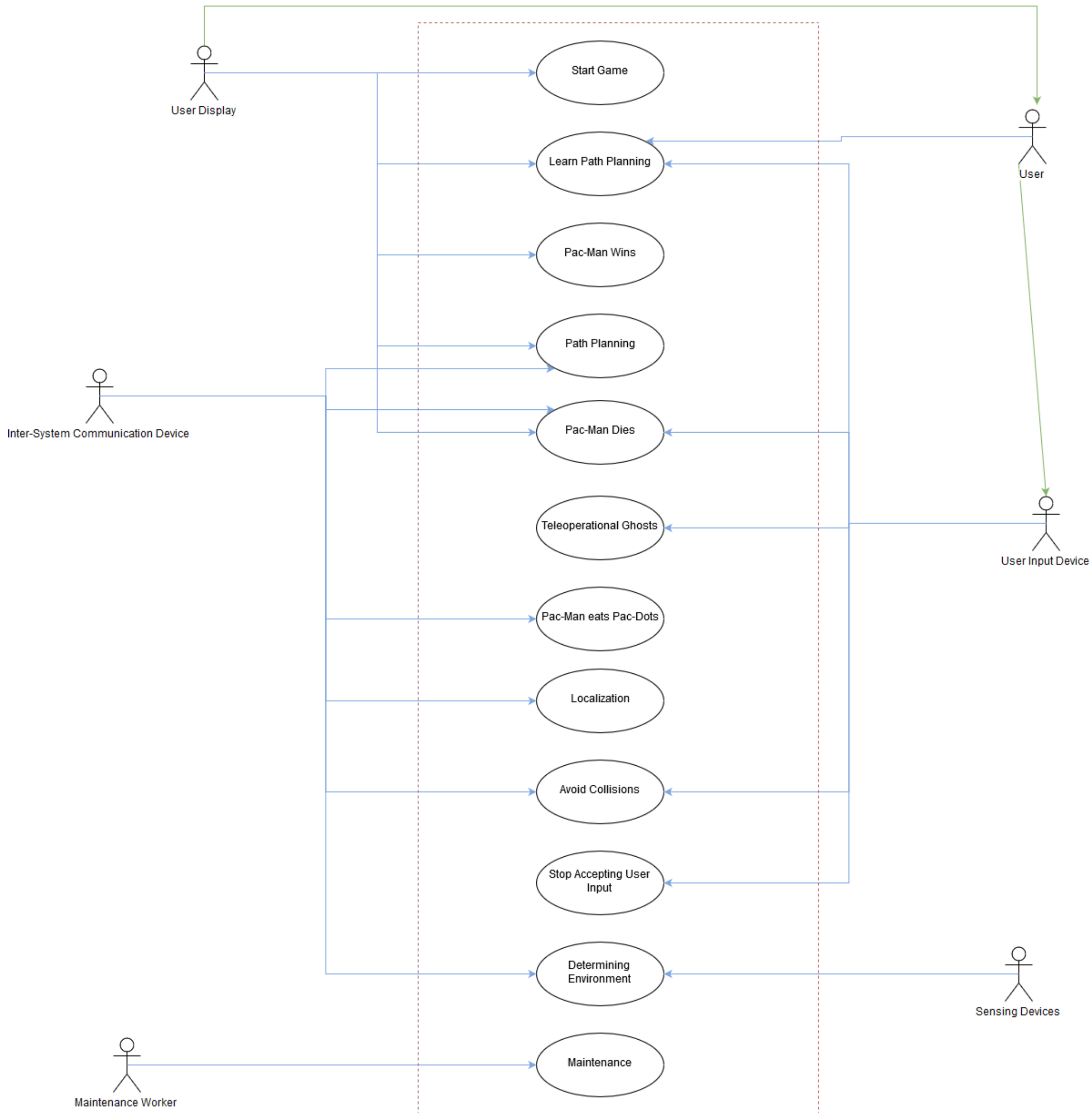


Figure 1: Use case diagram outlining the main actions the system should perform and their associated external users

3.2 Context Diagram

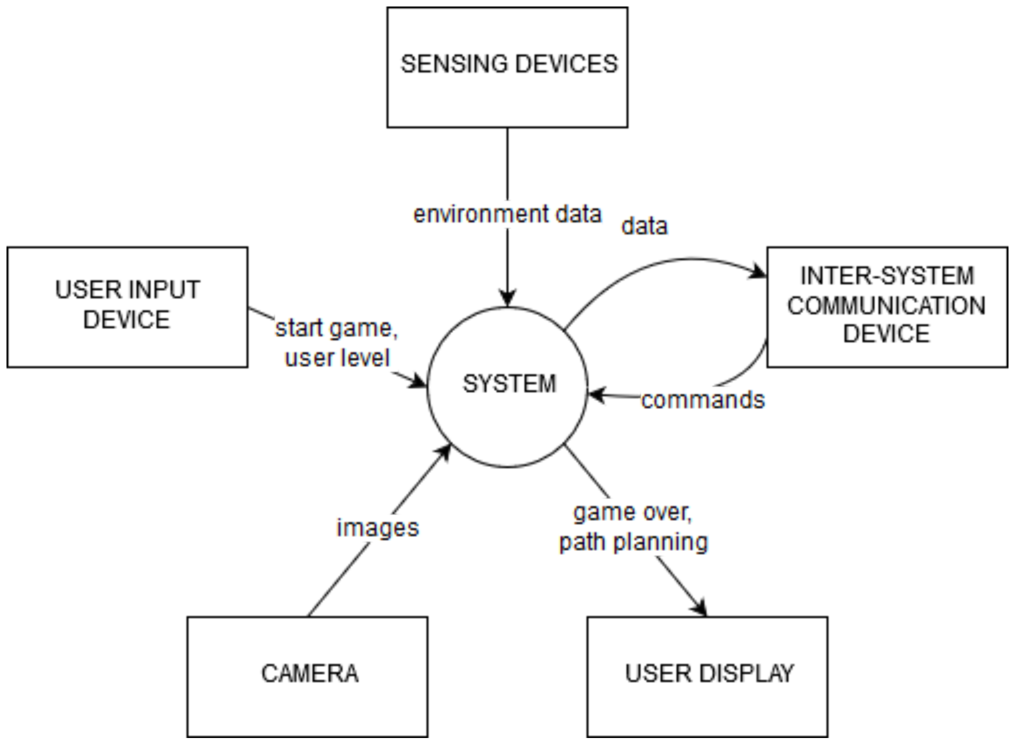


Figure 2: Context diagram outlining how external users interact with the system

3. Requirements

3.1 Assumptions

The following are the assumptions we make in the requirements specification for our system:

1. The system is installed in an indoor museum setting.
2. The system is installed in an environment with good lighting system.
3. The system is installed within a meter from an electric outlet for charging.
4. The user interacts only with the screen and the controller.
5. The user is literate.
6. The user is familiar with Pac-Man game.

3.2 Functional and Non-Functional Requirements

Education Requirements

<u>Functional Requirements</u>

R1.1: The system must provide a complete lesson on the path planning algorithm of Pac-Man to facilitate user's understanding of the overall concept.
R1.2: The system must provide different levels of education from beginning to advanced for literate users.
R1.3: The educational content must contain both images and texts for enhanced quality.

<u>Non-Functional Requirements</u>

R1.4: The educational content must be visibly displayed to both the user and the bystanders.
R1.5: The educational content must be well organized into sections for enhanced interface.
R1.6: The length of the educational content must not exceed 1 minute of reading time in order to balance education and entertainment.

Game Requirements

Functional Requirements

R2.1: The system must simulate dots being eaten by Pac-Man.
R2.2: The system must follow the original rule that Pac-Man wins the game when it eats all the dots.
R2.3: The system must follow the original rule that Pac-Man dies when the Ghost approaches Pac-Man.
R2.4: The Ghost must start from the designated starting region which is located at the center of the board.

Non-Functional Requirements

R2.5: The user must have the ability to win in the game in order to make the game more engaging.
R2.6: The background music played by the system must be original to that of the actual Pac-Man game.
R2.7: Any additional sound effects must have the purpose of improving user interface.

Maintenance Requirements

Functional Requirements

R3.1: The system must have an emergency stop.
R3.2: All parts on the system that require regular maintenance should be easily accessible.

Non-Functional Requirements

R3.3: The system must last 1 week before requiring maintenance.
R3.4: Regular maintenance must only take 1 hour.
R3.5: The system should have a lifespan of 1 year after which it is considered unrecoverable.

Charging Requirements

Functional Requirements

R4.1: All charging ports must be easily accessible.

Non-Functional Requirements

R4.2: It must only take maximum of 3 hours to fully charge the entire system.

Board Requirements

Functional Requirements

R5.1: The board must be self contained.
R5.2: The board must be waterproof.
R5.3: The board must be modular.

Non-Functional Requirements

R5.4: The board size should be no more than 5 ft. x 5 ft.

Sensing

Functional Requirements

R6.1: The Pac-Man robot in the maze must localize within the maze
R6.2: The Pac-Man robot in the maze must autonomously navigate within the maze.
R6.3 The mobile robots must be able to sense one another.
R6.4: The mobile robots must be able to avoid colliding due to their sensing of each other.
R6.5: The LED lights must have a method of sensing when a mobile robot passes over them.

Non-Functional Requirements

R6.6: The error in the sensed position of the mobile robots must be less than 0.5 cm.

Mobility

Functional Requirements

R7.1: The mobile robots must be able to move smoothly on the maze.
R7.2: The mobile robots must change directions smoothly.
R7.3: The maze must be smooth and the LEDs flush with the bottom of the maze.
R7.4: The mobile robots must stop moving if it is sensed that they are too close to each other.
R7.5: The mobile robots must not damage the maze while moving.

R7.6: The Pac-Man mobile robot must be able to autonomously plan and mobilize throughout the maze.

Non-Functional Requirements

R7.7: The mobile robots must have a constant speed of 4 cm/s whenever moving.
R7.8: The mobile robots must have only two degrees of freedom – they should only move up/down or left/right.

Communication

Functional Requirements

R8.1: The central processor must be able to send commands to the mobile robots.
R8.2: The user must be able to communicate movement commands to the Ghost robot through the central processor.
R8.3: The central processor must be able to command the LED array to shut off specific LEDs.
R8.4: A visual feed from a camera overlooking the maze must be communicated to the user.
R8.5: The locations of the robots must be communicated to a central processor.

Non-Functional Requirements

R8.6: The latency of the commands from the user to the Ghost robot must be less than 2ms.
R8.7: The latency of the commands from the central processor to the Pac-Man robot must be less than 2ms.
R8.8: The latency of the camera feed to the user must be less than 2ms.
R8.9: The latency of the commands from the central processor to the LED array must be less than 1ms.

User Interface

Functional Requirements

R9.1: The user interface must explain how to properly use the robots in the system.
R9.2: The user interface must contain a method for the user to change the difficulty of the game
R9.3: The user interface must explain to the user the rules of the game.

Non-Functional Requirements

R9.4: The general aesthetic must be friendly.
R9.5: The color scheme must match that of the Pac-Man game.
R9.6: The general aesthetic must match that of the Pac-Man game.
R9.7: The refresh rate of the interface must be at least 60hz.
R9.8: The refresh rate of the video feed of the maze must be at least 30hz.
R9.9: The input lag from the user must be under 15ms.

Decision Making Requirements

Functional Requirements

R10.1: The Pac-Man robot must autonomously plan actions that do no cause it to harm itself or other robots in the maze.
R10.2: The system must always predict and prevent collisions within the maze.

Non-Functional Requirements

R10.3: The Pac-Man robot must always choose the path that optimizes its score.

3.3 Requirements Priority Table

Priority Level	Requirements	
High	Education	R1.1, R1.3, R1.4, R1.5
	Game	R2.1, R2.2, R2.3, R2.5
	Maintenance	R3.1
	Charging	None
	Board	None
	Sensing	R6.1, R6.2, R6.3, R6.4, R6.5, R6.6
	Mobility	R7.4, R7.5, R7.6
	Communication	R8.1, R8.2, R8.3, R8.5, R8.6, R8.7, R8.8, R8.9

	User Interface	R9.2, R9.3, R9.7, R9.8, R9.9
	Decision Making	R10.1, R10.2
Medium	Education	R1.2, R1.6
	Game	R2.7
	Maintenance	R3.2, R3.3, R3.4
	Charging	None
	Board	R5.1, R5.2, R5.3, R5.4
	Sensing	None
	Mobility	R7.1, R7.2, R7.3, R7.7
	Communication	R8.4
	User Interface	R9.1, R9.4, R9.5, R9.6
	Decision Making	R10.3
Low	Education	None
	Game	R2.4, R2.6,
	Maintenance	R3.5
	Charging	R4.1, R4.2
	Board	None
	Sensing	None
	Mobility	R7.8
	Communication	None
	User Interface	None
	Decision Making	None

4. System Description

4.1 System Description

The physical Pac-Man game system provides entertainment via its game interface as well as education on path planning algorithm. The system consists of Pac-Man, a ghost, a game board, a controller, and a screen for user interface. The game board is a maze with LEDs laid out along the paths. Pac-Man moves autonomously to eat all the coins, indicated by the LEDs, and the user controls the ghost to catch Pac-Man before all the coins are eaten. The screen displays the educational content of the game. The system is geared for all ages with its varying levels of difficulty in the educational content.

The original Pac-Man game is shown below[1]. The system will recreate the general aesthetic of the game in our design.

4.2 Concept of Operation

In the entertainment aspect, the system interacts with one user who controls the ghost with the controller to catch Pac-Man. The ghost starts at the center of the game board, or the starting region. As the starting point for Pac-Man, the user can place Pac-Man anywhere at least two lanes away from the ghost's starting region. Pac-Man must autonomously escape from the ghost while eating all the coins in the paths indicated by LED lights on the game board. Each LED turns off as Pac-Man passes through it, which indicates that the coin has been eaten. The user wins the game when Pac-Man is caught by the ghost. The user loses the game when Pac-Man eats all the coins before being caught. Pac-Man is caught when the ghost is within 2 inches from it.

[1] https://www.arcade-museum.com/game_detail.php?game_id=10816

In the educational aspect, the system visually displays the localization of Pac-Man and the path it plans to take based on the location of the ghost. The level of details of the algorithm on display will depend on the user's choice of hard, medium, or easy.

4.3 Sub-Systems

Pac-Man

Pac-Man is an autonomous robot that uses path planning to stay away from the ghost while traversing every path to eat all the coins. Pac-Man has a propulsion system with motors and four wheels which allow it to move around the game board. Pac-Man uses sensing devices and the overhead camera to determine the relative location of the ghost. If the ghost is within 2 inches from Pac-Man, which means Pac-Man has been caught, the motor is locked and the game is over.

Ghost

The ghost is a robot that is tele-operated by the user to catch Pac-Man. It has a propulsion system with motors and four wheels which allow it to move around the game board. The ghost moves faster than Pac-Man to make winning feasible.

Game Board

The game board is a maze with LEDs laid out on its paths. The LEDs represent the Pac-Dots. The game board has a region at the center for the ghost's starting point. There is a camera above the board which tracks the path taken by Pac-Man and determines which LEDs to turn off. Each LED that turns off indicates a Pac-Dot that has been eaten.

Controller

The controller controls the movement of the ghost. The controller is a joystick that responds to forward and backward movement as well as left and right turns. The speed of the ghost is constant.

Screen

The screen provides the user interface. It displays the view of the camera mounted on the top of the system, which captures the entire game board, as a digitized Pac-Man game. An arrow is shown on the board next to Pac-Man to show the path Pac-Man plans to take.. Moreover, the screen shows a description of the path planning algorithm. The user has options to choose one of hard, medium, or easy in terms of level of details of the path planning algorithm described.

Computer Vision

The overhead camera will provide a video feed for the computer vision subsystem to process. This subsystem will determine the locations of the robots and the locations and orientations of the maze pieces. This subsystem will feed information into the path planner.

4.4 System Drawing

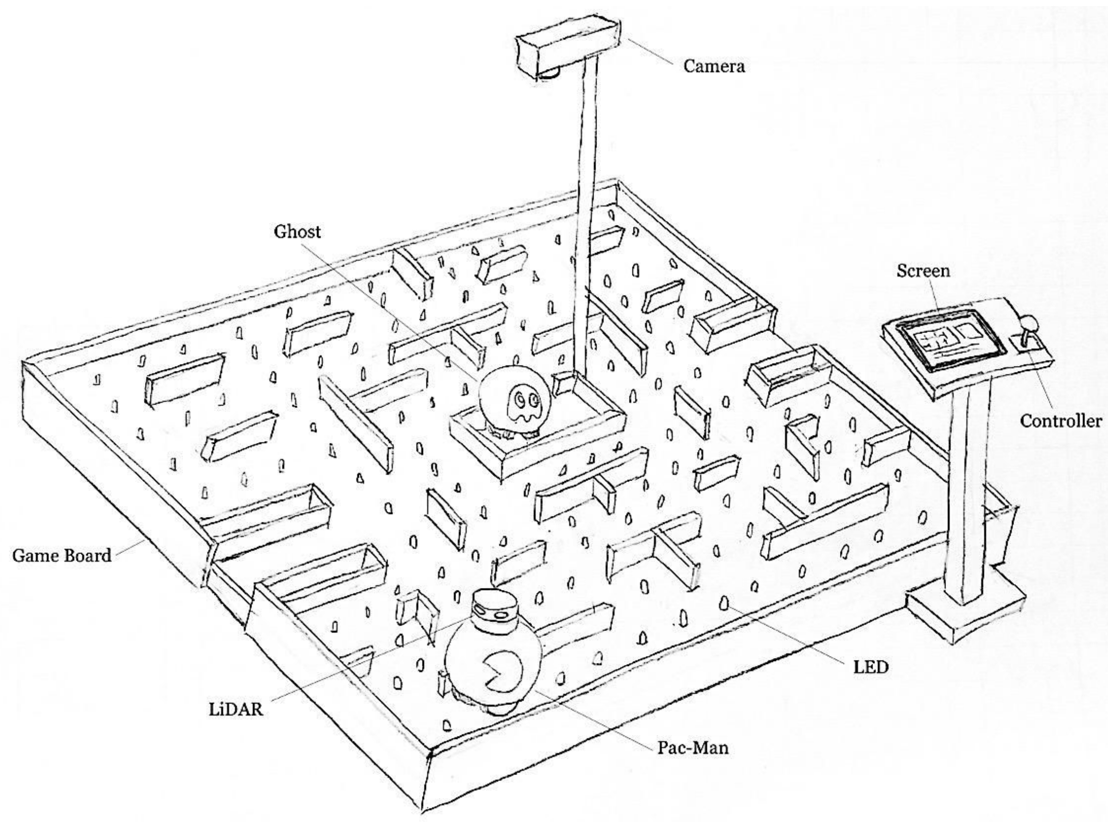


Fig. 1 - Physical Pac-Man Game System

4.5 Concept Design Treacability Matrix	Holonomic Drive	Bluetooth Communication	LED Grid	Overhead Camera	Joystick Controller	Difficulty Levels	One Charging Port for System	Starting Position of Pac-Man / Ghost	Screen Display	Audio System	Emergency Stop Button
	Holonomic drive for the Pac-Man and Ghost Robots for movement	Bluetooth Sensors on Pac-Man and Ghost Robots to determine relative location	Using an LED Grid as the maze to represent the Coins	An Overhead Camera above the system to conduct Computer Vision and utilized various algorithms and techniques from Localization and CV to send commands to Pac-Man robot	Controller that drives the Ghost robot from the user inputs	User Interface describes and sets difficulty levels to explain the path-planning algorithms to the users	One Charging Port on the entire system to make maintenance of the system easier	User can choose the starting position of the Pac-Man (with the limitaiton that it has to be 2 rows away from the Ghost) to make the system more engaging and user-defined but the Ghost position is always fixed	The Screen to display the User Interface. It should include a Menu Page to select settings, Game Rules page to properly explain how to use the robots. It should do so using bullet points to be succint	There will be an audio system to play game-music to make the system more engaging and entertaining	There will be an emergency stop button to shut off the system, if required
Satisfied Requirements											
EDUCATION											
R1.1									✓		
R1.2						✓					
R1.3									✓		
R1.4									✓		
R1.5								✓	✓		
R1.6									✓		
GAME											
R2.1			✓								
R2.2			✓								
R2.3		✓									
R2.4								✓			
R2.5								✓			
R2.6										✓	
R2.7										✓	
MAINTENANCE											
R3.1											✓
R3.2											
R3.3											
R3.4											
R3.5											
CHARGING											
R4.1							✓				
R4.2							✓				
BOARD											
R5.1			✓				✓				
R5.2							✓				
R5.3			✓								
R5.4			✓								
SENSING											
R6.1				✓							
R6.2				✓							
R6.3		✓		✓							
R6.4		✓									
R6.5			✓	✓							
R6.6				✓							
MOBILITY											
R7.1	✓										
R7.2	✓										
R7.3			✓								
R7.4		✓									
R7.5				✓							
R7.6				✓							
R7.7	✓										
R7.8	✓										
COMMUNICATION											
R8.1											
R8.2					✓						
R8.3											
R8.4				✓					✓		
R8.5											
R8.6					✓						
R8.7											
R8.8				✓							
R8.9											
USER INTERFACE											
R9.1									✓		
R9.2									✓		
R9.3									✓		
R9.4											
R9.5											
R9.6			✓						✓		
R9.7									✓		
R9.8				✓					✓		
R9.9					✓						
DECISION MAKING											
R10.1				✓							
R10.2				✓							
R10.3				✓							

4.5 Concept Design Treacability Matrix	Casing for Pac-Man / Ghost Robot	Replaceable and Modular Components	Central Processor
	There will be a casing around the mobile robots in order to make the system more aesthetic as well as mimic the style of the traditional game	Every component of the subsystems will be modular and repleacable when worn out	A central processor to control the robots and LEDs on the board
Satisfied Requirements			
EDUCATION			
R1.1			
R1.2			
R1.3			
R1.4			
R1.5			
R1.6			
GAME			
R2.1	✓		
R2.2			
R2.3			
R2.4			
R2.5			
R2.6			
R2.7			
MAINTENANCE			
R3.1			
R3.2		✓	
R3.3		✓	
R3.4		✓	
R3.5		✓	
CHARGING			
R4.1			
R4.2			
BOARD			
R5.1			
R5.2			
R5.3			
R5.4			
SENSING			
R6.1			
R6.2			
R6.3			
R6.4			
R6.5			
R6.6			
MOBILITY			
R7.1			
R7.2			
R7.3			
R7.4			
R7.5			
R7.6			
R7.7			
R7.8			
COMMUNICATION			
R8.1			✓
R8.2			
R8.3			✓
R8.4			
R8.5			✓
R8.6			
R8.7			✓
R8.8			
R8.9			✓
USER INTERFACE			
R9.1			
R9.2			
R9.3			
R9.4	✓		
R9.5	✓		
R9.6	✓		
R9.7			
R9.8			
R9.9			
DECISION MAKING			
R10.1			
R10.2			
R10.3			

5. Scenarios

Scenario 1: Pac-Man "eats" dots
Summary: As Pac-Man autonomously traverses the board, the LEDs turning off represent Pac-Man eating the dots.
Actors: camera, inter-system communication device
Preconditions: The system is on and running as usual. A game is in session.
Description: As Pac-Man traverses the board, the camera above the entire board tracks Pac-Man. As the board is tracking Pac-Man, it commands the LEDs that Pac-Man passes to turn off. This gives the illusion that Pac-Man is eating the dots.
Post condition: LEDs in the board under where Pac-Man has passed are turned off, while those Pac-Man has not gone over are on.

Scenario 2: Teleoperation
Summary: Users are utilizing the joystick in order to command the ghosts.
Actors: user, user input device, inter-system communication device
Preconditions: The system is on and running as usual. The user just started the game by interacting with the screen
Description: The user is using the joystick to command the ghost. The ghost responds accurately and timely to the user input.
Post condition: The ghost has moved in the direction the user commanded.

Scenario 3: Path planning for Pac-Man
Summary: Pac-Man determines the optimal path to both avoid the ghosts and to eat all the dots on the board.
Dependencies:
Includes: Localization, Determining the Environment
Actors: camera, sensing devices, inter-system communication device
Preconditions: The system is on and running as usual. A game is in session.
Description: Pac-Man, using information gathered through localization and determining the environment, figures out the optimal path that it should take. Pac-Man is keeping track of where it has already been, and thus knows where it needs to go to eat the rest of the dots. In this path planning algorithm, the location of the ghosts, determined by the individual ghosts and communicated to Pac-Man, is also considered. Once a path has been determined, this data is transferred to the user interface through the inter-system communication device. This path will be dynamically changing in order to account for the random movement of the user operated ghosts.
Alternatives: Path planning stops once Pac-Man has eaten all the dots, and thus won, or when Pac-Man has been eaten by a ghost, and thus loses.
Post condition: Pac-Man has an optimal path, in terms of avoiding the ghosts and eating dots, to follow, and moves accordingly.

Scenario 4: Localization
Summary: Pac-Man localizes itself.
Dependencies:

Includes: determining the environment
Actors: camera, sensing devices, inter-system communication device
Preconditions: The system is on and functioning as usual. A game has just started.
Description: Pac-Man is trying to determine its location in a given map. It uses the camera to and the sensing devices, which is receiving information on how far away the ghosts are, to determine its location. Additionally, it makes use of the information obtained from the environment. This is done at the beginning of a game, and as such there is a delay between when the user begins the game and when they can start moving the ghosts. This delay is at most 2ms, but also mimics game play in the original Pac-Man game.
Post condition: Pac-Man knows where it is located on the board.

Scenario 5: Determining the environment
Summary: Pac-Man uses computer vision and sensing devices to determine what's in its environment.
Actors: camera, sensing devices
Preconditions: The system is on and functioning as usual. A game is in progress.
Description: Pac-Man uses the sensing devices and a computer vision algorithm to determine all of the objects around. This includes the walls, openings, and ghosts.
Post condition: Pac-Man has collected information about the environment that will be utilized in Path Planning and Localization.

Scenario 6: Avoid collisions
Summary: The ghosts and Pac-Man do not collide.
Actors: sensing devices
Preconditions: The system is on and functioning as usual. A game is currently in process.
Description: The controllers of the ghosts and Pac-Man recognize that they are within 1 inch distance to each other. This causes both Pac-Man and the ghosts to stop. Specifically for the ghosts, they are no longer accepting user input. This holds true for any combination of ghosts and Pac-Man.
Post condition: The system no longer has moving components and the game is over.

Scenario 7: Stop accepting user input
Summary: The ghosts no longer accept user input.
Actors: user input device, user display, and user
Preconditions: The system is on and functioning as usual. A game is in progress.
Description: The system determines that it should no longer accept user input. This is usually because the game has ended. The system ignores the commands coming in from the user input device and the screen.
Post condition: The system is not accepting input from the user.

Scenario 8: UI
Summary: user selects appropriate learning level and starts the game
Actors: user, user display
Preconditions: The system is on and running as usual. The system is waiting for a game to start.

Description: The system displays several options to the user, (beginner, intermediate and advanced), which indicates the level of path planning the user is currently at. Based on the level chosen, the system displays the robot's path planning algorithm. Additionally, the system displays a digitized version of the overhead view of the system, obtained by the camera. The system waits for the user to press a button to begin a game.
Post condition: A game has begun and the system is displaying Pac-Man's path planning algorithm based on the user level selected.

<u>Scenario 9:</u> Pac-Man "dies"
Summary: Pac-Man is eaten by the user operated ghost
Dependencies:
    Includes: Avoid Collisions, Stop Accepting User Input
    Extends: UI
Actors: user display
Preconditions: The system is on and functioning as usual. A game is currently in process.
Description: The ghost, which is controlled by the user, comes within 1 inch to Pac-Man. This means the ghost has "eaten" Pac-Man and the user has won the game. This is indicated to the user through the display. The ghost is no longer accepting user input. The system waits a time 1 minute after which Pac-Man and the ghosts go back to their starting positions.
Alternatives: The game is manually shut off, at which point the system is not ready for another game.
Post condition: The game is over and the user knows that they have won. The system is ready for another game.

<u>Scenario 10:</u> Pac-Man "wins"
Summary: Pac-Man has eaten all the dots on the board.
Dependencies:
    Includes: Path Planning, Stop Accepting User Input, Pac-Man "eats" dots
    Extends: UI
Actors: user display
Preconditions: The system is on and functioning as usual. A game is currently in progress.
Description: Pac-Man has traversed the entire board and has "eaten" all the Pac-Dots. All the LEDs are off, indicating that they have all been eaten. This means the user has lost the game, and this is indicated to the user through the display. The ghost is no longer accepting user input. The system waits a time 1 minute after which Pac-Man and the ghosts go back to their starting positions.
Alternatives: The game is manually shut off, at which point the system is not ready for another game.
Post condition: The game is over and the user knows that they have lost. The system is ready for another game.

<u>Scenario 11:</u> Manual charging of the system
Summary: The maintenance worker decided to manually put the system to charge.
Actors: Maintenance worker
Preconditions: The system is on and functioning as usual. It has not reached its critical power level, but it is also not fully charged.

Description: The maintenance worker shuts off the system. They then proceed to charge the separate components of the system.
Alternatives: The charging is interrupted by the maintenance worker, who has unplugged any of the system components.
Post condition: The system is fully charged.

<u>Scenario 12:</u> Maintenance of the system
Summary: The system must be shut off for maintenance, which can include changing motors and changing LEDs.
Dependencies:
    Includes: System shut off
    Extends: System check
Actors: maintenance worker, shut off switch
Preconditions: The system gives an alert indicating something is wrong with the system and it cannot operate as usual. The maintenance worker received the alert and is not ready to service the system.
Description: The maintenance worker shuts the system off through a switch in the system that is hidden from the public. The maintenance worker then performs the necessary actions to get the system to operate as usual. The maintenance worker then turns the system back on.
Post condition: The system is ready for a self-diagnostic test (System Check scenario).

<u>Scenario 13:</u> System shut off
Summary: The system shuts off
Actors: maintenance worker
Preconditions: The system is on and functioning as usual. A game could be in progress, game could have just ended, or the system could be idle.
Description: The maintenance worker shuts the system off through the system administrator shut off. The system responds by stopping all game activity and returning to start positions. After which the system shuts off.
Post condition: The system is off.

<u>Scenario 14:</u> System Check
Summary: System self-diagnostics
Dependencies:
    Extends: Maintenance of the System
Actors: camera, user input device, user display, sensing devices, maintenance worker
Preconditions: The system just went through maintenance.
Description: The system indicates to the maintenance through the display that it is undergoing self-diagnostics. The system checks all sensors individually, including the sensing devices and camera. The maintenance worker is in charge of determining whether the sensor data is correct. After all sensors are checked, the system waits for the worker to use the user input device in order to test user input.
Post condition: The system has checked all components and has determined that it is all working properly. The system is now ready to start a game.

6. Trade Studies

Trade Study 1: Some way of representing Pac-Man "eating" the dots—want to use LEDs to represent the dots and want them to turn off when Pacman passes by
Chose: Option 1
Option 1: Camera over the entire system
Description: Have a camera over the entire system, and utilizing computer vision, determine where on the board Pacman is located. The camera must track Pacman over the run of the entire game and turn the LEDs off accordingly.
Option 2: Pacman communicates to LED controller where it is
Description: Since Pacman already has to keep track of where it is (localization) for path planning purposes, Pacman can also transmit this information to the LED controller.
Option 3: Sensors along the entire board
Description: Have sensors along the entire board that recognize only when Pacman passes by, and thus turns off the corresponding LED.

	Weight (1-5)	Pac-Man onboard sensing		Overhead camera		Sensors along board	
Cost	1	Might require additional sensors on Pac-Man to localize onboard	3	Already have overhead camera for user display, no additional costs	5	Need a lot of sensors because board is large	1
Power Consumption	5	If additional sensors required, will need more power available onboard	3	Already have overhead camera for user display, no additional power required	5	A lot of sensors, so will consume a lot more power	1
Time to Create	3	Need to wire smaller amount of sensors and write a program to send over information to LED controller	3	Already have the camera, just need to write a program to separate the necessary information	4	Time required to wire all the sensors and to program	3
Skills Required	4	A lot of information on sensors for localization, still need to transfer data to LED controller	4	Need computer vision algorithms to separate Pac-Man and then localize in an internal map	4	A lot of information available on using proximity sensors	5
Simplicity	5	Need to possible add additional sensor	3	Already have the overhead camera	5	Need to use a lot more sensors	1
Capability	3	Sensors have minimum and maximum distances	4	Computer vision algorithms pretty accurate	5	Sensors have minimum and maximum distances	4
TOTAL			**70**		**98**		**52**

Trade Study 2: Sensors for path planning –determining location
Options: LiDAR and camera
Chose: Option 2
Option 1: LiDAR sensor
Description: Use a 360 LiDAR sensor for computer vision
Option 2: camera
Description: Use multiple cameras for computer vision.

	Weight (1-5)	**LiDAR**		**Camera**	
Cost	1	expensive for an accurate one--which we will need	2	Not that expensive for quality cameras	4
Skills Required	4	Fine tuning will be necessary	3	A lot of computer vision libraries available	4
Simplicity	5	A lot more onboard processing necessary, need more processing power on Pac-Man	2	Already have the overhead camera	4
Capability	3	Will take a while to localize since it is onboard, need to find distinctive map characteristics	4	Have overhead view of the entire map, can determine location easily	5
TOTAL			**36**		**55**

7. System Components

7.1 Hardware Components

<u>Pac-Man</u>

The components that are of particular importance for the Pac-Man subsystem are the main processor board. The main processor will likely be a Raspberry Pi since it needs to be connected to various onboard sensing devices, including Wi-Fi. The Wi-Fi module is needed for communicating the path planning data to a display for the user to see, as well as possibly offloading some path planning processing in the case that the Raspberry Pi becomes overloaded.

The motors and wheels need to be configured in such a way that allows for holonomic motion – i.e. Pac-Man doesn't need to turn to change directions. A battery needs to be attached to the central processor to power the subsystem remotely. Finally, these components need to be housed in a spherical shell to resemble the real Pac-Man game.

<u>Ghost</u>

The Ghost subsystem components are the same as that of the Pacman robot, except for two differences. First, the Wi-Fi module is instead used for receiving tele-operation commands from an external user – thus no autonomy. Second, because this robot is not autonomously navigating the maze, it doesn't need as many onboard sensing devices. Otherwise, the encasement and locomotion are the same as that of the Pac-Man robot.

<u>Maze Board</u>

The platform upon which the robots will move will be a flat board as the base with a maze structure, also from wood, on top. The maze needs to be painted black for a greater allusion to the real Pac-Man game. The LED strips need to be laid on the base of the maze to simulate the dots of the Pac-Man game.

<u>LED Dots</u>

The LED Dot subsystem needs a microcontroller, such as a Raspberry Pi, to control the individual LEDs and receive information from the Pac-Man robot through a Wi-Fi module. Additionally, the subsystem needs an array of LEDs to lay upon the maze board. Finally, a battery is necessary for powering the subsystem.

User Interface

The user interface for the Ghost robot will be displayed through a laptop or desktop monitor. The display data will be shown through either MATLAB or a custom app. A physical controller will be necessary (e.g. an Xbox controller) for the user to provide directional input to the ghost robot.

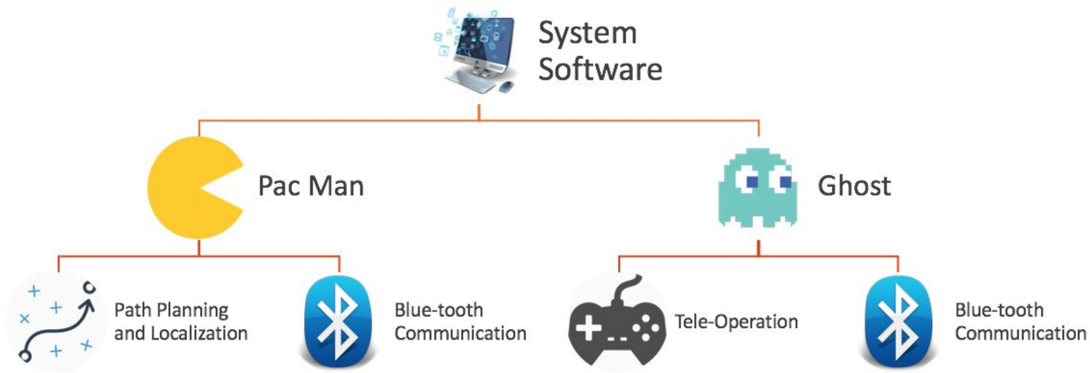


Fig. 3 - Necessary Components in System Software

Fig. 4 - Necessary Components in User Interface

7.2 Software Components/Architecture

The next three figures show some software architecture concepts. The first shows the overall idea of the system software by showing which subsystems interact. Figure 6 shows the general plan of how the path planning process and actuation will occur. Finally, figure 7 shows a more detailed view of the camera subsystem and path planning subsystem interaction, and how their processes end up feeding into the user interface subsystem.

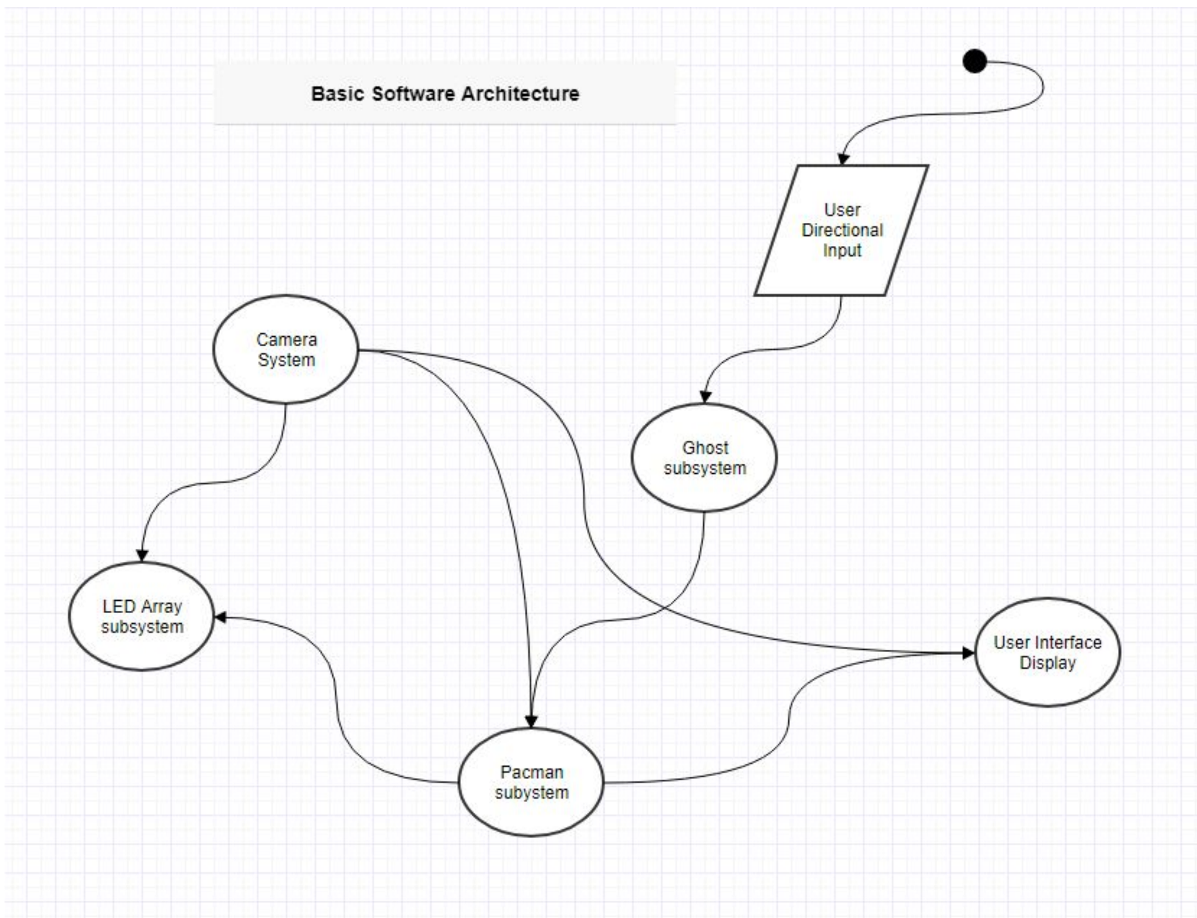


Fig. 5 - Overview of Software Architecture

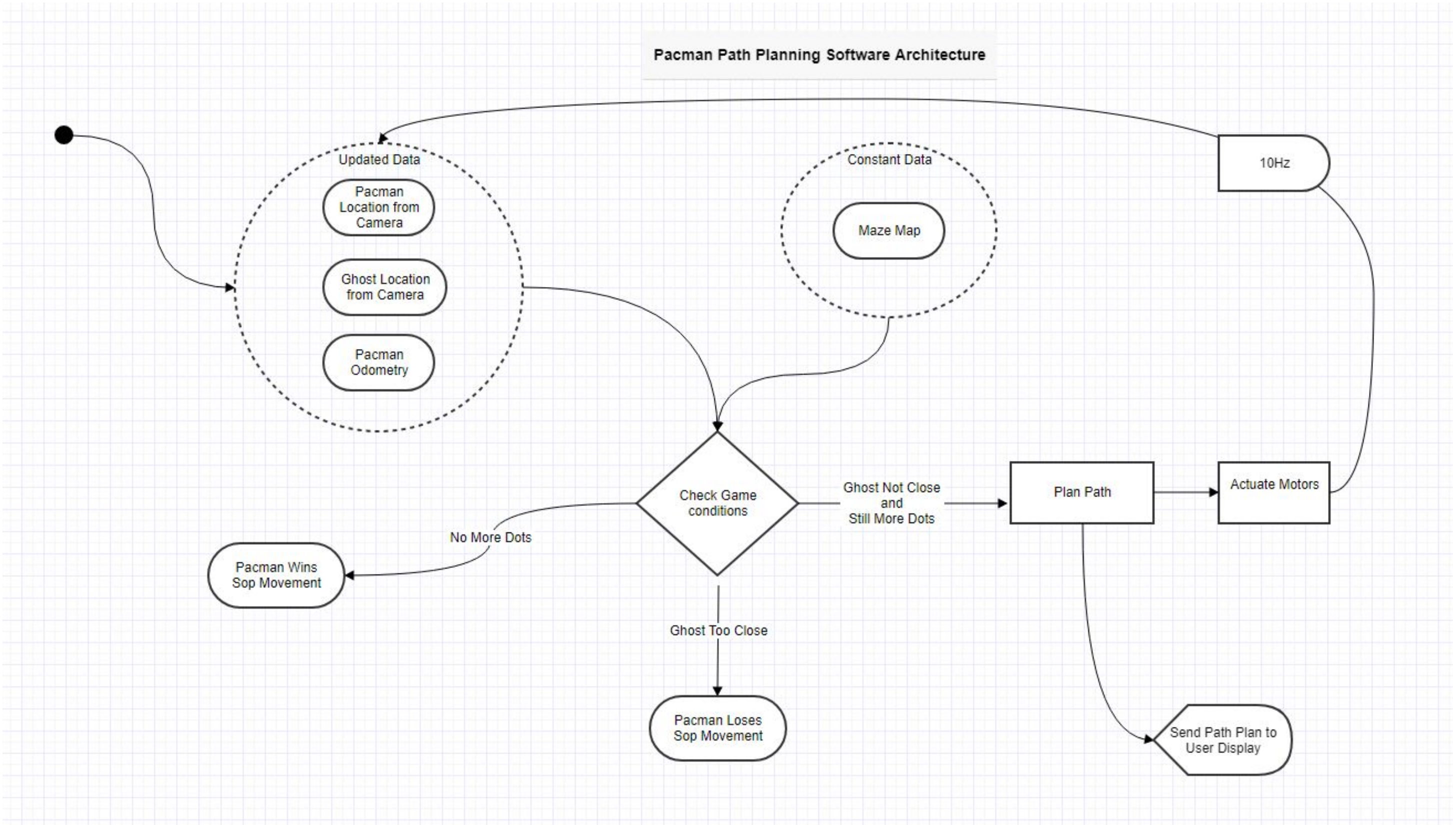


Fig. 6 – Path Planning through Camera

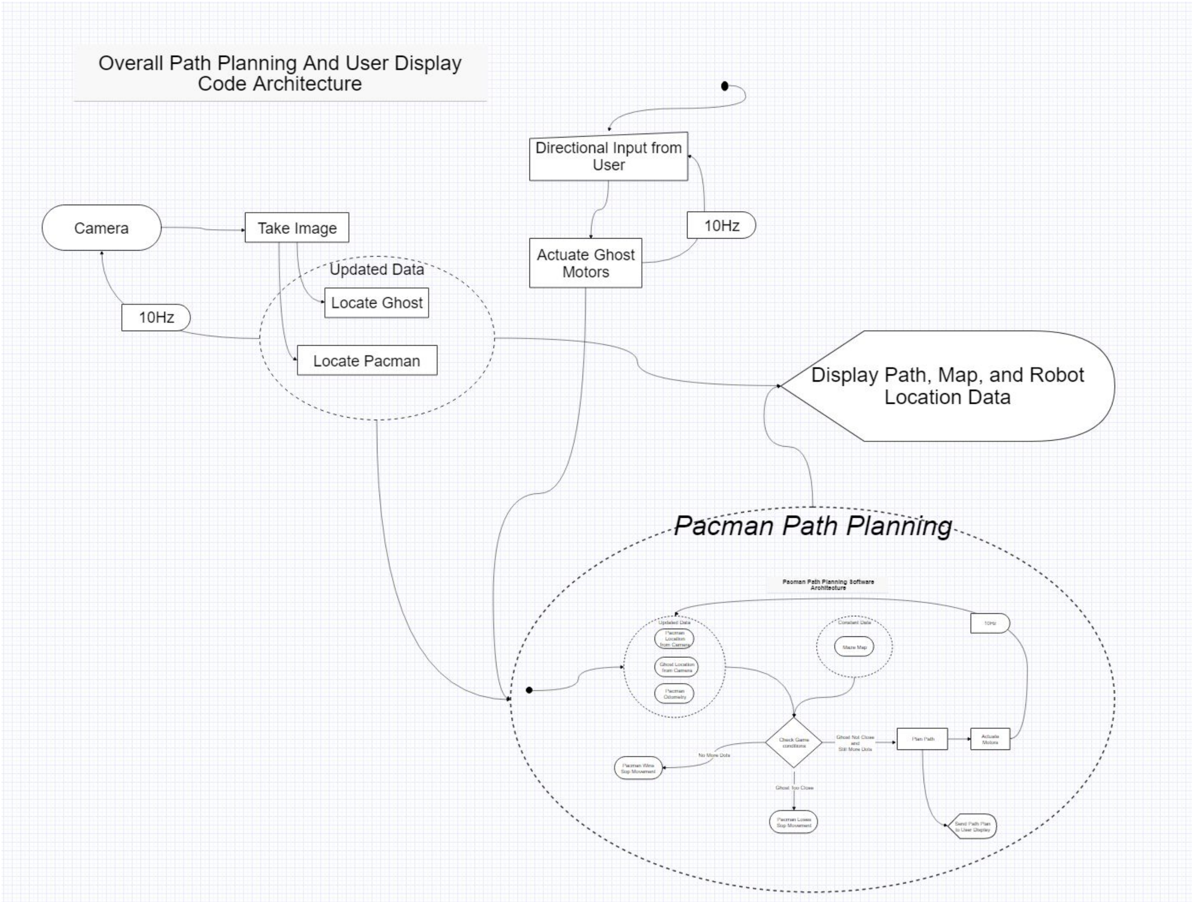


Fig. 7 - Overview of Camera-Based Path Planning

8. System Deployment

8.1 Deployment Model

The subsystems that need to be deployed onto hardware are the computer vision subsystem, path planning subsystem, robotic controls subsystem, user interface subsystem, communication subsystem, and the LED control subsystem.

The communication subsystem will consist of a router, camera, planner (central processor), robot boards (i.e. the boards the Pac-Man and Ghost robots use), and the LED control boards. This subsystem will be deployed onto the planner. The robot and LED boards will receive information from the planner, the planner will receive information from the camera subsystem, and the router will transmit/route the information between the necessary subsystems.

The path planning subsystem will be deployed onto the planner, and will receive information from the camera subsystem through the communication subsystem. Using the communication subsystem, the path planning subsystem will transmit the necessary commands to the robots on the board.

The LED controls subsystem will be deployed onto the LED control boards and the planner, and will use the communication subsystem to transmit the necessary control commands.

The robotic controls subsystem will be deployed onto the robot boards and the planner, and will use the communication subsystem to transmit the necessary control commands.

The user interface subsystem will contain the display and planner - it will be deployed solely onto the planner. This subsystem will take inputs from the camera subsystem and path planning subsystem to display results onto a screen.

8.2 Prototyping Activity

In order to develop the Pac Man system, we will need to prototype various components of the system individually before testing the whole integrated system.

<u>LED board</u>

We will need to prototype and test an LED Board. This LED Board will be used to represent the 'dots' that Pac-Man will have to 'eat'. This board will have to be created to ensure all the LEDs turn on and tested to ensure that when Pac Man goes over one, it will turn off.

Tele-operation

The tele-operation system will have to be tested and prototyped. This system will have to be created to ensure that the Ghost moves in the correctly and accurately. Moreover, we will have to test the system as it will be in the user's hands who could potentially misuse the robot in any fashion. Therefore, the system will have to be prototyped to minimize these types of interactions.

Robot Movement

We will also need to prototype the movements of the robot. After an initial prototype of the robot, we will need to test its movement capabilities to ensure that it can reliably move around the maze, particularly when turning tight corners.

User Interface

We will have to prototype the user interface program. This program will be the main source of information to users as well as a source to convey and aid learning more about robotic path planning to students. The user interface will have to be prototyped and created and repeatedly test to ensure accuracy and quality as well as a good user experience throughout the process.

8.3 Installation Plan of System

The installation plan for the Pac Man system is relatively simple. There must be enough space to fit the system as a whole into the room. The system will be manufactured in a box - approximately the size of 1m by 1m. Therefore, there should be enough space for the system to fit into the room.

The system will also require charging ports nearby to ensure the iPad device that the user interface is built upon is always sufficiently charged as well as our system is charged so that it does not run out of power.

The user flow of the system will be guided by our user interface that will tell the users that their turn with the game is up. The manual and set of instructions will also be provided through the user interface to minimize misuses of the robotic system.

9. Introduction

9.1 System Description

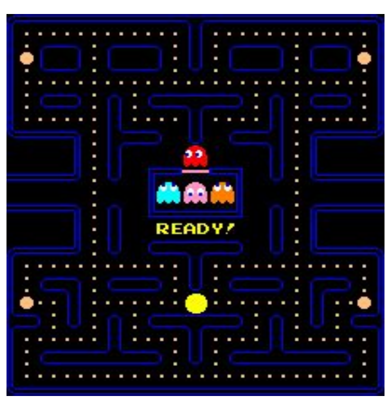
The robotic system will be a physical Pac-Man game that provides not only entertainment through its game interface but also an educational platform on path planning algorithms. The system consists of Pac-Man, a ghost, a game board, a controller, and a screen for user interface.

The game board will be a maze with LEDs laid out along the paths. Pac-Man moves autonomously to eat all the coins, indicated by the LEDs, and the user controls the ghost to catch Pac-Man before all the coins are eaten. The screen displays the educational content of the game and shows Pac-Man's path planning. The system is geared for all ages with its varying levels of difficulty in the educational content.

The system contains a twist, however. Unlike the traditional Pac-Man game, the player interacts with and controls the ghost via tele-operation and is trying to catch-up to Pac-Man. The ghost will start at the center of the game board, or the starting region. As for the starting point of Pac-Man, the user can place Pac-Man anywhere at least two lanes away from the ghost's starting region. Pac-Man must autonomously escape from the ghost while eating all the coins in the paths indicated by LED lights on the game board. Each LED turns off as Pac-Man passes through it, which indicates that the coin has been eaten. The user wins the game when Pac-Man is caught by the ghost. The user loses the game when Pac-Man eats all the coins before being caught. Pac-Man is caught when the ghost is within 2 inches from it.

The educational aspect of the system will be displayed through a screen that illustrates the localization of Pac-Man and the path it plans to take based on the location of the ghost. The level of details of the algorithm on display will depend on the user's choice of hard, medium, or easy.

The original Pac-Man game is shown below[2]. The system will recreate the general aesthetic of the game in our design.



[2] https://www.arcade-museum.com/game_detail.php?game_id=10816

9.2 System Diagram

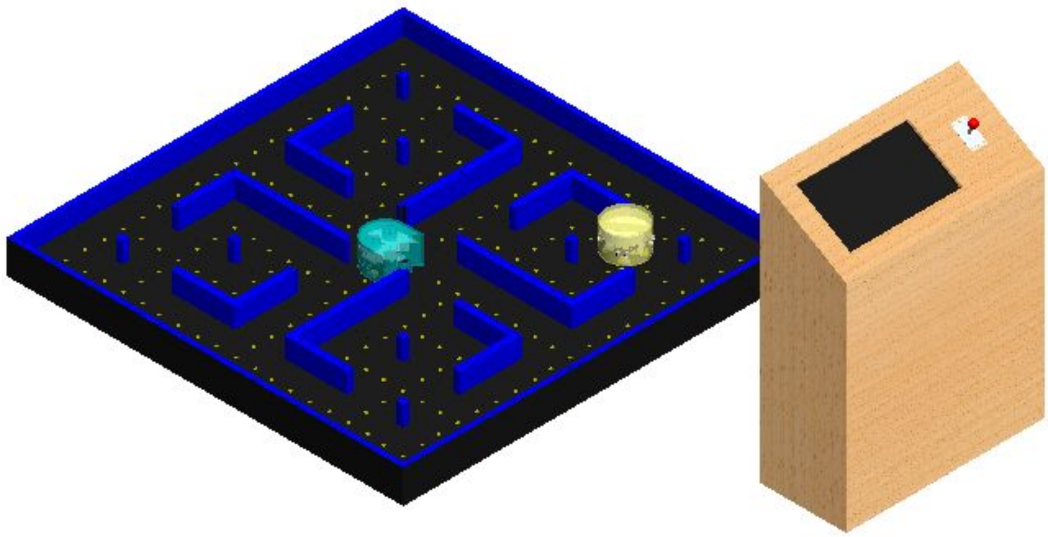


Fig. 1 - Isometric View of the System

9.3 Sub-System Diagrams

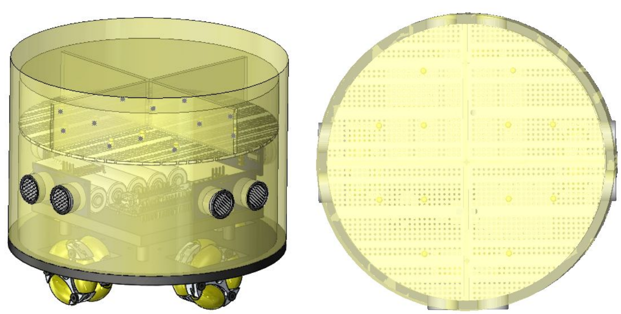


Fig. 2 - Isometric and Top Views of Pac-Man

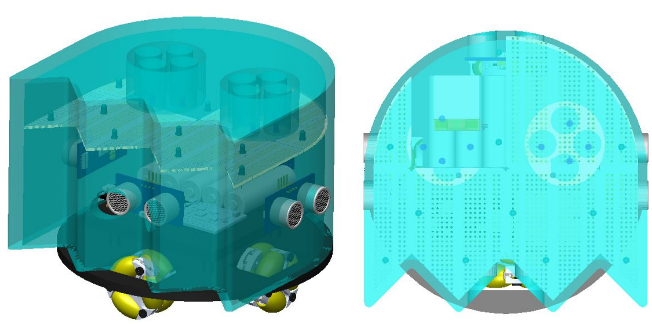


Fig. 3 - Isometric and Top Views of Ghost

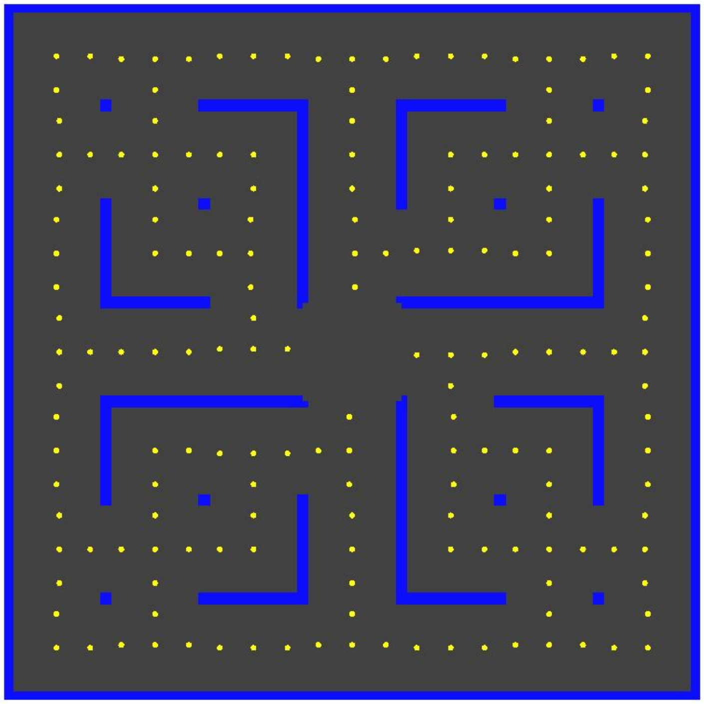


Fig. 4 - LED Board

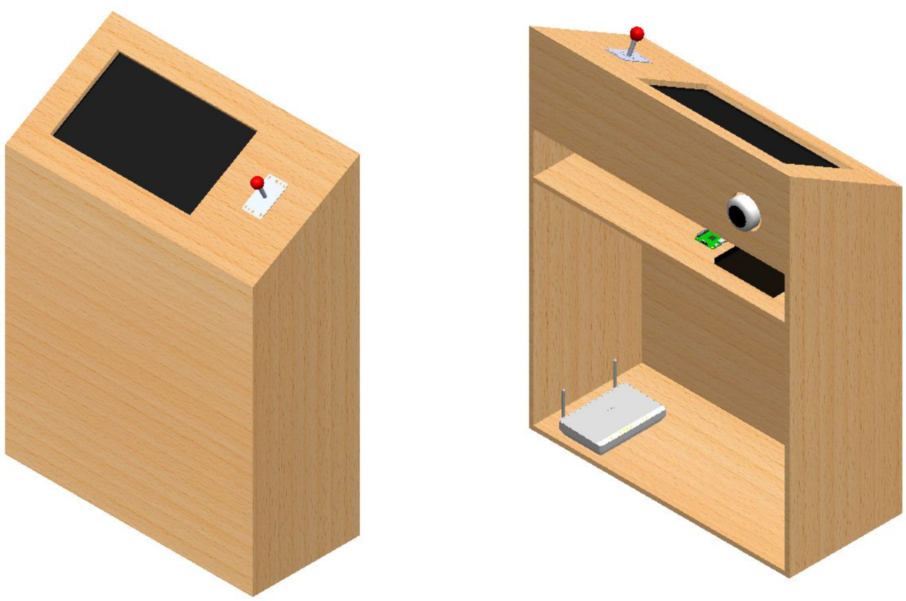


Fig. 5 - Isometric View of Cabinet with Monitor, Camera, Joystick, Router, Developer Kit, and Raspberry PI

1.4 Software Architecture Model

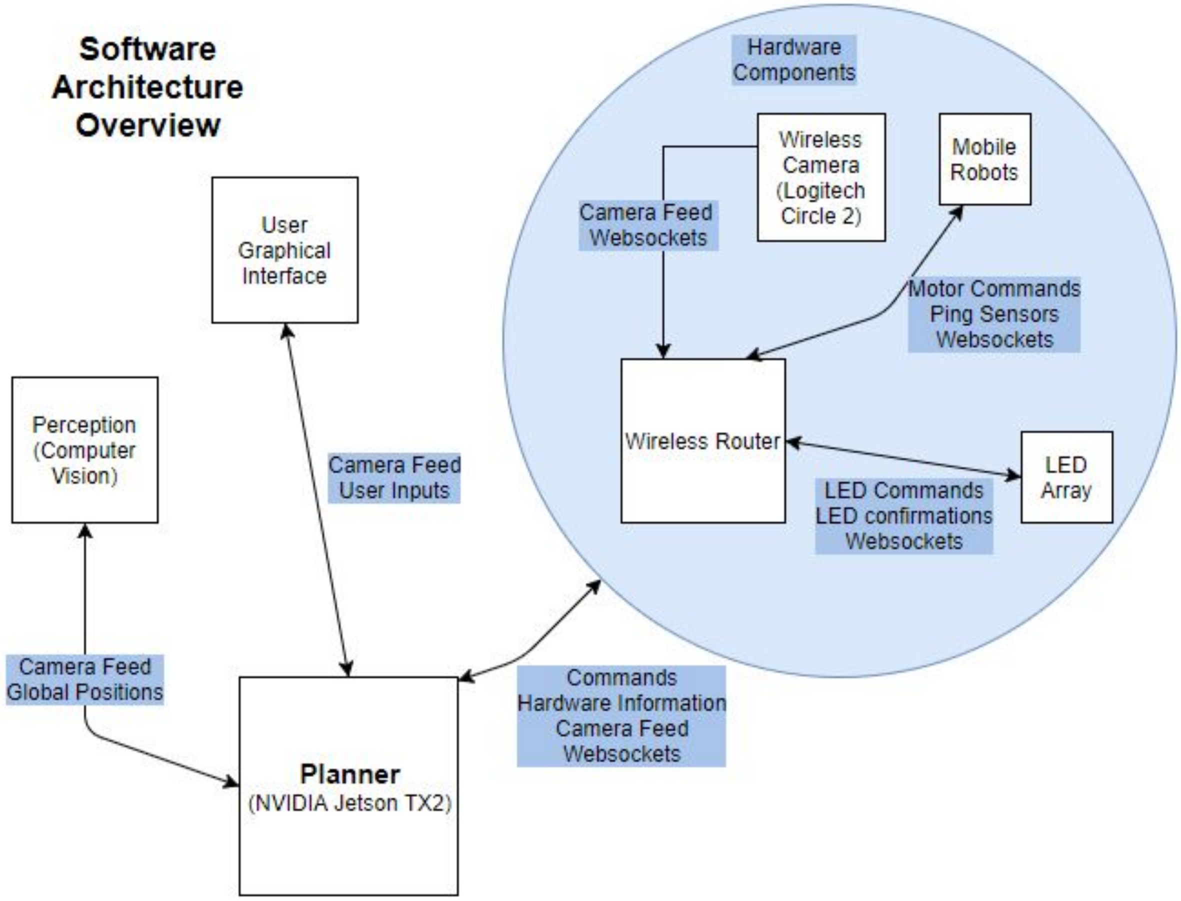


Fig. 6 - The general software architecture and connections between the software subsystems and hardware subsystems

9.5 Temporal Model

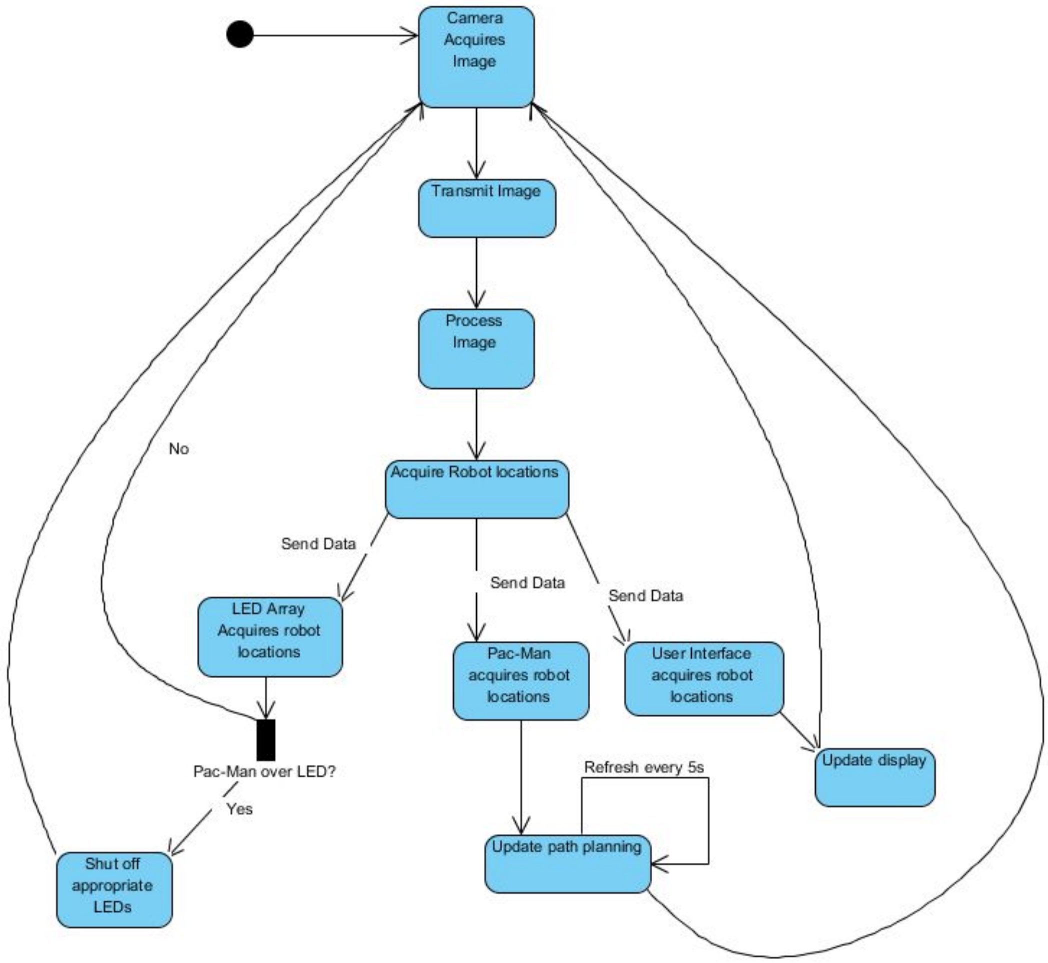


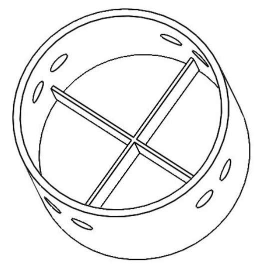
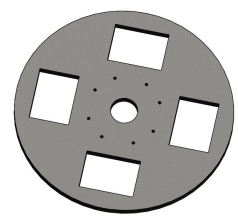
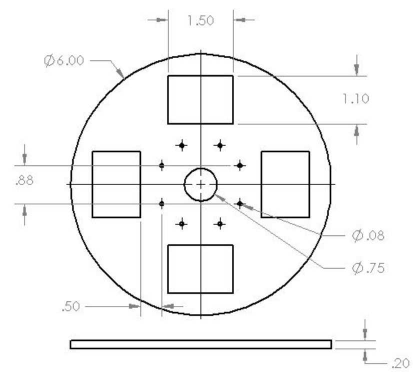
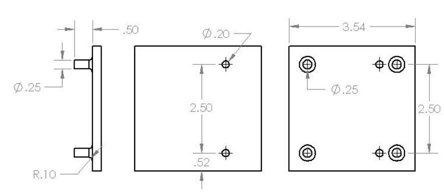
Fig. 7 Computer Vision State Diagram

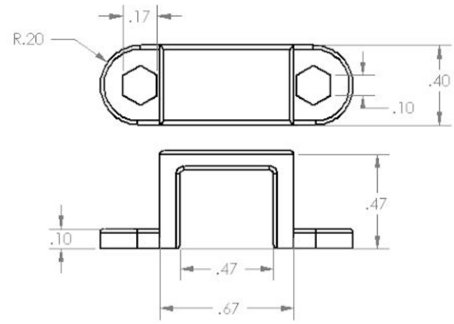
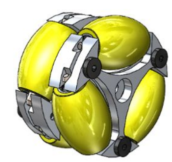
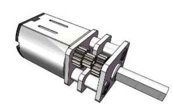
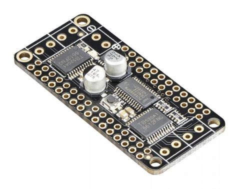
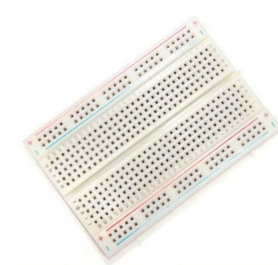
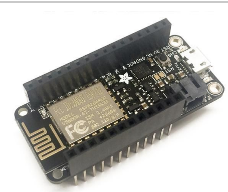
This model explains the various states that the perception subsystem will follow in the process of tracking the robots. Once the images are acquired and transmitted through the network to the planner (Jetson TX2), the computer vision algorithm will acquire the locations of the robots and use that data for various other processes. In context of the system, this model shows the main loop that keeps the game going.

10. Assembly and Installation Instructions

10.1 Components

Table 1 - Pac-Man Components Specification

ID	Component	Count	Specification	Image
P-1	Casing	1	3D printed with clear PLA filament	
P-2	Base	1	3D printed with black PLA filament	 
P-3	Support	1	3D printed with black PLA filament	 

P-4	Motor Holder	1	3D printed with black PLA filament	 
P-5	Omni-wheel	4	1.5" diameter	
P-6	Micro Gearmotor	4	150:1 gear ratio, 1.6A stall @ 6V	
P-7	Motor Controller	1	DC motor + stepper featherwing	
P-8	Breadboard	1	ABS, 2 power lanes	
P-9	Feather HUZZAH	1	Built-in wifi 802.11 b/g/n	

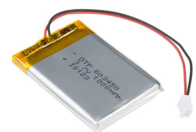
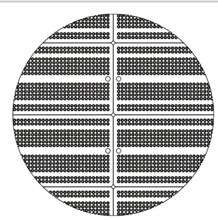
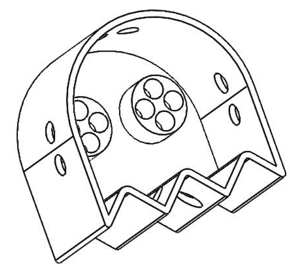
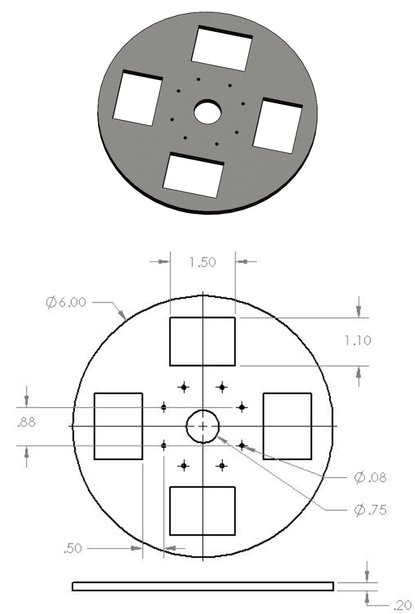
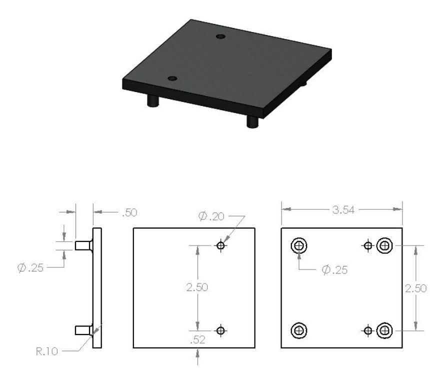
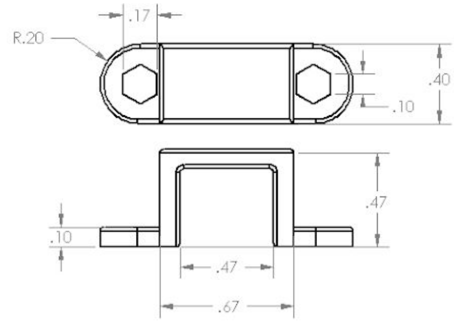
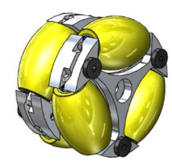
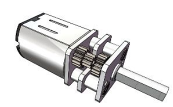
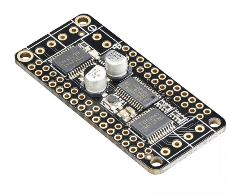
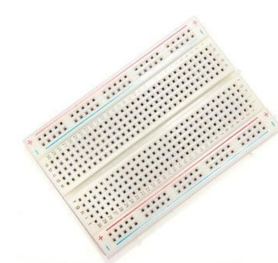
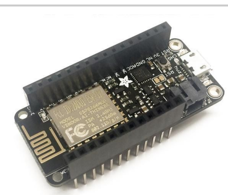
P-10	ping sensor	4	ranging distance 2cm-500cm, resolution 0.3 cm	
P-11	large battery	1	6V, 2200mAh	
P-12	small battery	1	3.7V, 500mAh	
P-13	Output I/O Expander Breakout	1	16-channel GPIO expander, operate in 1.2-3.6V	
P-14	Yellow LED	12	Diameter 5mm, 60 degree viewing angle, forward voltage 3.0-3.4V @ 20mA	
P-15	Solderable breadboard	1	Cut into circular shape to fit into the casing	

Table 2 - Ghost Components Specification

ID	Component	Count	Specification	Image
G-1	Casing	1	3D printed with clear PLA filament	
G-2	Base	1	3D printed with black PLA filament	
G-3	Support	1	3D printed with black PLA filament	

G-4	Motor Holder	1	3D printed with black PLA filament	 
G-5	Omni-wheel	4	1.5" diameter	
G-6	Micro Gearmotor	4	150:1 gear ratio, 1.6A stall @ 6V	
G-7	Motor Controller	1	DC motor + stepper featherwing	
G-8	Breadboard	1	ABS, 2 power lanes	
G-9	Feather HUZZAH	1	Built-in wifi 802.11 b/g/n	

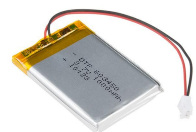
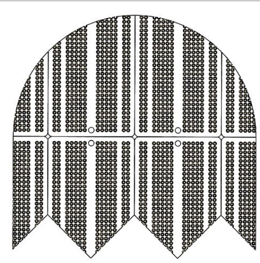
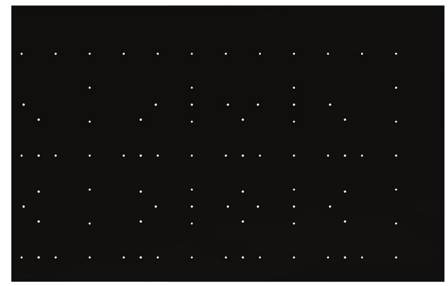
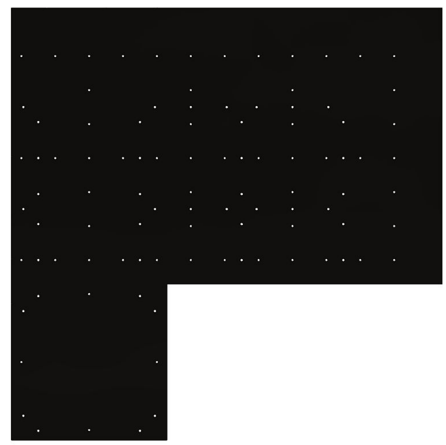
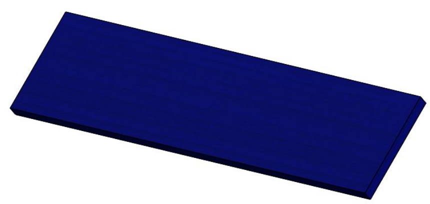
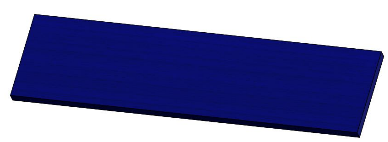
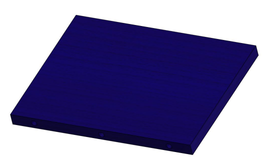
G-10	ping sensor	4	ranging distance 2cm-500cm, resolution 0.3 cm	
G-11	large battery	1	6V, 2200mAh	
G-12	small battery	1	3.7V, 500mAh	
G-13	Output I/O Expander Breakout	1	16-channel GPIO expander, operate in 1.2-3.6V	
G-14	Cyan LED	11	Diameter 5mm, 15 degree viewing angle, forward voltage 3.2V, reverse voltage 5V	
G-15	Blue LED	8	Diameter 5mm, 15 degree viewing angle, forward voltage 3.2V, reverse voltage 5V	
G-16	Solderable breadboard	1	Cut into circular shape to fit into the casing	

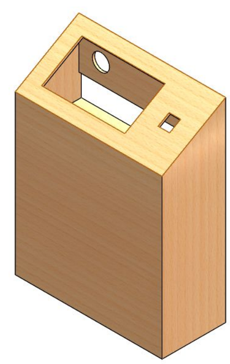
Table 3 - Board Components Specification

ID	Component	Count	Specification	Image
B-1	Base	3	Plywood, black paint	
B-2	Center base	1	Plywood, black paint	
B-3	Border long	4	Plywood, blue paint	
B-4	Border short	4	Plywood, blue paint	
B-5	Walls	24	Plywood, blue paint	

B-6	LED	240	Diameter 5mm, 10,000mcd brightness, forward voltage 3.2-3.4V, max current 20mA	
B-7	Feather HUZZAH	1	Built-in wifi 802.11 b/g/n	
B-8	Output I/O Expander Breakout	1	16-channel GPIO expander, operate in 1.2-3.6V	
B-9	Dowel pin	72	wooden, 1.5" long, 0.25" diameter	

Table 4 - User-Interface Components Specification

ID	Component	Count	Specification	Image
U-1	Joystick	1	8-way joystick	
U-2	Touchscreen Display	1	21.5-inch widescreen display with Height (extended): 14.60 in, Width: 20.57 in, Depth: 8.10 in Touch technology: Projected capacitive system Sensor stack thickness: 1.1 mm Cover glass thickness: 1.1 +/- 0.2 mm Touch method: Fingers and thin gloves Touch point: 10 touch-points Response time: 25 ms Supporting OS: Windows 8 Certified	

U-3	Logitech Circle 2 Camera	1	180 viewing angle	
U-4	Router	1	Smooth HD streaming and online gaming with high-performance AC1200 Wi-Fi over 5GHz (867Mbps) and 2.4GHz (300Mbps) bands	
U-5	Raspberry PI	1	Quad Core Broadcom BCM2837 64-bit ARMv8 processor, 1G RAM	
U-6	Jetson TX2 Developer Kit	1	2 Denver 64-bit CPUs + Quad-Core A57 Complex, 8 GB L128 bit DDR4 Memory, 32 GB eMMC 5.1 Flash Storage, Connectivity to 802.11ac Wi-Fi and Bluetooth-Enabled Devices	
U-7	Cabinet	1	Plywood	

10.2 Subsystem Dimensions

All units are in inches.

10.2.1 Pac-Man and Ghost Dimensions

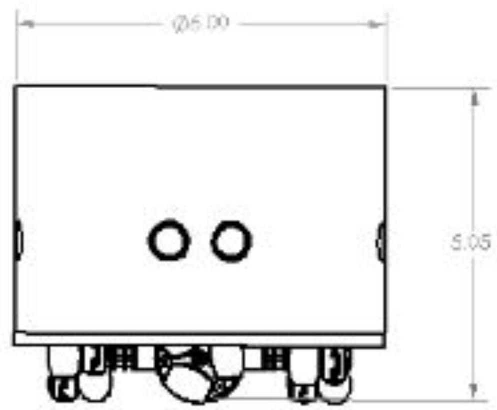


Fig. 8 - Pac-Man/Ghost Overall Dimensions

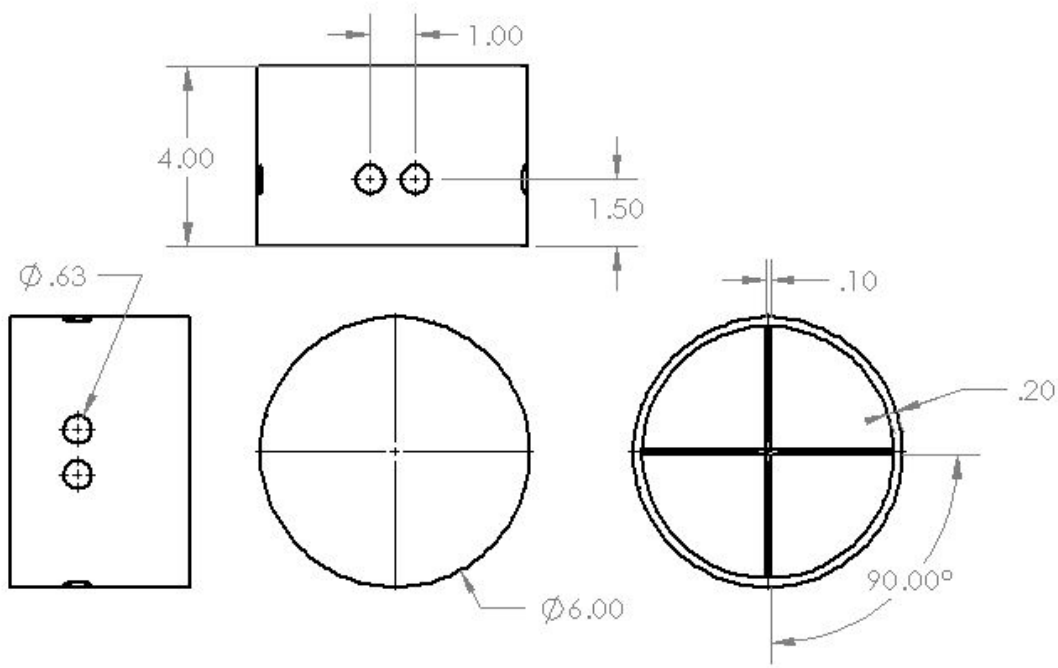


Fig. 9.1 - Pac-Man Casing Dimensions

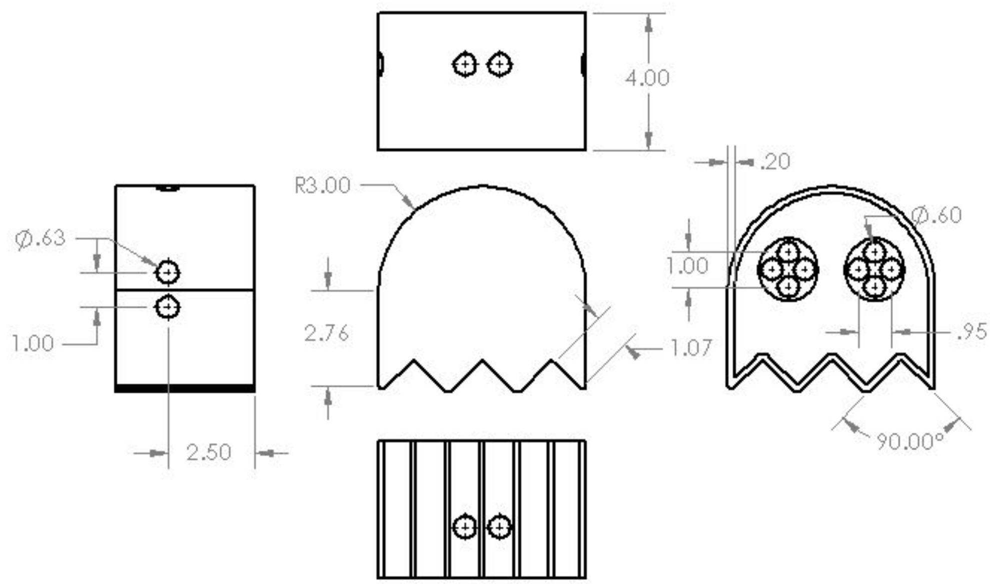
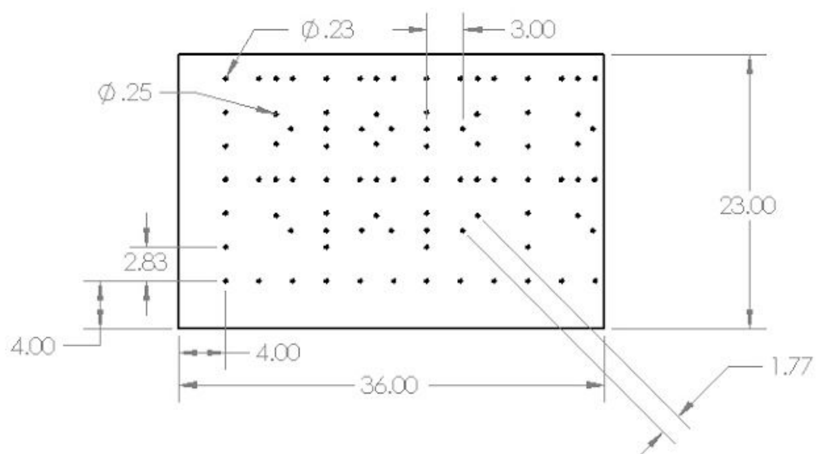


Fig. 9.2 - Ghost Casing Dimensions

10.2.2 Board Dimensions



(½" thick)
Fig. 10 - Base Dimensions

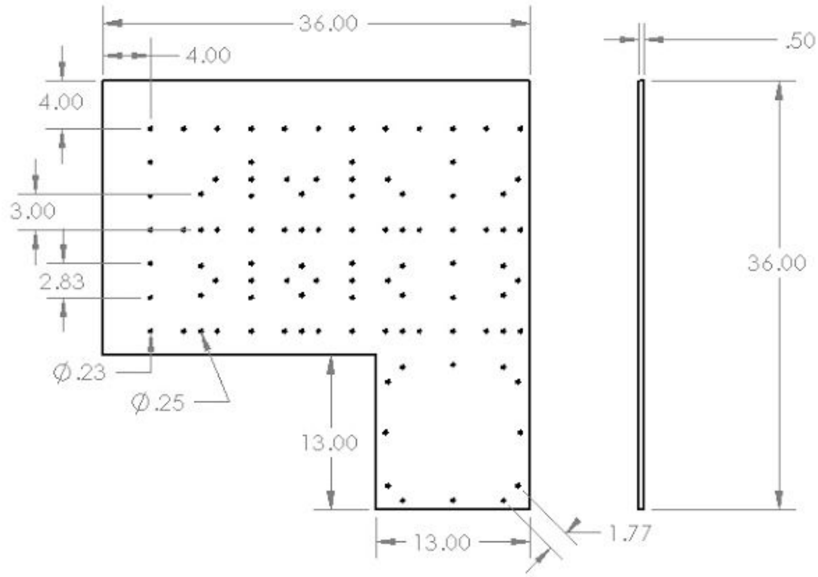
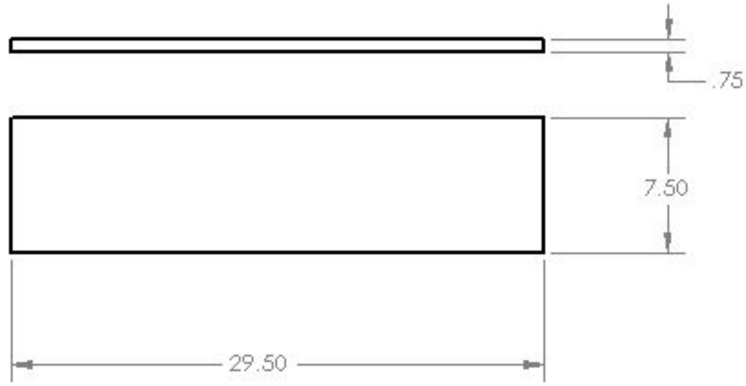
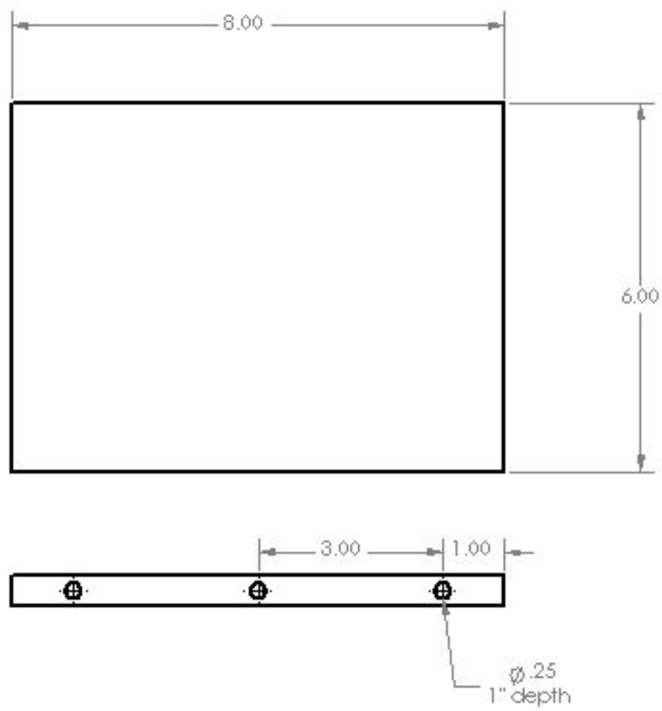


Fig. 11 - Base Center Dimensions

(¾" thick)
Fig. 12 - Border Long Dimensions



(¾” thick)
Fig. 13 - Border Short Dimensions



(½” thick)
Fig. 14 - Wall Dimensions

10.2.3 Cabinet Dimensions

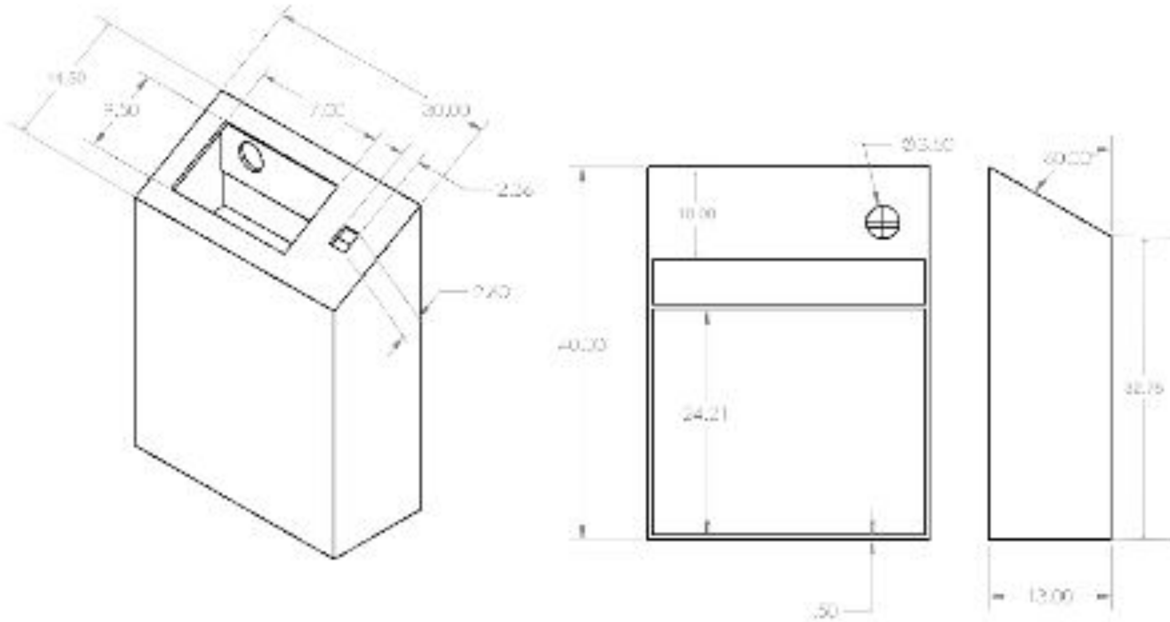


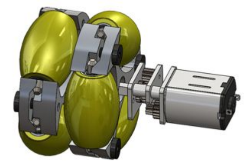
Fig. 15 - Cabinet Dimensions

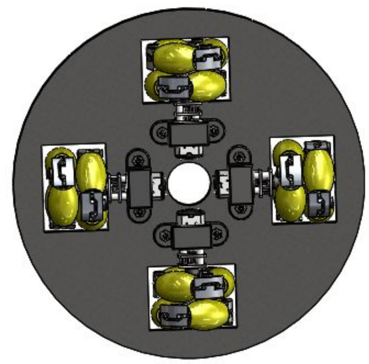
10.3 Assembly Instructions

10.3.1 Pac-Man and Ghost

<u>Platform:</u>

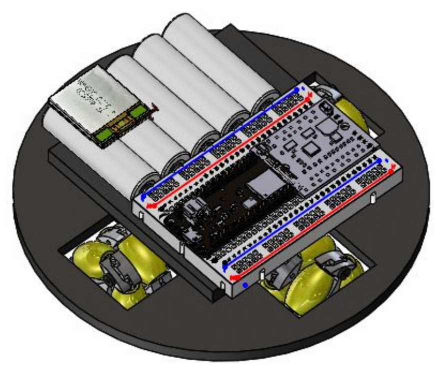
1. Attach each of the four motors (P-6, G-6) to an omni-wheel (P-5, G-5).
2. Position each of the four omni-wheels (P-5, G-5) inside a rectangular slot on the base (P-2, G-2) with the flat side of the motor flushed onto the base.
3. Place a motor holder (P-4, G-4) over each motor (P-6, G-6) and screw it onto the base (P-2, G-2).

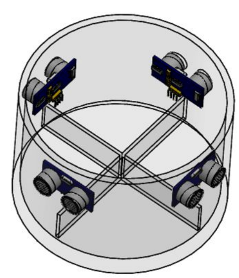


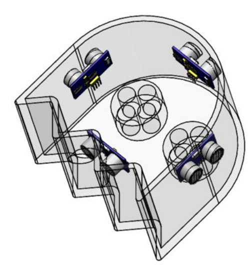


<u>Circuit:</u>

1. Attach the support (P-3, G-3) onto the platform on the side without the motors using an adhesive.
2. Attach the breadboard (P-8, G-8) onto the support (P-3, G-3) using double-sided tape.
3. Connect the Feather HUZZAH (P-9, G-9) to the breadboard (P-8, G-8).
4. Wire the motors (P-6, G-6) to the motor controller (P-7, G-7).
5. Connect the motor controllers (P-7, G-7) to the output pins on the Feather HUZZAH (P-9, G-9) by passing the wires through the center hole on the platform.
6. Connect the large battery (P-11, G-11) and small battery (P-12, G-12) to the breadboard (P-8, G-8).
7. Secure the batteries onto the support (P-3, G-3) next to the breadboard (P-8, G-8) using a zip tie.







Cover:

1. Wire the four ping sensors (P-10, G-10) to the input pins on the Feather HUZZAH (P-9, G-9).
2. Fit the four ping sensors (P-10, G-10) into the holes on the translucent Pac-Man/Ghost casing (P-1, G-1).
3. Wire the I/O expander (P-13, G-13) to the Feather HUZZAH (P-9, G-9).
4. Cut the solderable breadboard (P-15, G-16) to the shape of Pac-Man and the ghost, one for each.
5. For Pac-Man: Solder 12 yellow LEDs (P-14) onto the breadboard (P-15) in the configuration shown in Fig. 16.
   For the Ghost: Solder 4 blue LEDs (G-15) and 11 cyan LEDs (G-14) onto the breadboard (G-16) in the configuration shown in Fig. 16.
6. Wire the LEDs to the I/O expander (P-13, G-13) according to the circuit shown in Fig. 16.
7. Position the breadboard such that the LEDs line up with the casing (P-1, G-1).
8. Screw in the casing (P-1, G-1) to the platform.

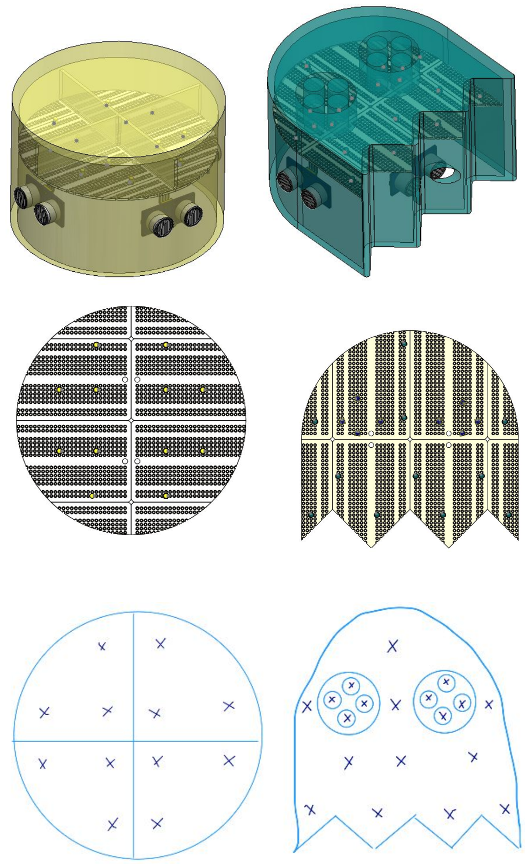


Fig. 16 - Breadboard Configurations

Software:

1. Program the Feather HUZZAH (G-9) to follow the flow chart outlined in Fig. 17. The ghost should start out in an idle state while it waits for the game to start. Once the game starts, all of the LEDs corresponding to the standard ghostface should be turned on. After that, the ghost should follow the user input, including changing the eye LEDs depending on the specific movement.
2. Program the Feather HUZZAH (P-9) to follow the flow chart outlined in Fig. 18. Pac-Man should start out in an idle state while it waits for the game to start. Once the game starts, all of the LEDs corresponding to the standard Pac-Man face should be turned on. Afterwards, after every 6 inches, the Pac-Man face should close to stimulate Pac-Man eating. If Pac-Man dies, the death sequence should start.

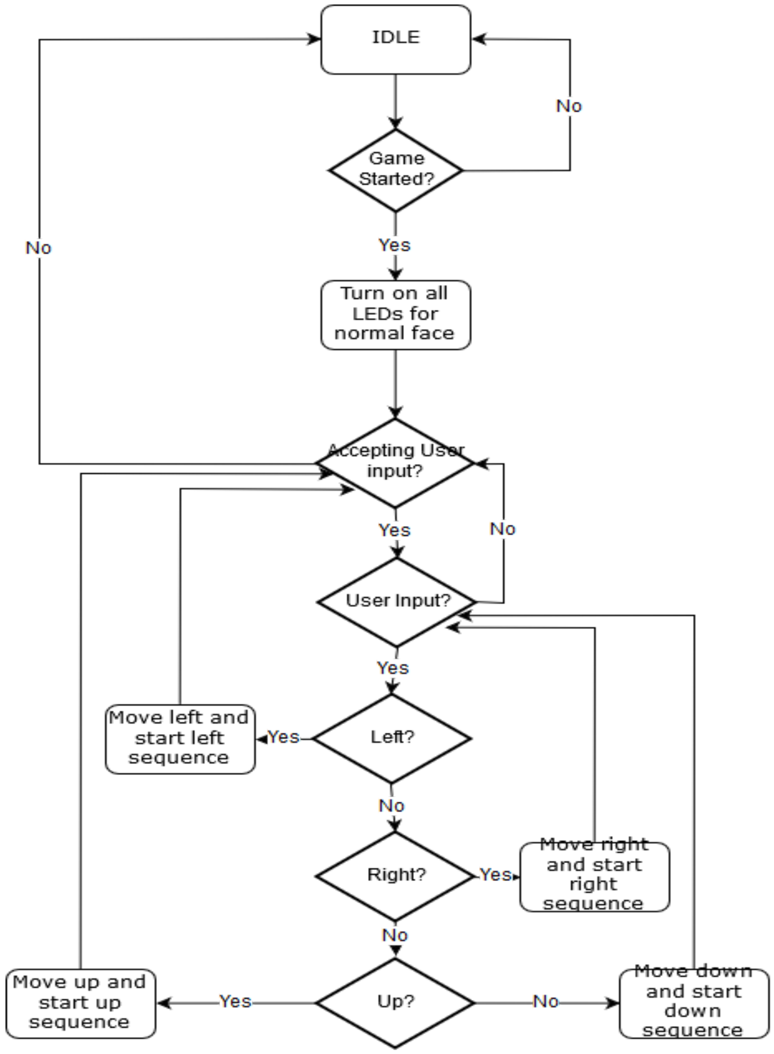


Fig. 17 - Ghost Software Schematic

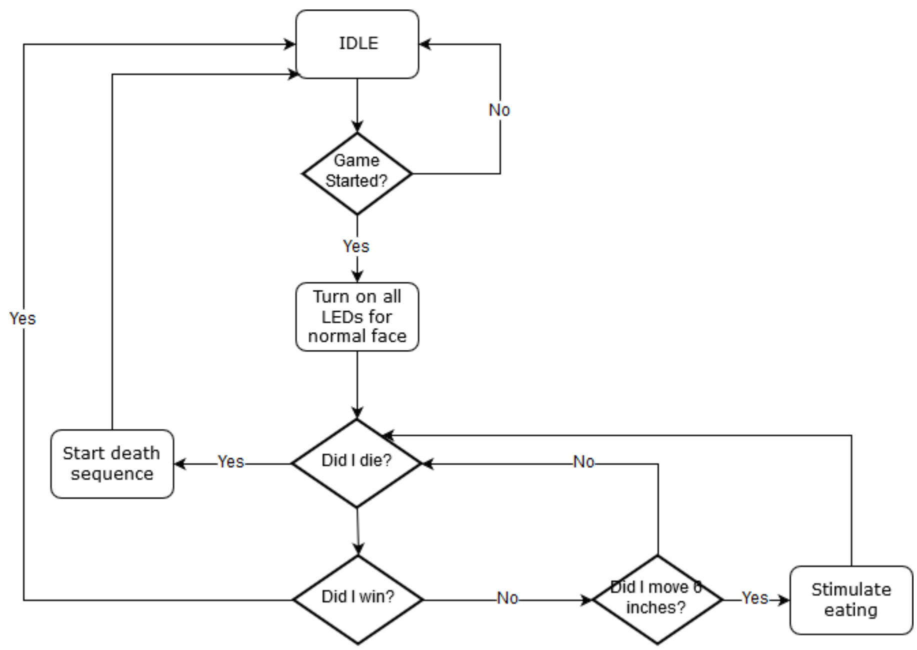


Fig. 18 - Pac-Man Software Schematic

10.3.2 LED Board

Base Plate:

1. Cut out 3 base plates (B-1) out of ½ inch thick plywood.
2. Cut out 1 base center plate (B-2) out of ½ inch thick plywood.
3. Drill 0.23" holes for the LEDs according to the layout in Fig. 10 and 11.
4. Drill 0.25" holes for the dowel pins according to the layout in Fig. 10 and 11.
5. Paint all four base plates black.

Border:

6. Cut out 4 border long pieces (B-3) out of ¾ inch thick plywood.
7. Cut out 4 border short pieces (B-4) out of ¾ inch thick plywood.
8. Paint all border pieces blue.

Walls:

9. Cut out 24 walls (B-5) out of ½ inch thick plywood.
10. Drill three 0.25" holes according to the layout in Fig. 14.
11. Paint the walls blue.

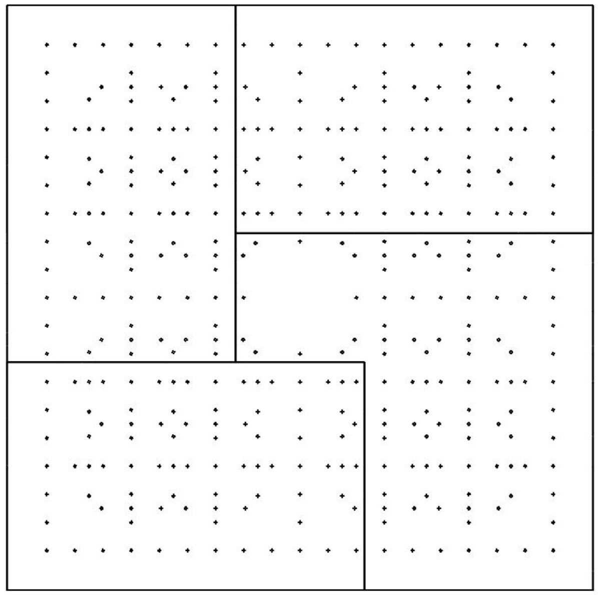


Fig. 19 – Meshed Base Plates

<u>LEDs:</u>

1. Solder 240 white LEDs (B-6) in the configuration shown in Fig. 19. Pay close attention to make sure the distances between each LED are accurate.
2. Wire the LEDs (B-6) to the I/O expander (B-8).
3. Wire the I/O expander (B-8) breakout to the Feather HUZZAH (B-7).

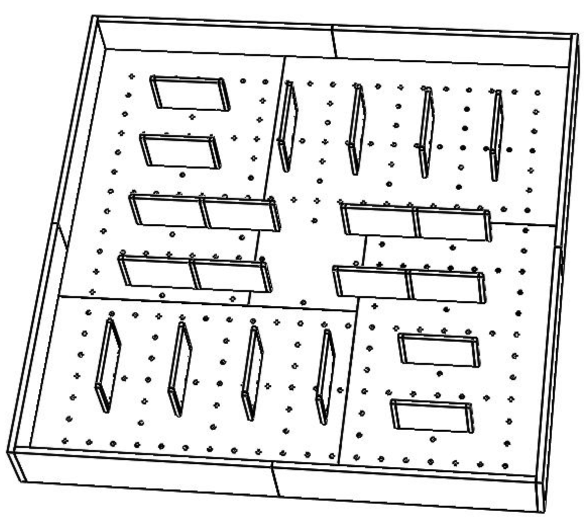


Fig. 20 - Assembled Board

Board:

1. Mesh the base plates together as seen in Fig. 19 to form a complete square.
2. Place two long borders on opposite sides of the square board base.
3. Place two short borders on the other opposite sides of the square board base.
4. After making sure that the borders fully encloses the square board, nail the borders to the base plate, leaving 1 inch space under the base plates.
5. Insert each of the wired LEDs into the holes on the base plates (B-1, B-2) from underneath.
6. Secure the LEDs with tape.
7. Insert the walls in customized configuration.

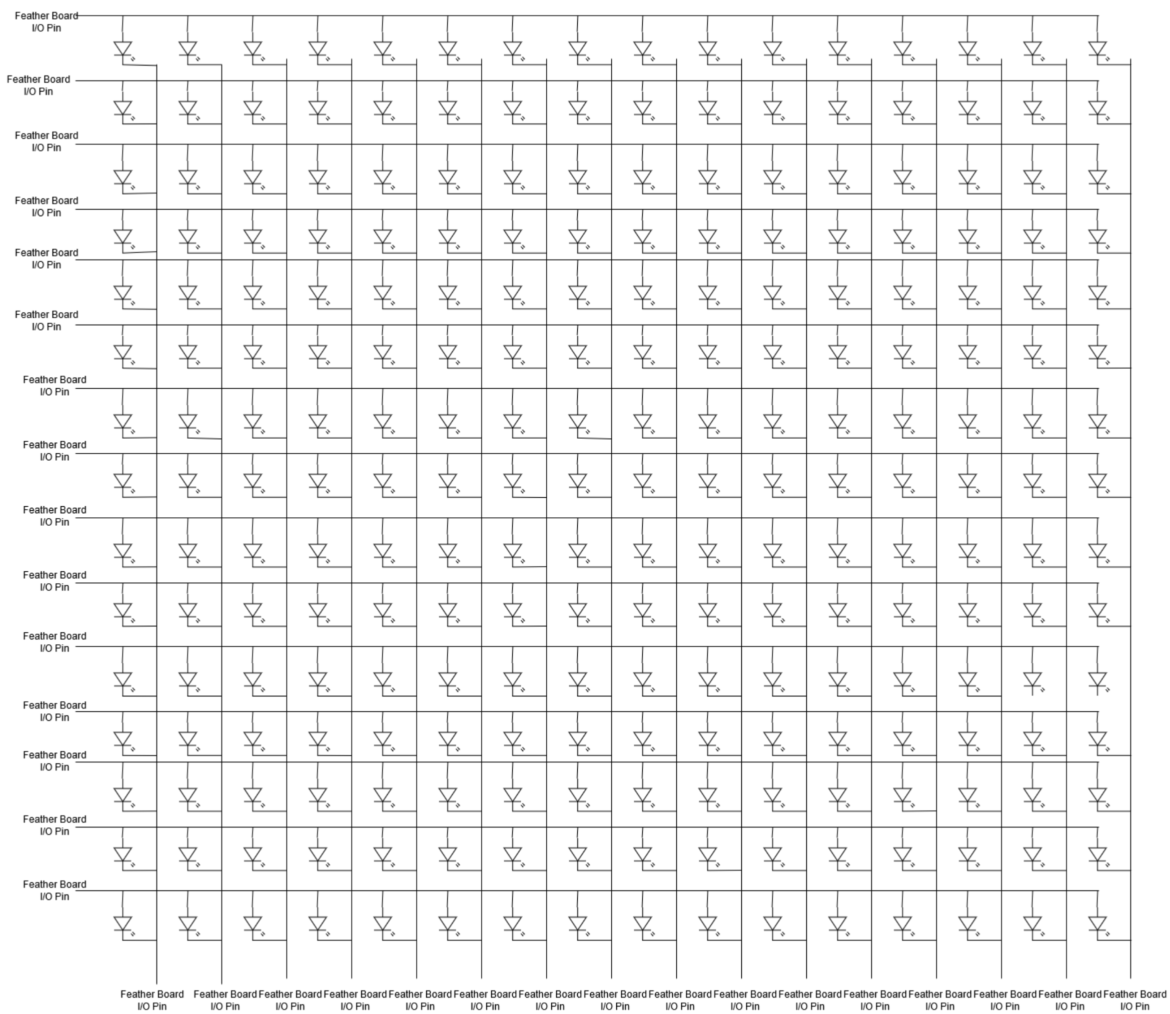


Fig. 21 - LED Configuration

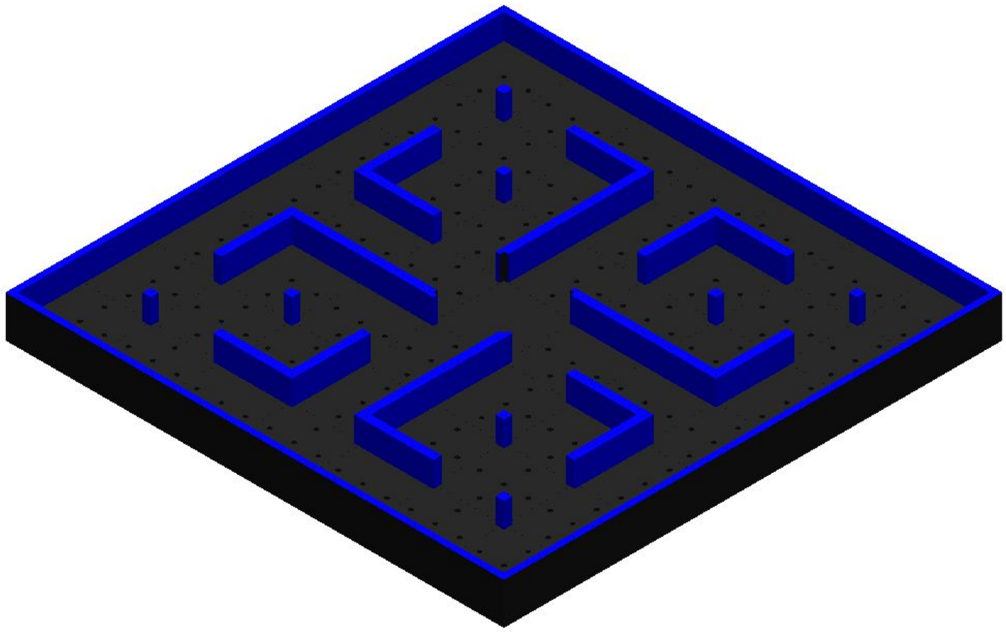


Fig. 22 - Board Layout

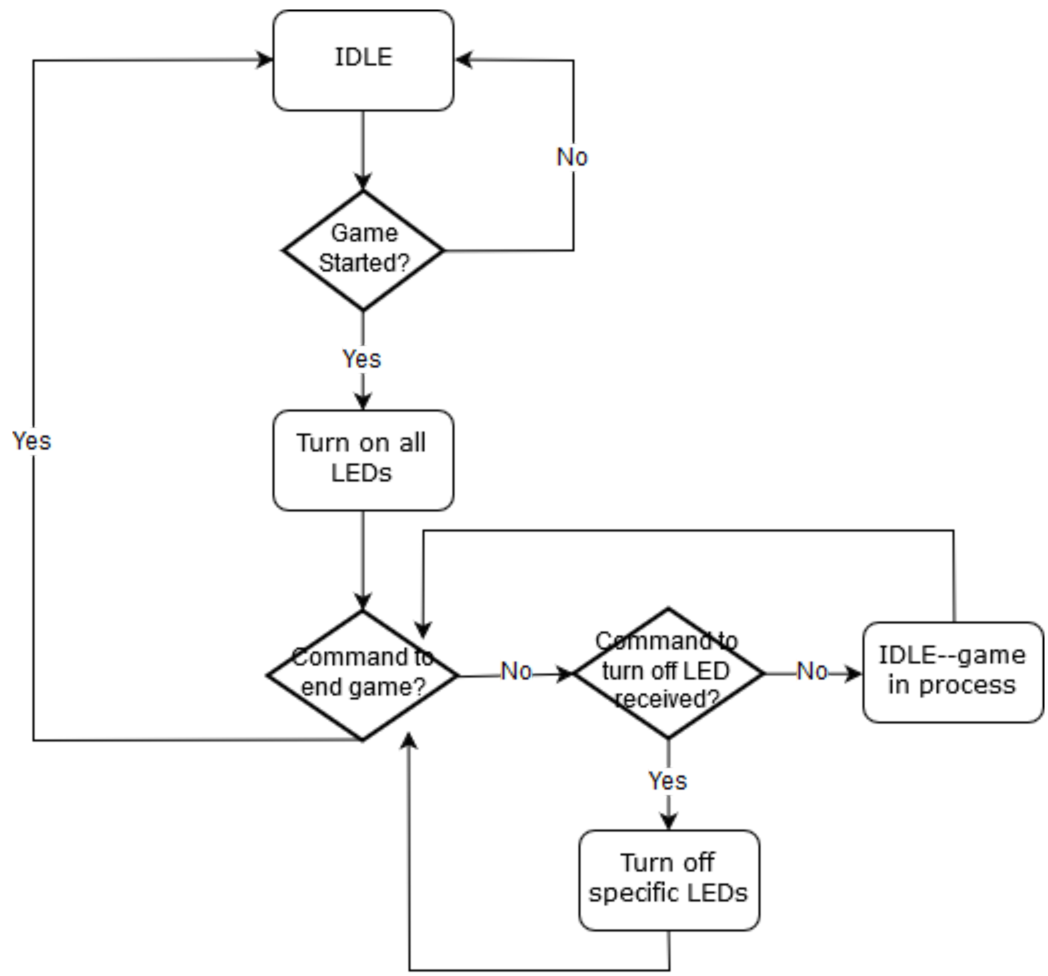


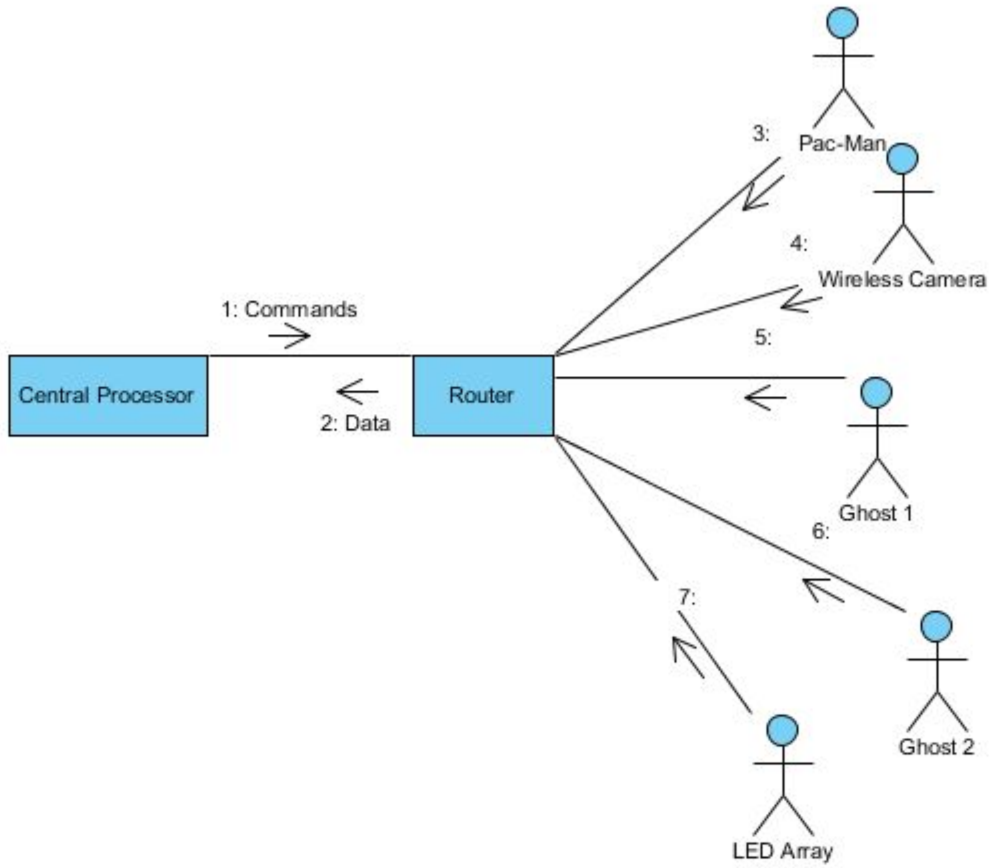
Fig. 23 - LED Maze Software Schematic

<u>Software:</u>

1. Program the Feather HUZZAH (B-7) for the LED maze to follow the flow chart outlined in Fig. 23. The LED board should start out in an idle state until the game starts. When the game starts, all of the LEDs should be turned on. Afterwhich, while the game is in progress, the LEDs near Pac-Man should be turned off to stimulate Pac-Man eating the dots.

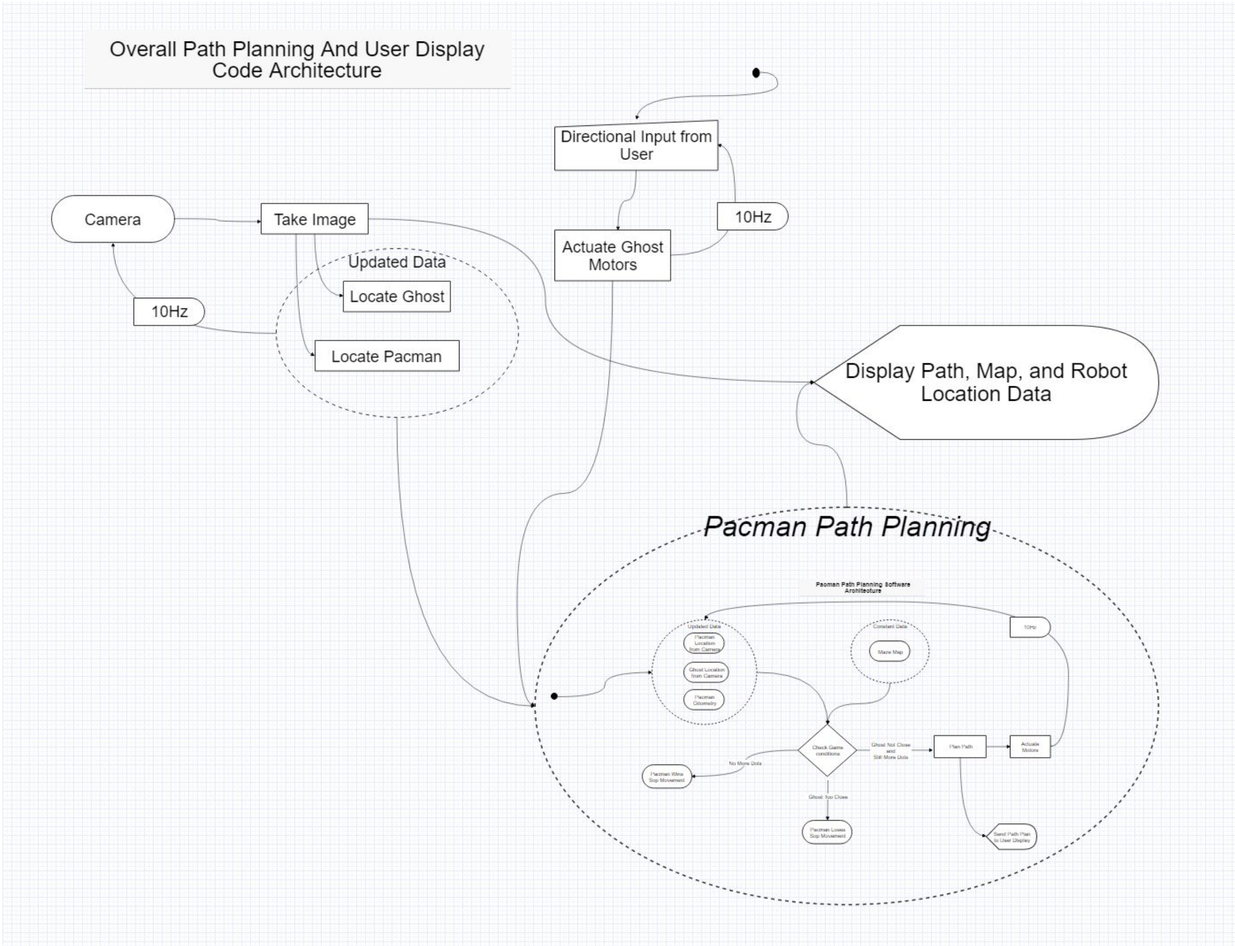
10.3.3 Wireless Communications

The router mentioned in the Bill of Materials is needed to set up a local server to handle all the connections and traffic. A WebSockets library will be used to handle the data, connections, and transmission in a unified way. The figure below shows the necessary connections.



10.3.4 Computer Vision

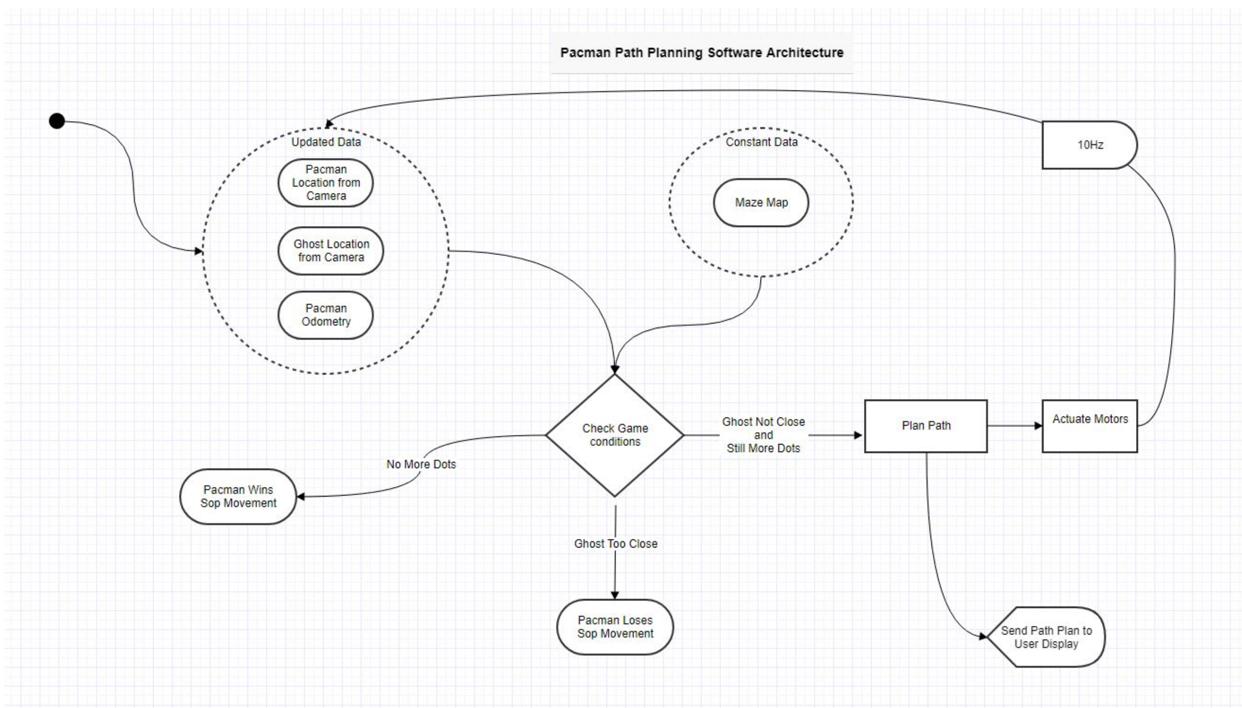
The computer vision subsystem will mostly be used to keep track of the robots locations. However, it's also necessary to keep track of changes in the maze and when Pac-Man eats a dot. For robot tracking, threshold the image provided by the camera to the known colors on the robots, rasterize the image to gather the biggest blobs of those colors, then find the centroids of those blobs. When the centroid matches a known dot position, send a command to the LED array to shut off that LED and mark it as eaten. For maze tracking, keep a logical array initialized to 0, then fill in 1s where there are walls. Detect walls by thresholding the image for the color of the LEDs on the walls. Once the array is filled with the map, use flood-fill to detect any blockages, and alert the user of such blockages in the maze. Below is a figure showing a basic architecture for the computer vision software.



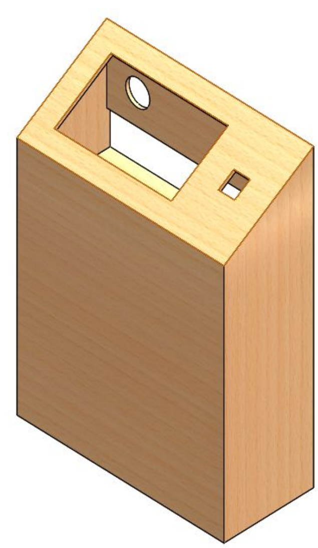
Overall Path Planning And User Display
Code Architecture
Directional Input from User
Camera
Take Image
10Hz
Actuate Ghost Motors
Updated Data
Locate Ghost
10Hz
Locate Pacman
Display Path, Map, and Robot Location Data
Pacman Path Planning
Pacman Path Planning Software Architecture
Updated Data
Pacman Location from Camera
Ghost Location from Camera
Pacman Odometry
Constant Data
Maze Map
10Hz
Check Game conditions
Ghost Not Close and Still More Dots
Plan Path
Actuate Motors
No More Dots
Pacman Wins Sop Movement
Ghost Too Close
Pacman Loses Sop Movement
Send Path Plan to User Display

10.3.5 Path Planning

Use a path planning algorithm that maximizes the Pac-Mans score. It has to simultaneously avoid the Ghost robots and collect the dots on the map. It will do this by initially using a basic greedy algorithm that checks which of the four possible moves (up, left, down, or right) move Pac-Man the furthest away from the Ghost robot. After a few moves have caused the Pac-Man to collect some dots and clear some of the map, Pac-Man will start using a planning method based on Anytime Repairing A* (ARA*). This planning method will allow the Jetson TX2 board to find the best possible plan in under a certain time (<300 ms) to the best nearest dot to Pac-Man. The edge costs and heuristics will be based on proximity to the Ghost robot. Below is a basic path planning architecture.



10.3.6 User Interface Cabinet

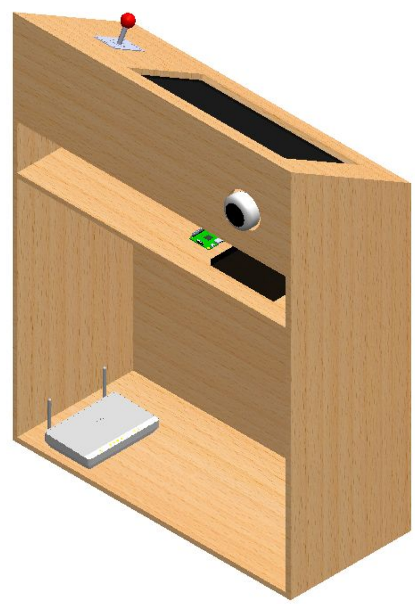


Cabinet:

1. Using ½ inch plywood, cut out a 10in x 30in rectangular back board.
2. Drill 4.5 inch diameter hole near one corner of the back board.
3. Using ½ inch plywood, cut out a 14.5in x 30in rectangular top board.
4. Cut out 9.5in x 17in rectangle hole to the left side of the top board for the display.
5. Cut out 2.6in by 2.36in rectangle to the right side of the top board for the joystick.
6. Using ½ inch plywood, cut out two 13in x 30in rectangular shelf boards.
7. Using ½ inch plywood, cut out a 32.75in x 30in rectangular front board.

8. Using ½ inch plywood, cut out two 13in x 40in side boards.
9. Draw a 60 degree slanted line from one corner of the two 13in x 40in side boards as shown in Fig. 15.
10. Cut along the slanted line.
11. Nail the side boards to the front board.
12. Nail one of the shelf boards to the flat side of the side boards.
13. Nail the remaining shelf board 24.21 inch above the other shelf board.
14. Nail the top board to the slanted side of the side boards.
15. Nail the back board the top board on the opposite side of the front board.

Hardware:



1. Position the joystick (U-1) in the small square cutout of the top of the cabinet (U-7) and screw it in.
2. Connect the joystick (U-1) to Raspberry PI (U-5).
3. Angle the neck of the touch screen display (U-2) so that the whole screen aligns with the large square cutout on the top of the cabinet (U-7) and the bottom of the touch screen base is parallel to the inner wall of the cabinet.
4. Attach the base of the touchscreen display (U-2) to the cabinet (U-7) wall using double-sided tape.
5. Place the Jetson TX2 (U-6) somewhere inside the cabinet (U-7).
6. Connect the display (U-2) to Jetson TX2 (U-6) using a displayport cable.
7. Secure the camera (U-3) onto the circular cutout at the back of the cabinet (U-7) at 28 angle from vertical.
8. Connect the camera (U-3) to Jetson TX2 (U-6).
9. Place the router (U-4) somewhere inside the cabinet (U-7).

Software:

1. Program the Raspberry Pi (U-5) to follow the flow chart outlined in Fig. 24. The user interface should always be ready for a new game. Once a game starts, it should allow the user to pick a user level, and then wait for the user to start the game. Once the game is started, the feed from the overhead camera should be displayed as a digitized Pac-Man game. When the game is over, it should display the results of the game, either the user lost or they won, and then display path planning information appropriate for the user level chosen.

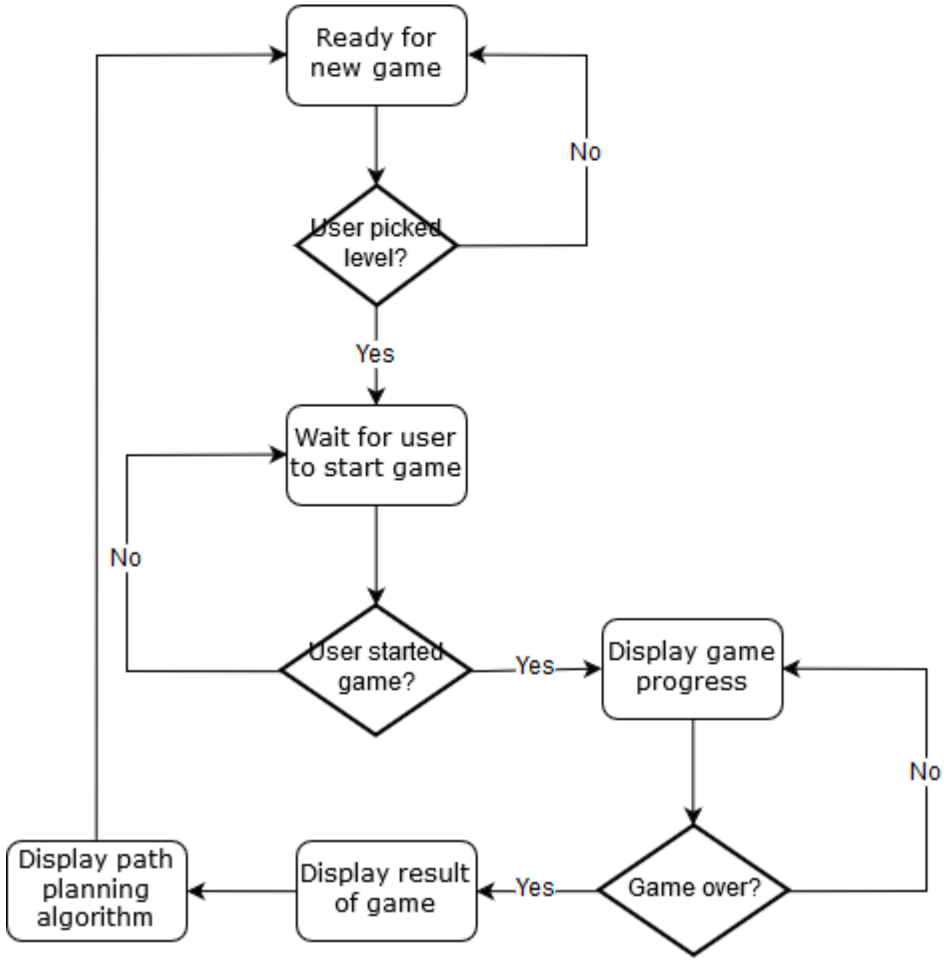


Fig. 24 - User Interface Software Schematic

10.4 Installation Instructions

The system will be set-up in a museum setting where it can be displayed. After assembling the system as shown above, the setup process for the system is as follows:

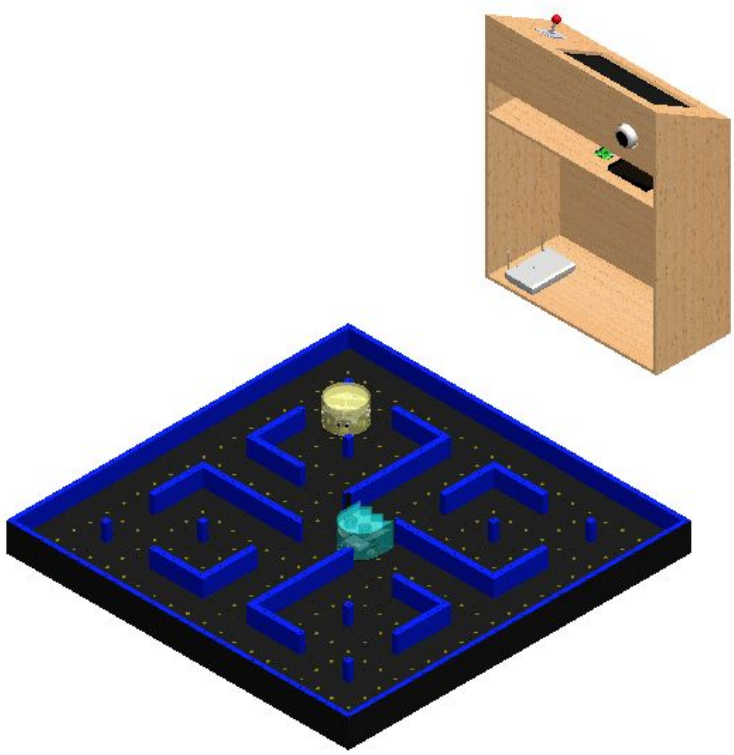
1. Place the maze or the outer box of the system on the floor. This will allow the system to be viewed from above comfortably for all age groups.

2. The Ghost robot should be placed at the center of the maze – at the location marked with a star. The Pac-Man robot can start at any location but it must be at least two lanes apart from the Ghost robot to prevent the game from being too easy and quick.

3. The screen must be connected to the system so that we can transfer the data from the Pac-Man robot onto the screen and use it to display the path-planning and localization aspects in order to facilitate the educational aspect of the game.

4. The screen and joystick components should be placed on a small table that faces the system to allow the player to view the ghost and Pac-Man robots.

5. The whole robotic system as a whole should be set-up near a couple of outlets so that the system can be charged and draw power during operation.



11. Bill of Materials

Vendor Name	Vendor URL	Item Description	Price	Quantity	Shipping	Purchase Justification
Pac-Man and Ghost						
Amazon	https://www.amazon.com/3DRAX-PLA-Filament-Accuracy-Spool-BLACK/dp/B07564QWCF/ref=sr_1_2_sspa?ie=UTF8&qid=1511337922&sr=8-2-spons&keywords=black+filaments&psc=1	Black PLA filament for 3D printing platform, support, and motor holder	14.97	1	Standard	Black PLA filament to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Amazon	https://www.amazon.com/Printing-Filament-1-75mm-Dimensional-Accuracy/dp/B06WP5CJPP/ref=sr_1_5?s=industrial&ie=UTF8&qid=1511338048&sr=1-5&keywords=translucent+pla+filaments	Translucent PLA filament for 3D printing Pac-Man and ghost casings	17.98	1	Standard	Translucent PLA filament to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Digi-Key	https://www.digikey.com/product-detail/en/ROB-12408/1568-1163-ND/5673749?WT.mc_id=IQ_7595_G_pla5673749&wt.srch=1&wt.medium=cpc&&gclid=Cj0KCQiA3dTQBRDnARIsAGKSflmKR8NSEERC9Dl97YfPWnMrFUpVnda5duQYrEYnaW-N1bK2xHO5TkMaAsIrEALw_wcB	Metal gear motors for the wheels	12.95	8	Standard	Metal gearmotors to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Amazon	https://www.amazon.com/UniHobby-1-5inch-mounting-couplings-Platforms/dp/B01M24F2FJ/ref=sr_1_4?ie=UTF8&qid=15	Omni-wheels for Pac-Man and ghost	59.99	2	Standard	Omni-wheels to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207

	11338227&sr=8-4&keywords=omni+wheels					
Polulu	https://www.pololu.com/product/2223	Large battery to power motors	15.15	2	Standard	Battery to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Amazon	https://www.amazon.com/WYPH-Ultrasonic-HC-SR04-Distance-Measuring/dp/B00P5W1JLM/ref=sr_1_9?ie=UTF8&qid=1510853358&sr=8-9&keywords=ultrasonic+sensor	WYPH Ultrasonic Module HC-SR04 Distance Measuring Ranging Sensor for Arduino Pack of 10pcs	14.99	1	Standard	Ping sensors to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Adafruit	https://www.adafruit.com/product/3055	FeatherWing to control motors, ping sensors, and LEDs	35	2	Standard	Raspberry PI to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
SparkFun	https://www.sparkfun.com/products/13601	I/O expander breakout	4.95	2	Standard	I/O expander breakout to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Adafruit	https://www.adafruit.com/product/2927	Motor controller to control the motors	19.95	2	Standard	Motor controller to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Adafruit	https://www.adafruit.com/product/1898	Breadboard to power featherwing	1.85	2	Standard	Breadboard to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207

Adafruit	https://www.adafruit.com/product/1578	Small battery to power featherwing	7.95	2	Standard	Lithium battery to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Amazon	https://www.amazon.com/dp/B0778G64QZ/ref=asc_df_B0778G64QZ5296686/?tag=hyprod-20&creative=395033&creativeASIN=B0778G64QZ&linkCode=df0&hvadid=228696667650&hvpos=1o4&hvnetw=g&hvrand=3078550231463747541&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9005929&hvtargid=pla-386019820774	Solderable breadboard	9.68	2	Standard	Solderable breadboard to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Adafruit	https://www.adafruit.com/product/2700	Yellow LED	4.95	1	Standard	Yellow LEDs to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
superbrightleds.com	https://www.superbrightleds.com/moreinfo/through-hole/5mm-aqua-led-505nm-t1-34-through-hole-led-w-15-degree-viewing-angle/4044/	Cyan LED	0.39	25	Standard	Cyan LEDs to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
SparkFun	https://www.sparkfun.com/products/9662	Blue LED	7.95	1	Standard	Blue LEDs to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Amazon	https://www.amazon	Black spray paint	5.81	1	Standard	Black spray paint to be part of a robot being built by

	.com/Krylon-51601-Interior-Exterior-Decorator/dp/B0009XB3VI/ref=sr_1_3?ie=UTF8&qid=1513045793&sr=8-3&keywords=black+spray+paint	for the board				Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Amazon	https://www.amazon.com/Krylon-51910-Interior-Exterior-Decorator/dp/B000RZJS6K/ref=sr_1_2?s=automotive&ie=UTF8&qid=1513045812&sr=1-2&keywords=blue+spray+paint	Blue spray paint for the board	11.44	1	Standard	Blue spray paint to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
LED Board						
Adafruit	https://www.adafruit.com/product/754	Super bright 5mm white LED (25 pack)	6.95	8	Standard	LEDs to be part of the Pac-Man maze being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie Domingo. Item to reside in NSH 1207
Adafruit	https://www.sparkfun.com/products/13601	I/O expander breakout	4.95	2	Standard	I/O expander breakout to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie Domingo. Item to reside in NSH 1207
Adafruit	https://www.adafruit.com/product/3055	Raspberry Pi 3	35	1	Standard	Raspberry Pi used to control the LED maze being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie Domingo. Item to reside in NSH 1207
Amazon	https://www.amazon.com/dp/B002BWOS9Y/ref=sxbs_sxwds-stvp_1?pf_rd_m=ATVPDKIKX0DER&pf_rd_p=3341940462&pd_rd_wg=tVun2&pf_rd_r=7TC6DF49QJKQCKVQZPFE&pf_rd_s=desktop-sx-bottom-slot&pf_rd_t=301	Rust-Oleum 249854 Painter's Touch Multi Purpose Spray Paint, 12-Ounce, Satin Midnight Blue	11.78	2	Standard	Spray paint for the LED maze being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie

	&pd_rd_i=B002BWOS9Y&pd_rd_w=wiqbR&pf_rd_i=matte%20blue%20spray%20paint&pd_rd_r=2f32b61b-82da-4443-8adf-84a6ac418db2&ie=UTF8&qid=1511208762&sr=1					Domingo. Item to reside in NSH 1207
Amazon	https://www.amazon.com/Rust-Oleum-285093-Stops-Protective-Enamel/dp/B00SVWJ92E/ref=sr_1_2?ie=UTF8&qid=1511303325&sr=8-2&keywords=clear+matte+spray+paint	Rust-Oleum 285093 Stops Rust Protective Enamel Spray Paint, Matte Clear	3.68	2	Standard	Spray paint for the LED maze being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie Domingo. Item to reside in NSH 1207
Board and Cabinet						
Home Depot	https://www.homedepot.com/p/Sande-Plywood-Common-1-2-in-x-4-ft-x-8-ft-Actual-0-472-in-x-48-in-x-96-in-454532/100017950	Sande Plywood (Common: 1/2 in. x 4 ft. x 8 ft.; Actual: 0.472 in. x 48 in. x 96 in.)	31.95	4	Standard	Wood for the LED maze being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie Domingo. Item to reside in NSH 1207
Home Depot	https://www.homedepot.com/p/Sande-Plywood-Common-1-4-in-x-4-ft-x-8-ft-Actual-0-205-in-x-48-in-x-96-in-479023/100073744	Sande Plywood (Common: 1/4 in. x 4 ft. x 8 ft.; Actual: 0.205 in. x 48 in. x 96 in.)	19.92	3	Standard	Wood for the LED maze being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie Domingo. Item to reside in NSH 1207
Home Depot	https://www.homedepot.com/p/MDF-Panel-Common-1-2-in-x-4-ft-x-8-ft-Actual-1-2-in-x-49-in-x-97-in-M31240849097000000A/202332602	Plywood for building cabinet	23.45	5	Standard	Plywood to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Camera						
Amazon	https://www.amazon.com/TP-Link-Wireless-Ethernet-Archer-C50/dp/B0168G0KZY/ref=sr_1_3?ie=UTF8&qid=1511310452&sr=8-3&keywords=router+5ghz	Router to handle wire-less communications between the components	44.99	1	Standard	Router to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Angel Macias. Item to reside in NSH 1207
Logitech	https://www.logitech.com/en-us/product/circle-2-home-security-camera?buy=1	Wire-less Camera to be used for computer vision subsystem of the system	199.99	1	Standard	Camera to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Angel Macias. Item to reside in NSH 1207
Logitech	https://www.logitech.com/en-us/product/circle-2-re	Rechargeable, portable battery for the wire-less camera above	49.99	1	Standard	Camera battery to be part of a robot being built by Team General Robotics in 16-450 Robotics

	chargeable-battery?crid=1720					Systems Engineering for class project; requested by Angel Macias. Item to reside in NSH 1207
User Interface and Joystick						
Adafruit	https://www.adafruit.com/product/480	Small arcade joystick	14.95	1	Standard	Small joystick to be part of a robot being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Jae-Eun Lim. Item to reside in NSH 1207
Adafruit	https://www.adafruit.com/product/3055	Raspberry Pi 3	35	1	Standard	Raspberry Pi used for the user interface in the robotic system being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Nathalie Domingo. Item to reside in NSH 1207
Amazon	https://www.amazon.com/Dell-S2240T-21-5-Inch-LED-lit-Monitor/dp/B00CTODKIO/ref=sr_1_4?ie=UTF8&qid=1513495579&sr=8-4&keywords=touch%20screen%20display	Display	261.99	1	Standard	Display used for the user interface in the robotic system being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Angel Macias. Item to reside in NSH 1207
Main Processor						
Nvidia	https://www.nvidia.com/en-us/autonomous-machines/embedded-systems-dev-kits-modules/	Jetson TX2 Developer kit	299.99	1	Standard	Central processor used for the robotic system being built by Team General Robotics in 16-450 Robotics Systems Engineering for class project; requested by Angel Macias. Item to reside in NSH 1207

12. Fault Recovery

There are many potential faults or degraded modes of operations that can occur during an operation of the Pac-Man game. The following section highlights these events and how the system will handle them.

12.1 Misalignment of Overhead Camera

The overhead camera can become misaligned due to external shocks on the system, such as bumping into the board or knocking the camera accidentally. This can be a critical issue as the distorted view of the board will interrupt the Computer Vision process, which guides the path planning. To prevent misalignment of the overhead camera, we will attach the camera to an encoder that senses its orientation. Using feedback control, we will maintain the camera angle to make the camera face straight down at the board.

12.2 Losing a Component on the Robot During Operation

When the system wears out, some of the components of Pac-Man and Ghost will start to fall off, whether it be screws, LEDs, or wheels. These cases are inevitable as the system gets used for a long period of time. Maintenance would be the only possible recovery mode for such a case. Losing components can be detected from Computer Vision as it would cause abnormality either in visual or behavioral aspect. If an LED on the robot falls off, for instance, the shape of Pac-Man or ghost may be abnormal in the vision. Or if a wheel falls off, the robot would not move according to the command. When these abnormalities are detected by the Planner, the system can send a signal to the maintenance person to fix the problem.

12.3 Stuck Wheels

In the system, the wheels can get stuck through a few different ways. The wheels can get stuck behind one of the LEDs on the maze, stuck around the corner of a wall or event stuck behind a collapsed wall or obstacle in the way.

The system will resolve the first two degraded modes of operations by backing up the ghost or Pac-Man robot and re-orienting the robot with the help of computer vision to aid the robot and wheels to get unstuck from its position. However, for potential faults with unwanted obstacles or collapsed walls in our map, we will detect these using our overhead camera as the walls will be a distinct color from the maze background. We can then send a signal to the maintenance persons at the museum so that they are alerted to the issue and can promptly fix it.

12.4 Incorrect User Inputs

Moreover, there also potential faults with some user inputs that can be fed into the system – particularly with the ghost robot which is tele-operated by the users. There is a myriad of potentially faulty or incorrect user inputs that our system will have to correct such as attempting to move forward with the joystick command when there is a wall or obstacle in the way. Our system will solve this issue by using Ping sensor that enable us to determine the distance to an obstacle such as a wall. This will allow us to create a threshold where we prevent the user input that attempts to collide the robot into the obstacle. Another degraded mode of operation is feeding the diagonal inputs into the joystick. We will solve this issue by creating thresholds with each direction in which will only move the robot in the 4 constrained ways that we require – forward, backwards, left and right as opposed to including diagonally.

13. Detailed Design Treacability Matrix

	Visual Aids and Definitions	3 Difficulty Levels	Game Map	Overhead Camera	Screen Display	LED Grid	LED Screen	Game Music	Removeable Cover of Robots / Piecewise Components
13. Detailed Design Treacability Matrix	Visual Aids, Game Maps and Definitions and Rules should be displayed on the User Interface to ensure the user understands the rules and better engages the user	User Interface describes and sets 3 difficulty levelts to explain the path-planning algorithms to the users	The Game Map should be displayed onto the User Interface to show Pac-Man's path planning to better facilitate understanding	An Overhead Camera above the system to conduct Computer Vision and utilized various algorithms and techniques from Localization and CV to send commands to Pac-Man robot	The Screen to display the User Interface. It should include a Menu Page to select settings, Game Rules page to properly explain how to use the robots. It should do so using bullet points to be succint	Using an LED Grid as the maze to represent the Coins	The LED Screen will be hosted on top of the Pac-Man and Ghost robots to simulate the robots - i.e. Pac-Man eating the dots	The system will also have music similar to that of the Pac-Man game to make it more engaging in a museum setting	The Robots will have removeable covers to make sure that the robot and electronics are easily accesible. The Covers will be in the traditional shapes of Pac-Man and Ghosts to simulate the game experience. Other components in the system will also be peicewise to increase maintanability. The Game board will consist of layers to make the system aesthetic but also have all the neccesary components within the system
Satisfied Requirements									
EDUCATION									
R1.1	✓								
R1.2		✓							
R1.3	✓		✓						
R1.4					✓				
R1.5					✓				
R1.6	✓				✓				
GAME REQUIREMENTS									
R2.1						✓			
R2.2						✓			
R2.3							✓		
R2.4						✓			
R2.5		✓							
R2.6								✓	
R2.7								✓	
MAINTENANCE									
R3.1									
R3.2									✓
R3.3									✓
R3.4									✓
R3.5									
CHARGING									
R4.1									✓
R4.2									
BOARD REQUIREMENTS									
R5.1									✓
R5.2									
R5.3									
R5.4						✓			
SENSING									
R6.1									
R6.2									
R6.3									
R6.4									
R6.5									
R6.6									
MOBILITY									
R7.1									
R7.2									
R7.3						✓			
R7.4									
R7.5									
R7.6				✓					
R7.7									
R7.8									
COMMUNICATION									
R8.1									
R8.2									
R8.3									
R8.4				✓	✓				
R8.5									
R8.6									
R8.7									
R8.8									
R8.9									
USER INTERFACE									
R9.1	✓				✓				
R9.2					✓				
R9.3	✓				✓				
R9.4						✓	✓		✓
R9.5			✓						✓
R9.6					✓				
R9.7									
R9.8					✓				
R9.9									
DECISION MAKING									
R10.1									
R10.2									
R10.3									

	Long Lasting Battery	Replaceable Components	Plexiglass Cover	Computer Vision and Path Planning Algorithms	Ping Sensors	1080p Logitech Camera	Omni-Wheels	3/4" Gap between Robot and Wall	Featherwing / Rasberry Pi	Emergency Stop Button
13. Detailed Design Treacability Matrix	The system should have a long lasting battery to last throughout the day in a museum	The components should be easily replaceable to ensure the down time of the system is minimal if there is a part failure	There should be a glass cover over our board system (5ft by 5ft)	The system will utilize Computer Vision and Path Planning Algorithms for Localization and other techniques within the system	There will be Ping sensors mounted on each of the Pac-Man and Ghost robots	We will use a 1080p Logitech Camera for our overhead camera and use it for Computer Vision	Utilizing 4 Motors and Omni-Wheels for the Pac-Man and Ghost Robots for movement	There will be a 3/4" Gap between the robot and the wall on each side. This will ensure the robots will easily be able to fit through the maze	We will be using the Featherwing and Rasberry Pi boards in our system	There will be an emergency stop button to shut off the system, if required
Satisfied Requirements										
EDUCATION										
R1.1										
R1.2										
R1.3										
R1.4										
R1.5										
R1.6										
GAME REQUIREMENTS										
R2.1				✓						
R2.2				✓						
R2.3										
R2.4										
R2.5										
R2.6										
R2.7										
MAINTENANCE										
R3.1										✓
R3.2										
R3.3		✓								
R3.4		✓								
R3.5		✓								
CHARGING										
R4.1										
R4.2	✓									
BOARD REQUIREMENTS										
R5.1										
R5.2			✓							
R5.3			✓							
R5.4			✓							
SENSING										
R6.1				✓						
R6.2				✓	✓					
R6.3				✓	✓					
R6.4				✓						
R6.5				✓						
R6.6				✓	✓	✓				
MOBILITY										
R7.1							✓			
R7.2							✓			
R7.3										
R7.4				✓	✓					
R7.5					✓			✓		
R7.6				✓	✓	✓			✓	
R7.7							✓		✓	
R7.8							✓			
COMMUNICATION										
R8.1									✓	
R8.2									✓	
R8.3									✓	
R8.4						✓				
R8.5				✓					✓	
R8.6									✓	
R8.7									✓	
R8.8									✓	
R8.9									✓	
USER INTERFACE										
R9.1										
R9.2										
R9.3										
R9.4										
R9.5										
R9.6										
R9.7						✓				
R9.8										
R9.9									✓	
DECISION MAKING										
R10.1				✓	✓					
R10.2					✓					
R10.3				✓						

14. Trade Studies

Trade Study on the robot aesthetics

	Weight (1-5)	LED Screen		LED & 3D Printed Casing		3D Printed Casing		LCD Screen	
Cost	1	Screen costs $50, battery costs $30	2	$10 for I/O expansion boards, $12.95 for LEDs	4	3D printer filament costs $23	4	Screen costs $60	3
Power Consumption	5	Needs a separate power supply--cannot be powered by the raspberry pi	2	Can be powered off of the featherboard	4	None	5	Can be powered by the featherboard	4
Time to Create	3	Have to wire and program the screen--will only take about an hour or two	4	Wait time for 3D printer, about 1-2 hours to wire and program the LEDs	5	Only as long as the 3D printer takes--no code involved	5	Easy to upload animations to an LED screen and easy to determine the animation to display	4
Skills Required	4	Simple programming--libraries available that work with the LED screen	4	Simple programming and wiring--a lot of information available	4	Simple CAD	5	Libraries available--controlling the animation done through the raspberry pi	4
Wow Factor	5	Screen is very bright and the pixelated look goes well with the original game	5	Very bright, will cover the entire robot	5	Not bright like LEDs, but will be in the shape of Pac-Man or ghost	2	Will display the same animations as those in the real game	5
Size	4	Size is either too small (3in x 3in)l or too big (5in x 5in)	3	Adjustable	5	Adjustable	5	Make screens in the size we want	5
Capability	3	Can simulate Pac-Man getting eaten by the Ghost. Can simulate various movements.	5	Can simulate various movements, the ghost eyes moving, Pac-Man death sequence, and Pac-Man eating the Pac-Dots	5	Stationary	2	Can simulate any animation	5

TOTAL		**92**	**115**	**100**	**111**

Trade Study on the robot wheels

	Weight (1-5)	Non-holonomic Platform		3 Mecanum Wheels		4 Omni-Directional Wheels	
Cost	2	$39.90 (including a base and 2 motors)	5	$73/pack of 4 $20/motor x 3	2	$60/pack of 4 $20/motor x 4	2
Power Consumption	5	More power since every change in directions require 90-180 degree turns.	3	Least power since need only 3 motors.	5	More power since at least 4 motors needed	4
Time to Code	3	Forward, 90-degree turn, 180-degree turn (< 1 hour)	4	A bit more difficult to code (> 2 hours)	3	Forward, backward, left, right (< 30 minutes)	5
Time to Build	4	0 (pre-built platform)	5	Need to CAD base, attach wheels and motors. A bit more difficult to make the wheels in balance. (> 5 hours)	2	Need to CAD base, attach wheels and motors. (> 3 hours)	3
User Interface	4	Must turn 90 degree to make left or right. Must turn 180 degree to go backwards.	2	Smooth and instantaneous transition especially at turns.	5	Smooth and instantaneous transition especially at turns.	5
Size	4	Can make the robot 5 inch in diameter	5	Small mecanum wheels are difficult to find. The robot may be bigger than 6 inch in diameter.	2	Small omni wheels can be found but are still larger relative to non-holonomic wheels.	4
TOTAL			**85**		**74**		**87**

Trade Study on the main processor

	Weight (1-5)	Jetson TX2		Jetson TK1		Intel NUC (i5 - 7300U)		MSI GL62M - Laptop	
Cost	3	$300	4	$129	5	$418	2	$899.99	
Ease of Use	2	Requires some low level modification for use in this project	2	Requires some low level modification for use in this project	2	Ready to use	5	Ready to use	
Skills Required to Use	2	Unix based system, involved setup - decent documentation	3	Unix based system - lots of documentation	3	Windows	5	Windows	
Size	3	6.69 x 6.69 x 1.96 in	5	5 x 5 x 1 in	5	4 x 4 x 2 in	5	10.23 x 15.07 x 1.06 in	
Capability	5	4 + 1 CPU cores, 256 Pascal GPU cores, 8GB Memory	4	4 + 1 CPU cores, 192 Kepler GPU cores, 2GB Memory	2	2 Core/4 Thread CPU, Integrated GPU, Variable Memory	2	4 Core/8 Thread CPU, 1050 Ti Pascal GPU, 8GB Memory	
TOTAL			**57**		**50**		**51**		**51**

Trade Study on the display for the user interface

	Weight (1-5)	Asus Designo MX279H		Dell SS40T		ViewSonic TD2421		AOC I2267FW	
Cost	3	$300	2	$261.99	4	$288.29	3	$104.99	
Color Gamut (sRGB space)	4	~80%	3	84%	4	~68%	1	~70%	
Resolution	2	1920x1080	5	1920x1080	5	1920x1080	5	1920x1080	
Touch Capable	5	No	0	Yes	5	Yes	5	No	
Response Time	3	5ms	4	12ms	2	5ms	4	5ms	
Panel Technology (IPS,VA,TN)	4	IPS	5	VA	4	VA	4	IPS	
Contrast Ratio	3	1000:1	1	3000:1	3	3000:1	3	3000:1	
TOTAL			**63**		**88**		**85**		**70**

15. Verification Methods

15.1 Purpose

The purpose of this document is to analyze and create test plans or verification methods of our physical Pac-Man game that is used for both entertainment and educational purposes. In this document, we will outline the how we plan to test each system requirement that was discussed earlier. The document highlights why each requirement is important and the meaning of each test failure.

15.2 Verification Methods

The Table below illustrates our verification methods for each requirement and the meaning of each test failure. It outlines how we plan on testing each requirement.

Requirement	Metrics	Procedure	Meaning of Test Failure
Education			
R1.1	The test passes if each person correctly summarizes the concepts, which indicates that the lessons were effective.	Performance Test: 1. Recruit at least three non-major users of three different levels of knowledge (easy, medium, hard) in path planning. 2. Have each of them go through the lesson for their level of learning. 3. After the lesson, have each of them reiterate the path planning algorithm.	If users are not able to correctly summarize the concepts, it means the explanation of path planning is insufficient. To enhance the content of the lesson, take note of what particular aspects users have trouble understanding and add more details to enhance understanding of those specific aspects.
R1.2	The test passes if there are at least 3 levels of learning, namely easy, medium, and hard.	Functional Test: 1. Navigate through the interface. 2. Check if there are 3 levels of learning.	Less than 3 levels of learning limits the scope of users. Having easy, medium, and advanced levels will allow users of all literate ages to play the game. Make the missing level(s).
R1.3	The test passes if there is at least one image with texts on	Functional Test: 1. Navigate through the interface.	Path planning is difficult to visualize without visual aid, like diagrams and maps.

	pages, except for the menu page.	2. Note down how many images are in each page.	Visual aids should be included on all pages to facilitate user's understanding.
R1.4	The test passes if the screen is visible to the player and the bystanders throughout the game.	Functional Test: 1. Find one player, who will be playing the game, and at least two bystanders, who will be watching the game. 2. Check if the whole screen is visible to each of them throughout the game.	The game is intended not only for the player but also for the bystanders to learn about path planning. If the screen is not visible to at least two bystanders, the system is will be only useful to one or two people at a time. This is inefficient. The screen should be placed in a location where the player and at least two other bystanders should be able to see.
R1.5	The test passes when the average rating is above 6.	Performance Test: 1. Recruit at least 5 non-team members. 2. Have them play around with the interface. 3. Have them rate the easiness to navigate from 1 to 10, where 10 is easiest.	If the average rating is under 6, it means that the interface is difficult to navigate. This will waste unnecessary time for the user who is having trouble navigating.
R1.6	The test passes if the reading takes less than 1 minute.	Functional Test: 1. Read all the texts in one lesson in slow pace. 2. Time how long it takes to read everything.	In a museum setting, traffic flow must be controlled to prevent jams, especially for popular games like Pac-Man. If reading the texts takes more than 1 minute, there is most likely to be a line of people waiting to play the game.
Game			
R2.1	The test passes if the LED turns off immediately after Pac-Man passes	Functional Test: 1. Play one game of Pac-Man. 2. Check if each LED on the board turns	If an LED does not turn off immediately after Pac-Man passes through it or if it turns off before Pac-Man passes through it, the game

	through, but not before.	off when Pac-Man passes through.	rule will become unclear to the user. This will cause confusion.
R2.2	The test passes if the motors of both Pac-Man and the Ghost locks after all dots are eaten, and the music indicates that the game is over.	Functional Test: 1. Play one full game of Pac-Man. 2. Intentionally allow Pac-Man to win. 3. When all dots are eaten, check if game ends.	If the motors of Pac-Man and the Ghost do not lock after the dots are all eaten, the user will be able to keep moving the Ghost. The user may think that the game is still going. Similarly if the ending music is not played when the game is over, the user may not realize that the game has ended.
R2.3	The test passes if Pac-Man's LEDs turn off and the music indicates the game is over.	Functional Test: 1. Tele-operate the Ghost to move close to Pac-Man. 2. When the Ghost approaches Pac-Man within 1 inch, observe what happens to Pac-Man.	If the ending music is not played when the Ghost approaches within 1 inch of Pac-Man or Pac-Man's LEDs do not turn off, which are meant to simulate Pac-Man dying, the user may not realize that Pac-Man died and try to continue playing the game.
R2.4	The test passes if the Ghost is placed at the center of the board at the beginning of the game.	Functional Test: 1. Start a game. 2. Place the Ghost at the center of the board.	If the Ghost is not placed at the center of the board at the start of a game, the game rule is violated and the player may not know that it is the beginning of the game. This will cause confusion.
R2.5	The test passes if the Ghost can win at least 1/4 games on average.	Functional Test: 1. Play at least 8 games of Pac-Man. 2. Track how many games are won.	If the Ghost cannot win at least ¼ games on average, players are likely to get frustrated. The user is likely to get stressed than entertained, which defeats the purpose of the system.
R2.6	The test passes if the intro song matches that of the real game.	Functional Test: 1. Start the game. 2. Listen to the intro song.	If the intro song is not the same as that of the real Pac-Man game, the user will be confused.

R2.7	The test passes if the sound effects do not distract the players.	Performance Test: 1. Recruit at least 3 non-team members play the game. 2. Ask them if the sound effects of the game distracts them in any way.	If the sound effects distract the player, he or she will not be able to perform his or her best. This may frustrate the user.
Maintenance			
R3.1	The test passes if both robots can be turned off from the computer. This means that the robots can be shut off remotely in emergency situations.	Functional Test: 1. Turn on Pac-Man and the Ghost. 2. Turned them off from the computer.	If both robots cannot be turned off from the computer in emergency situations, unexpected things can result such as fire that destroys the robot. To prevent destruction of the robots, there must be a way to control them remotely.
R3.2	The test passes for the robots if each cover requires less than 2 tools to remove and less than 30 seconds to remove. The test passes for the board if it requires less than 2 tools to dismantle the board layers and less than 1 minute.	Functional Test: 1. Remove the covers from Pac-Man and the Ghost. 2. Count how many tools are required to remove the covers. 3. Measure how long it takes to remove the covers. 4. Dismantle the board layers. 5. Count how many tools are required to dismantle the layers. 6. Measure how long it takes to dismantle the layers.	If it requires more than 2 tools to remove the cover of the robots or dismantle the board layers, the task may become tedious. Also, if these tasks require more than 30 seconds to perform, then the user may become impatient with the maintenance, especially in a museum setting.
R3.3	The test passes if it lasts longer than TBD until maintenance is required.	Functional Test: 1. Play multiple games. 2. Measure how many hours pass until maintenance is required.	If it lasts less than TBD until maintenance is required, then the system would be under maintenance too frequently which may frustrate users.

R3.4	The test passes if it takes less than TBD to change the batteries of each subsystem.	Functional Test: 1. With the entire system fully assembled, change the batteries of the subsystems. 2. Measure how long it takes.	If it takes more than TBD to change the batteries of each system, the maintenance time would be too long for the user to be patient.
R3.5	The test passes if the system is operating normally at the end of TBD days of usage.	Performance Test: 1. Use the system TBD per day for TBD days.	If the system becomes unrecoverable before TBD days, the museum may be reluctant to set up the system because it is costly to replace it too frequently.
Charging			
R4.1	The test passes if the entire system is 100% charged with one charging port.	Functional Test: 1. Charge the entire system with one charging port.	If the system cannot be charged with one charging port, the charging process may become tedious. Since charging must be done frequently, this may frustrate users.
R4.2	The test passes if it takes less than TBD hours to fully charge the entire system.	Functional Test: 1. Drain all the batteries in the system. 2. Charge all the batteries to 100%.	If it takes more than TBD hours to fully charge the entire system, the system will be under maintenance for too long for the user to be patient.
Board			
R5.1	The test passes if none of the parts fall off from the system during transportation.	Performance Test: 1. Transport the system across the room for at least 3 yards. 2. Check if any parts of the system falls off.	Transportation of the system may become necessary when the system location is changed. It would be much more user friendly if the system does not have to be disassembled every time. If any parts of the system fall off during transportation of the system in its assembled state, it would be difficult to figure out where the part came from. The user may

			have to disassemble the system anyway to put the fallen parts back into their places.
R5.2	The test passes if the system still operates normally without any hazard.	Performance Test: 1. Spill water on the system. 2. Observe the system's response.	In a museum setting, people may have drinks and can accidentally spill them on the system. If the system burns or gets destroyed from fluids, the system would not last long in a museum.
R5.3	The test passes if none of the parts fall off from the system after being shaken.	Performance Test: 1. Shake the system. 2. Check if any parts of the system fall off.	In a museum setting, excited children may accidentally bump into the system. If parts fall off from the shock, the system would have to be disassembled to put the parts back into their places. This would frustrate users.
R5.4	The test passes if the length and width are within 5 ft x 5 ft.	Functional Test: 1. Measure the length and width of the outer border of the board.	The space inside the museum is limited. If the board is larger than 5ft x 5ft, many museums may not have enough space to accommodate it.
Sensing			
R6.1	The test passes if a coordinate corresponding to the centroid of the robot is given.	Functional Test: 1. Run CV algorithm on a video feed of the maze. 2. Calculate centroids 3. Check that centroids make sense in regards to the ground truth.	The true location of the robot is inaccurate or unknown.This can cause catastrophic failure with high-velocity collisions.
R6.2	The test passes if the Pac-Man robot autonomously navigates the maze without any collisions.	Functional Test: 1. Run CV algorithm to track robot locations. 2. Run AI and path planning code to	If the Pac-Man robot can't autonomously navigate the maze, then there is no adversary for the user to face. Further, an error here could allow for collisions.

		generate commands for Pac-Man. 3. Let the Pac-Man drive itself around maze.	
R6.3	The test passes if the mobile robots can detect each other.	Functional Test: 1. Run CV algorithm to track robot locations. 2. Turn on ping sensors. 3. Place robots near each other and near the maze. 4. Observe their observations about their environment.	If the robots can't detect each other, they may collide with each other, or with the maze.
R6.4	The test passes if the mobile robots stop moving before they collide with each other or with a wall.	Functional Test: 1. Run CV algorithm to track robot locations. 2. Turn on ping sensors. 3. Set robots on a collision course with each other or the maze walls. 4. Observe their observations in either case.	Collisions can cause the robots to damage each other or the maze.
R6.5	The test passes if the proper LED is shut off due to the Pac-Man robot passing through a specific coordinate.	Functional Test: 1. Run CV algorithm to track robot locations. 2. Track Pac-Man as it passes over an LED. 3. Send command to LED array to shut off LED and update Pac-Man's score.	If the proper LED doesn't shut off, then there's no way to keep track of Pac-Mans score, which will greatly reduce the entertainment value for the user.
R6.6	The test passes if the observed centroid of the robots is within	Performance Test:	If the test fails, the inaccuracy can lead to collisions and/or the

	0.5 cm of the ground truth centroids of the robots.	1. Run CV algorithm to track robot locations. 2. Measure the actual centroids of the robots within a set reference frame. 3. Calculate the error.	Pac-Man robot not being able to properly navigate the maze.
Mobility			
R7.1	The test passes if there is no significant vibrations in the robots motions.	Performance Test: 1. Run the robots on the board. 2. Observe their movements. 3. Note down any significant vibrations in their motions.	If the robots vibrate violently during movement, it may distract the player.
R7.2	The test passes if the turnings are visibly instantaneous and smooth.	Performance Test: 1. Drive the robots around corners on the board. 2. Observe the movement of the robots.	If it takes too long for the robots to turn, the player may become impatient with the game.
R7.3	The test passes if all LEDs are flush with the board.	Functional Test: 1. Run your hand over the entire path on the board. 2. Check if any LEDs are obtruding from the board.	If any of the LEDs are not flush with the board, the LEDs may get caught in the wheels of the robots. Also, the movement of the robots will become bumpy, which is not pleasant.
R7.4	The test passes if the motors of both robots get locked when they are 1 inch from each other.	Functional Test: 1. Bring Pac-Man and the Ghost close together. 2. When they are 1 inch away from each other, check if the motors get locked.	If the motors don't get locked when the robots are 1 inch from each other, they may collide into one another. This may cause damage.
R7.5	The test passes if the robots never bump	Performance Test:	If the robots bump into walls or damage any part of the

	into walls or damage any part of the board.	1. Drive the robots on the board. 2. Observe their movements. 3. Note down if they bump into walls or damage any part of the board.	board, the system would have to go under maintenance to replace the damaged components. If this happens frequently, users will be frustrated.
R7.6	The test passes if Pac-Man can eat all the dots for every game when it is allowed to win.	Functional Test: 1. Play at least 3 games. 2. Intentionally, do not kill Pac-Man. 3. Check if Pac-Man can eat all the dots for every game.	The purpose of the game is to show how Pac-Man can use path planning to complete the paths. If Pac-Man is unable to do that, the purpose of the system is defeated.
R7.7	The test passes if the robots can drive at a constant speed of TBD at all times.	Functional Test: 1. Drive the robots on the board. 2. Measure their speed. 3. Note any accelerations.	If the robots drive slower than TBD, the user may become impatient. If the robots drive faster than TBD, the game may end too soon.
R7.8	The test passes if the robots can turn within 300 ms.	Functional Test: 1. Move the robots horizontally and vertically. 2. Measure how long it takes to turn.	If turning of the robots takes more than 300ms, the flow of the game will not be smooth. This may frustrate users.
Communication			
R8.1	The test passes if all messages are received by the featherwing correctly	Functional Test: 1. Connect to main raspberry pi controller and featherwing. 2. Send messages between the Raspberry Pi and featherwing 3. Check to make sure all information is communicated	If messages are not received in their entirety, communication between components is not working correctly and will cause incorrect game play

R8.2	The test passes if the joystick commands are communicated to the central Raspberry Pi correctly	Functional Test: 1. Connect to the central Raspberry Pi 2. Use the joystick 3. Check the central Raspberry Pi to see the data that is being read from the joystick	If the joystick data does not correspond to the joystick input, the joystick could be connected incorrectly, or there could be communication issues, in which case the Ghost will not move according to the user input
R8.3	The test passes if the LED maze executes the commands correctly	Functional Test: 1. Connect to the central Raspberry Pi 2. Send commands to the LED maze controller 3. See if the command is executed by the LED maze	If the LED matrix does not execute commands correctly, incorrect game play will occur
R8.4	Test passes if during game play, the feed from the overhead camera is seen on the display	Functional Test: 1. Start a game 2. Check what is on the display	If the feed from the overhead camera is not seen on the display, it might be difficult for viewers to see the entire maze
R8.5	The test passes if the location of the robots according to the central Raspberry Pi matches the location of the actual robots	Functional Test: 1. Connect to the central Raspberry Pi 2. Start a game 3. Look at the location data the central Raspberry Pi has	If the location of the robots is not correct in the central Raspberry Pi, incorrect game play will occur
R8.6	The test passes if the time between moving the joystick and the corresponding motion in the robot is at most 2ms	Functional Test: 1. In conjunction with the tests for R8.2, time the latency between moving the joystick and the corresponding motion in the robot.	If the latency between user input and actual robot motion is greater than 2ms, the user might become irritated and it will affect game play
R8.7	The test passes if the time between sending a command from the central	Functional Test: 1. Connect to the central Raspberry Pi 2. Start a game	If the latency is greater than 2ms game play will be affected because it might be too easy for users to win

	Raspberry Pi and Pac-Man executing the command is at most 2ms	3. Send commands between the central Raspberry Pi and Pac-Man 4. Time the latency between sending the command and the execution of the command.	
R8.8	The test passes if the time between what is displayed on the screen versus actual game play is at most 2ms	Functional Test: 1. In conjunction with the test for R8.4, one look at the time difference between what is being displayed on the screen and what is actually occurring in real time in the system.	If the latency is greater than 2ms, displaying the camera feed is not useful for users
R8.9	The test passes if the time between sending a command from the central Raspberry Pi to the LED maze is at most 1ms	Functional Test: 1. In conjunction with the test for R8.3, one can time the difference between when a command is sent and when the corresponding command occurs in the LED maze.	If the latency is greater than 1ms, the user may become irritated or confused because the LED maze is not reacting to Pac-Man's motions quickly
User Interface			
R9.1	The test passes if there is a page on how to properly use the robot and all users can understand the instructions.	Functional Test: 1. Navigate through the interface. 2. Check to see if there is a page on how to properly use the robots. Performance Test: 1. Recruit at least 3 non-team members.	If the users are not able to navigate through the user interface intuitively, it will cause confusion about how to control the robot. This can lead to incorrect user inputs to the robot that can cause damage to the system.

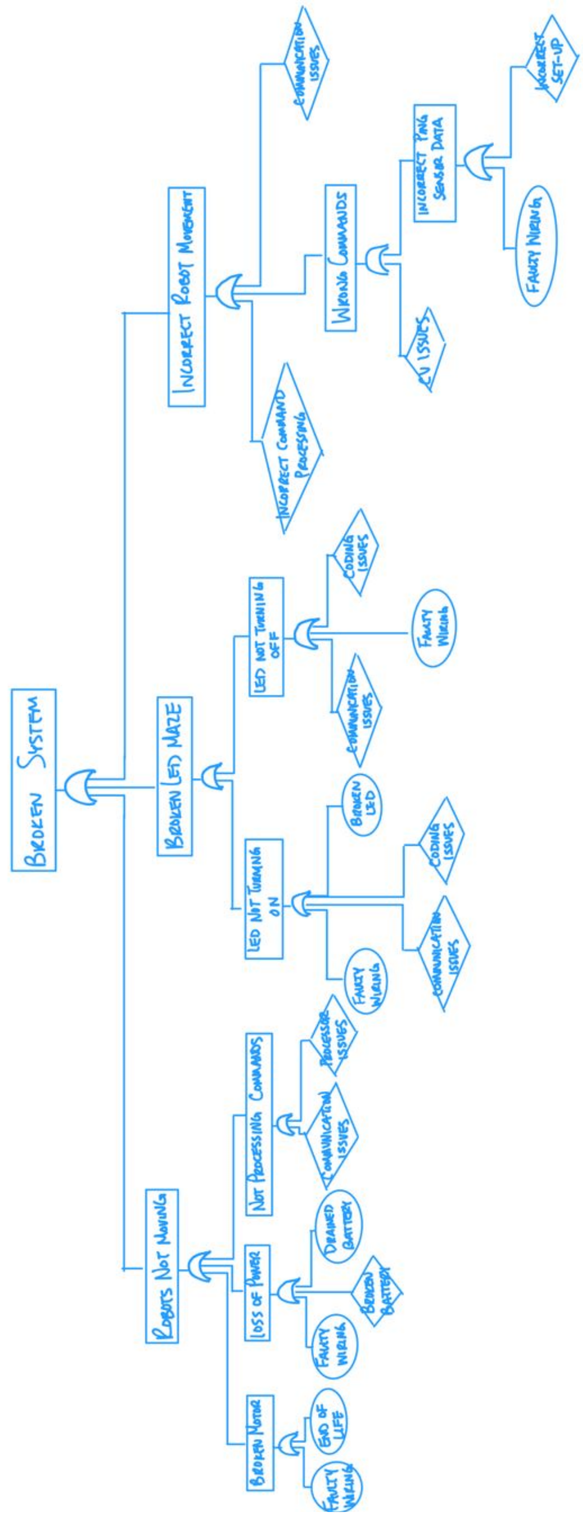
		2. Have them read over the instructions on how to properly use the robots. 3. Test them to see how well they have understood how to use the robots.	
R9.2	The test passes if there is a menu with 3 levels of learning. This menu will enhance user experience by making navigation easier.	Functional Test: 1. Navigate through the interface. 2. Check that there is a menu with 3 levels of learning.	If there is no menu for the different levels of learning, it may not realize that there are 3 levels of learning. A menu page can make the options clear.
R9.3	The test passes if there is a page on game rule and all users can understand the rules very well.	Functional Test: 1. Navigate through the interface. 2. Check to see if there is a page on game rule. Performance Test: 1. Recruit at least 3 non-team members. 2. Have them read over the game rules. 3. Test them to see how well they understood the rules.	If the user is not intuitively able to understand the game rules, it can create confusion about what the user has to do in the system. This can take away from the learning and educational component of the game.
R9.4	The test passes if the color scheme is not monotone and the average rating of the aesthetics is above 5.	Functional Test: 1. Navigate through the interface. 2. Check to see that the color scheme is not monotone. Performance Test: 1. Recruit at least 3 non-team members. 2. Have them rate the aesthetic of the	If the game is monotone, it removes the engaging and entertaining aspect of the system. Since the system will be displayed in a museum, it must be engaging to draw users in. If this test fails, our system will not be able to draw users in to participate and learn more about our system.

		educational subsystem.	
R9.5	The test passes if all subsystems have similar color to the real game counterpart.	Functional Test: 1. Compare each subsystem to its real game counterpart. 2. Check to see if all colors of the subsystem matches the real Pac-Man game.	If the system does not have a similar color scheme to the original game, the users may not recognize the game.
R9.6	The test passes if everyone who knows about Pac-Man can recognize that our system is Pac-Man.	Performance Test: 1. Recruit at least 3 non-team members who do not know anything about our system but have played Pac-Man before. 2. Show them our system. 3. Check if they can recognize the game.	Pac-Man is well known and easily recognizable. If a third party cannot recognize that our system is Pac-Man, it indicates that the general aesthetic of our system does not match the real game.
R9.7	The test passes if the refresh rate of the interface is at least 60Hz.	Functional Test: 1. Run a complete round of path planning lesson on the interface. 2. Measure the refresh rate of the interface.	If this test fails, then the user cannot see the update map based on the path-planning algorithm. This will lead to a decreased amount of understanding of what the robot is doing.
R9.8	The test passes if the refresh rate of the video feed is at least 30Hz.	Functional Test: 1. Show the video feed on the display. 2. Measure the refresh rate of the video feed.	If the refresh rate of the video feed is greater than 30Hz, the video would not be real-time. This lag can distract the player.
R9.9	The test passes if the input lag from the user is under 2ms.	Functional Test: 1. Control the Ghost with the joystick. 2. Measure the input lag from the user.	If this tests fails, the user commands to the robot will be delayed. This can lead to damage to the system by colliding with walls and not turning when the user

			wanted and disrupt the flow of the game which can lead to dissatisfaction from the users.
Decision Making			
R10.1	The test passes if there is no behavior from which Pac-Man harms itself or others.	Performance Test: 1. Play at least 3 games. 2. Observe Pac-Man's behavior. 3. Note down any behavior from which Pac-Man harms itself or others.	If Pac-Man harms itself during the game, it can lead to damage of the robot that will lead to degraded operation and it will not run as required or at all. If the robot harms the users in any way, it will decrease the safety of the users and humans around the system which will take away from the education and entertaining requirements of the system as well as put the users at risk.
R10.2	The test passes if the robots never collide with the walls or each other.	Functional Test: 1. Play at least 3 games. 2. Observe the robots' behavior. 3. Note down if the robots collide with the walls or each other.	If this test fails, the collisions with the walls or between robots can cause damage to the robots or the maze.
R10.3	The test passes if Pac-Man always takes the optimal path.	Performance Test: 1. Play at least 5 games. 2. Check if Pac-Man takes the optimal path.	If this tests fails, the system can cause confusion about how path-planning works and takes away from the educational component of the system. It takes away from some of the challenge of playing the game as well.

16. Fault Trees and Tables

16.1 Fault Tree



16.2 Fault Table												
Component/Item	Function/Process	Failure Mode	Failure Cause	Failure Effect	Severity	Severity Reasoning	Occurence	Occurence Reasoning	Detection	Detection Reasoning	Risk	Failure Handling
Pac-Man	Movement	Stuck wheels	stuck on an LED, stuck around corner of a wall, or trapped by collapsed wall or obstacle	System down time	3	The system should not be delayed and should not require considerable down time	1	All of the failure causes are prevented through the design of our system. LEDs will be flush with the ground, and thus will not be an obstacle to the robot. The maze will be built such that it should not fall apart and such that the robot can easily manuever within it. We also have additional sensors on the robot to prevent collisions.	1	Computer vision that utilizes the overhead camera will be monitoring Pac-Man during the game	3	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
Pac-Man	Movement	No movement	broken motors, loss of power, not processing commands, faulty wiring	System down time	3	The system should not be delayed and should not require considerable down time	2	Broken motors is something we expect of the system as replacement of the motors is one of the expected maintenance tasks. Loss of power should not occur often because of wireless charging. Pac-Man should be processing commands correctly. Wires will be connected correctly and in such a way to minimize possible disconnects.	1	Computer vision that utilizes the overhead camera will be monitoring Pac-Man during the game	6	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
Pac-Man	Movement	Collides into a wall	incorrect path planning, motors not wired correctly, incorrect processing of commands	Incorrect game play, possibly system down time	2	The system should work according to specification, including the movement of the Pac-Man robot. System should also not require considerable down time	1	Path planning should work correctly, the processor on Pac-Man should process commands correctly. Motors will be wired correctly.	1	Computer vision that utilizes the overhead camera will be monitoring Pac-Man during the game	2	Pac-Man robot has ping sensors to have real time data on the distance between the robot and the walls. This data is used to correct movement and to prevent collisions with the walls.
Pac-Man	Movement	Collides into Ghost	motors not wired correctly, incorrect processing of commands	Incorrect game play, possibly system down time	2	The system should work according to specification, including the movement of the Pac-Man robot. System should also not require considerable down time	1	The processer on Pac-Man should process commands correctly. Motors will be wired correctly. Will have commands sent to the robot to prevent collisions between Pac-Man and ghost.	1	Computer vision that utilizes the overhead camera will be monitoring Pac-Man during the game	2	Computer vision will monitor distance between the ghost and Pac-Man and send a command for Pac-Man to stop if the robots get too close. Additionally, the ping sensors on Pac-Man will be used to prevent collisions with obstacles, including the Ghost robot.
Pac-Man	Movement	Component falls off of robot	extended use of the system, incorrect mounting of parts	possibly system down time, obstacles for robots	3	Depending on the part that comes off, it can either be critical to the operation of Pac-Man, in which case maintenance will be required. If it is a non-critical part, it will become an obstacle in the maze which will require maintenance to remove the part	2	This is common with extended use of the system.	1	Computer vision that utilizes the overhead camera will be monitoring Pac-Man during the game.	6	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.

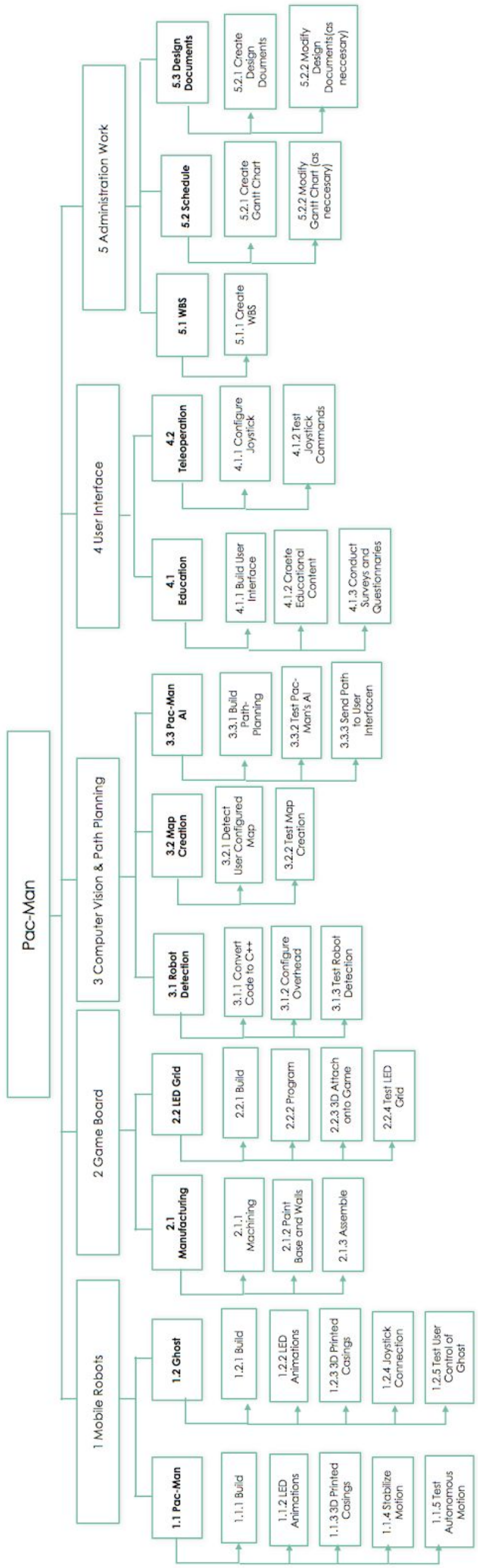
Ghost	Movement	Stuck wheels	stuck on an LED, stuck around corner of a wall, or trapped by collapsed wall or obstacle	System down time	3	The system should not be delayed and should not require considerable down time	1	All of the failure causes are prevented through the design of our system. LEDs will be flush with the ground, and thus will not be an obstacle to the robot. The maze will be built such that it should not fall apart and such that the robot can easily manuever within it. We also have additional sensors on the robot to prevent collisions.	1	Computer vision that utilizes the overhead camera will be monitoring Ghost during the game	3	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
Ghost	Movement	No movement	broken motors, loss of power, not processing commands, faulty wiring	System down time	3	The system should not be delayed and should not require considerable down time	2	Broken motors is something we expect of the system as replacement of the motors is one of the expected maintenance tasks. Loss of power should not occur often because of wireless charging. Ghost should be processing commands correctly. Wires will be connected correctly and in such a way to minimize possible disconnects.	1	Computer vision that utilizes the overhead camera will be monitoring Ghost during the game	6	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
Ghost	Movement	Incorrect diagonal movement	user input	ghost does not move according to specification. Might run into an obstacle	2	The system should work according to specification, including the movement of the ghost robot	1	The ghost will only be able to move forward, backward, left, and right	1	Computer vision that utilizes the overhead camera will be monitoring Ghost during the game	2	Ghost programmed so that it can only move in the allowable directions
Ghost	Movement	Collides into a wall	incorrect user input, motors not wired correctly, incorrect processing of commands	Incorrect game play, possibly system down time	2	The system should work according to specification, including the movement of the Ghost robot. System should also not require considerable down time	1	User input will be monitored to ensure incorrect commands are not carried out. The processor on Ghost should process commands correctly. Motors will be wired correctly.	1	Computer vision that utilizes the overhead camera will be monitoring Ghost during the game	2	Ghost robot has ping sensors to have real time data on the distance between the robot and the walls. This data is used to correct movement and to prevent collisions with the walls.
Ghost	Movement	Collides into Pac-Man	incorrect user input, motors not wired correctly, incorrect processing of commands	Incorrect game play, possibly system down time	2	The system should work according to specification, including the movement of the Ghost robot. System should also not require considerable down time	1	User input will be monitored to ensure incorrect commands are not carried out. The processor on Ghost should process commands correctly. Motors will be wired correctly. Will have commands sent to the robot to prevent collisions between Pac-Man and ghost.	1	Computer vision that utilizes the overhead camera will be monitoring Ghost during the game	2	Computer vision will monitor distance between the ghost and Pac-Man and send a command for Ghost to stop if the robots get too close. Additionally, the ping sensors on Ghost will be used to prevent collisions with obstacles, including the Pac-Man robot.
Ghost	Movement	Component falls off of robot	extended use of the system, incorrect mounting of parts	possibly system down time, obstacles for robots	3	Depending on the part that comes off, it can either be critical to the operation of Ghost, in which case maintenance will be required. If it is a non-critical part, it will become an obstacle in the maze which will require maintenance to remove the part	2	This is common with extended use of the system.	1	Computer vision that utilizes the overhead camera will be monitoring Ghost during the game.	6	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.

LED Board	Simulating Pac-Dots	LEDs don't turn off when Pac-Man passes above	cv data not received, data not processed correctly, faulty wiring	Incorrect game play, possibly system down time	2	The system should work according to specification, including the LED maze. System should also not require considerable down time	1	CV data should be sent, data should be processed correctly for the maze, and the LEDs will be hooked up correctly and in such a way to minimize possible failure	1	Computer vision that utilizes the overhead camera will be monitoring the LED maze during the game	2	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
LED Board	Simulating Pac-Dots	LEDs don't turn on at beginning of game	faulty wiring, broken LEDs	Incorrect game play and system down time	3	The system should work according to specification, including the LED maze. System should also not require considerable down time	2	after continual use, LEDs are likely to break	1	Computer vision that utilizes the overhead camera will be monitoring the LED maze during the game	6	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
LED Board	Simulating Pac-Dots	incorrect LEDs turn off in response to Pac-Man movement	faulty wiring, cv data not processed correctly, wrong cv data received	Incorrect game play, possibly system down time	2	The system should work according to specification, including the LED maze. System should also not require considerable down time	1	correct CV data should be sent. Data should be processed correctly for the maze. LEDs will be hooked up correctly and in such a way to minimize possible failure	1	Computer vision that utilizes the overhead camera will be monitoring the LED maze during the game	2	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
Wireless Charging	Charging robots	robots don't charge	faulty wiring, coils too far	robots lose power leading to system down time	3	The system should not be delayed and should not require considerable down time	1	wireless charging will be wired correctly. Coils will be positioned such that the robots can charge.	1	Computer vision monitors the Ghost and Pac-Man robots throughout the game and will notice if either or both are not moving. Central controller can also ping the robots to see if they are on	3	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
Wireless Communications	Receiving data	router does not receive data	components are not functioning correctly	Incorrect game play and system down time	3	The system should not be delayed and should not require considerable down time	1	Communication between componets and the router will work correctly	3	Router will not know when game is in play and thus it is up to the system operator to recognize when data is not being received	9	
Wireless Communications	Delivering commands	router does not send out commands	router not functioning properly, central controller not functioning	Incorrect game play and system down time	3	The system should work according to specification, including the computer vision. System should also not require considerable down time	1	Communication between componets and the router will work correctly. Central controller will also function properly	3	Router will not know when game is in play and thus it is up to the system operator to recognize when commands are not being sent out	9	
Computer Vision	Tracking robots	incorrect robot localization	camera not mounted correctly, incorrect computer vision algorithm, bad environment (lighting, obstructions, etc.)	Incorrect game play and system down time	3	The system should work according to specification, including the computer vision. System should also not require considerable down time	1	The system will be in the correct enviroment, the camera will be securely mounted and the computer vision algorithm will correctly process all of the images	3	If computer vision is faulty, it is up to the system operator to recognize that there is something wrong with the system	9	
Computer Vision	Processing images	cannot process images	camera not mounted correctly, insufficient computer vision algorithm, bad environment (lighting, obstructions, etc.)	Incorrect game play and system down time	3	The system should work according to specification, including the computer vision. System should also not require considerable down time	1	The system will be in the correct enviroment, the camera will be securely mounted and the computer vision algorithm will be able to process all the images	3	If computer vision is faulty, it is up to the system operator to recognize that there is something wrong with the system	9	

Computer Vision	Commanding LED maze	sends wrong Pac-Man location to LED maze	incorrect tracking of Pac-Man, communication issue	Incorrect game play and system down time	3	The system should work according to specification, including the computer vision. System should also not require considerable down time	1	Computer vision algorithm will correctly track Pac-Man and the central controller will process the data correctly and send the correct command	3	If computer vision is faulty, it is up to the system operator to recognize that there is something wrong with the system	9	
Path Planning	Eat Pac-Dots	Pac-Man does not eat any of the Pac-Dots	broken Pac-Man robot, incorrect path planning algorithm, faulty data/commands	Incorrect game play and system down time	3	The system should work according to specification, including the computer vision. System should also not require considerable down time	2	Path planning will be extensively tested to ensure that the algorithm is optimal. After continue use of the system, Pac-Man is likely to break	2	Computer vision will be tracking Pac-Man and its movements throughout the game	12	If the system has to be shut down, an alert will be shown on the system to signal to the operator that the system requires attention.
User Interface Screen and Joystick	User commands	Not processing joystick data	joystick not hooked up properly, communication issue	Incorrect game play and system down time	3	The system should work according to specification, including the computer vision. System should also not require considerable down time	1	Joystick will be hooked up correctly and communication between components will work	3	It is up to the system operator to recognize that there is something wrong with the system	9	
User Interface Screen and Joystick	User interface screen	Not displaying any information on the screen	display not hooked up properly, communication issue	Incorrect game play and system down time	3	The system should work according to specification, including the computer vision. System should also not require considerable down time	1	Display will be hooked up correctly and communication between components will work	3	It is up to the system operator to recognize that there is something wrong with the system	9	

17. WBS and WBS Dictionary

17.1 WBS



17.2 WBS Dictionary

WBS #:	1.1.1	Task:	Building Pac-Man Robot
Est. Level of Effort:	10 Hours	Owner:	Esther, Mehar
Resources Needed:	Arduino, Robot Hardware, LEDs	Work Products:	Pac-Man Robot
Description of Task:	Assemble the hardware pieces for the robot. Assemble the LED array on the top of the robot.		
Input:	Robot hardware and LEDs. Hardware schematics.		
Dependencies:	Shipment of hardware pieces. Assembly of LED array.		
Risks:	3D printed pieces might not perfectly fit. LED array might need a diffuser.		

WBS #:	1.1.2	Task:	LED Animations
Est. Level of Effort:	5 Hours	Owner:	Nathalie
Resources Needed:	Arduino, Robot Hardware, LEDs	Work Products:	Pac-Man Robot w/ LED animations
Description of Task:	Program the LEDs on the top of the Pac-Man robot to perform certain animations.		
Input:	Robot hardware and LEDs.		
Dependencies:	Assembled LED array.		
Risks:	LEDs might burn out during operation.		

WBS #:	1.1.3	Task:	Printed Casing
Est. Level of Effort:	2 Hours	Owner:	Esther, Mehar
Resources Needed:	3D printer	Work Products:	Pac-Man Casing
Description of Task:	Finish designing and then printing final casing for the Pac-Man robot.		
Input:	Robot design specs.		
Dependencies:	CAD models.		

Risks:	Case may not fit with other components.

WBS #:	1.1.4	Task:	Stabilize Motion
Est. Level of Effort:	5 Hours	Owner:	Nathalie, Esther
Resources Needed:	Ping sensors, PID	Work Products:	Pac-Man Robot
Description of Task:	Tune PID parameters for the ping sensors so that the robots move equidistant from the walls.		
Input:	PID tuning		
Dependencies:	Robot chassis must be done.		
Risks:	PID sensor readings hard with maze layout.		

WBS #:	1.1.5	Task:	Test Autonomous Motion
Est. Level of Effort:	10 Hours	Owner:	Angel
Resources Needed:	Pac-Man, Board, Code	Work Products:	Autonomous Pac-Man Robot
Description of Task:	Test if the path planning algorithm works with the actual system.		
Input:	Path planning algorithm		
Dependencies:	Board and robots must be completely built. A complete draft of path planning code must be written.		
Risks:	It may take hours to debug the code.		

WBS #:	1.2.1	Task:	Building Ghost Robot
Est. Level of Effort:	10 Hours	Owner:	Esther, Mehar
Resources Needed:	Arduino, Robot Hardware, LEDs	Work Products:	Ghost Robot
Description of Task:	Assemble the hardware pieces for the robot. Assemble the LED array on the top of the robot.		
Input:	Robot hardware and LEDs. Hardware schematics.		
Dependencies:	Shipment of hardware pieces. Assembly of LED array.		
Risks:	3D printed pieces might not perfectly fit. LED array might need a diffuser.		

WBS #:	1.2.2	Task:	LED Animations
Est. Level of Effort:	5 Hours	Owner:	Nathalie
Resources Needed:	Arduino, Robot Hardware, LEDs	Work Products:	Ghost Robot w/ LED animations
Description of Task:	Program the LEDs on the top of the Ghost robot to perform certain animations.		
Input:	Robot hardware and LEDs.		
Dependencies:	Assembled LED array.		
Risks:	LEDs might burn out during operation.		

WBS #:	1.2.3	Task:	Printed Casing
Est. Level of Effort:	2 Hours	Owner:	Esther, Mehar
Resources Needed:	3D printer	Work Products:	Ghost Casing
Description of Task:	Finish designing and then printing final casing for the Ghost robot.		
Input:	Robot design specs.		
Dependencies:	CAD models.		
Risks:	Case may not fit with other components.		

WBS #:	1.2.4	Task:	Joystick Connection
Est. Level of Effort:	3 Hours	Owner:	Esther, Mehar
Resources Needed:	Joystick, microcontroller, Ghost	Work Products:	User Control System
Description of Task:	Connect the joystick to the Ghost robot.		
Input:	Joystick code, controller layout.		
Dependencies:	Ghost robot must be completed.		
Risks:	Joystick selection may turn out to be not optimal.		

WBS #:	1.2.5	Task:	Test User Control of Ghost
Est. Level of Effort:	1Hours	Owner:	Esther
Resources Needed:	Joystick, Ghost, Board	Work Products:	User Control System
Description of Task:	Test user control of the Ghost using joystick.		
Input:	Control scheme		
Dependencies:	Joystick installation completed.		
Risks:	Control may not be intuitive as intended.		

WBS #:	2.1.1	Task:	Machining
Est. Level of Effort:	12 Hours	Owner:	Mehar, Esther
Resources Needed:	Drill press, band saw, grinder, plywood, nails	Work Products:	Board components
Description of Task:	Cut out board pieces, drill holes, and polish plywood.		
Input:	Board drawing, CAD models		
Dependencies:	Have complete detailed design of board.		
Risks:	Design flaws might cause difficulty in assembling the board. Costly to redo.		

WBS #:	2.1.2	Task:	Paint Board Parts
Est. Level of Effort:	8 Hours	Owner:	Angel
Resources Needed:	Brush, paint	Work Products:	Painted Board
Description of Task:	Paint walls, borders, pillars blue, paint base black		
Input:	Color scheme, design specs		
Dependencies:	Board parts must be machined.		
Risks:	Incorrect paint color when dried, incomplete machining may require repaint later		

WBS #:	2.1.3	Task:	Assemble
Est. Level of Effort:	2 Hours	Owner:	Esther and Mehar
Resources Needed:	Nails, hammer	Work Products:	Board
Description of Task:	Nail borders and pillars to the base. Insert walls into the base.		
Input:	Assembly instructions		
Dependencies:	Board machined and painted.		
Risks:	Machined board parts may not fit well together. Parts may be missing.		

WBS #:	2.2.1	Task:	Build
Est. Level of Effort:	8 hours	Owner:	Nathalie
Resources Needed:	Soldering iron, solder, wire	Work Products:	Board
Description of Task:	Solder LEDs together so they fit into the game board and such that we can control each LED individually.		
Input:	Board drawing, LED schematic		
Dependencies:	Board machined and holes drilled		
Risks:	LEDs fragile and soldering connection might be weak		

WBS #:	2.2.2	Task:	Program
Est. Level of Effort:	2 hours	Owner:	Nathalie
Resources Needed:	Assembled LED board	Work Products:	Board
Description of Task:	Program the LED grid to take in a command about specific LED status and change the LEDs accordingly		
Input:	LED schematic		
Dependencies:	LED board must be assembled		
Risks:	Difficult to keep track of all the individual LEDs		

WBS #:	2.2.3	Task:	Attach onto game board
Est. Level of Effort:	1 hour	Owner:	Nathalie
Resources Needed:	tape	Work Products:	Board
Description of Task:	Attach the LEDs to the physical game board, make sure they are on securely to ensure movement of parts		
Input:	LED schematic		
Dependencies:	Assembled LED grid		
Risks:	LEDs fragile and solder connections might break from movement		

WBS #:	2.2.4	Task:	Test LED grid
Est. Level of Effort:	1 hour	Owner:	Nathalie
Resources Needed:	LED grid	Work Products:	Board
Description of Task:	Make sure LED grid works with commands from other systems		
Input:	LED schematic		
Dependencies:	Assembled LED grid, CV		
Risks:			

WBS #:	3.1.1	Task:	Convert MATLAB code to python/c++
Est. Level of Effort:	10 hours	Owner:	Angel
Resources Needed:	Python, C++, MATLAB	Work Products:	Computer vision program to detect colored robots
Description of Task:	Convert the prototype code from MATLAB to C++/Python.		
Input:	MATLAB prototype code.		
Dependencies:	Camera, compiler/interpreter		
Risks:	Unseen bugs creep into the conversion.		

WBS #:	3.1.2	Task:	Configure overhead camera
Est. Level of Effort:	2	Owner:	Angel
Resources Needed:	Camera, computer	Work Products:	Properly situated camera
Description of Task:	Camera placement needs to be tuned to properly view the board without too much distortion due to high FOV (180 degrees)		
Input:	Design specs, desired reference frame (i.e. how much of the image should be just the board)		
Dependencies:	Ultimate height of the camera rig/podium, lighting conditions, finished board		
Risks:	Field of view might skew image too much.		

WBS #:	3.1.3	Task:	Test Robot Detection
Est. Level of Effort:	5 hours	Owner:	Angel
Resources Needed:	Pac-Man, Ghost, finished board	Work Products:	Robust robot detection and collision warning
Description of Task:	Test the robot detection code on the actual robots.		
Input:	Design specs for desired distance threshold to detect for.		
Dependencies:	Proper camera placement, finished robot controls, finished board.		
Risks:	Bugs from earlier might still be present and unfound.		

WBS #:	3.2.1	Task:	Detect User Configured Map
Est. Level of Effort:	10 hours	Owner:	Angel
Resources Needed:	LED array schematic, Board hole map, computer with C++/Python/MATLAB	Work Products:	CV program to detect the user configured map
Description of Task:	This program will store an array of hole/LED positions from the board to detect the locations that the user placed the walls. Then the program detects whether the map is valid.		
Input:	Design specs.		
Dependencies:	Finished board, configured camera		
Risks:	There may be errors in detecting the walls in certain spots.		

WBS #:	3.2.2	Task:	Test Map Creation
Est. Level of Effort:	5 hours	Owner:	Angel
Resources Needed:	Computer with C++,Python, video feed of board	Work Products:	Finished Map detection code
Description of Task:	Test and remove bugs from the map configuration detector on the board.		
Input:	Design specs, user configured map		
Dependencies:	Finished board, configured camera		
Risks:	There may be bugs left in the code.		

WBS #:	3.3.1	Task:	Build Path-Planning Algorithm
Est. Level of Effort:	30 Hours	Owner:	Angel
Resources Needed:	Computer with C++/Python/MATLAB	Work Products:	A.I. for Pac-Man
Description of Task:	Create and A.I. for the Pac-Man robot that allows it maximize its score against an opponent.		
Input:	Design specs, research papers for A.I./path-planning		
Dependencies:	Finished robot controls		
Risks:	Effort estimate may be inaccurate. The path-planning may end up being too robust for a human to be able to defeat.		

WBS #:	3.3.2	Task:	Test Pac-Man AI
Est. Level of Effort:	10 hours	Owner:	Angel, Nathalie, Esther, Mehar
Resources Needed:	Router, computer with C++/Python	Work Products:	Properly tested/configured AI for Pac-Man
Description of Task:	Test the Pac-Man A.I. against a person playing with it on the board.		
Input:	User input to the Ghost robot.		
Dependencies:	Finished boards, finished robots, finished LED grid.		
Risks:	Bugs may be left unfound, A.I. might be difficult to change to make easier/harder to defeat.		

WBS #:	3.3.3	Task:	Send Path to User Interface
Est. Level of Effort:	3 hours	Owner:	Angel
Resources Needed:	Router, computer with C++/Python	Work Products:	User interface has information for the educational segment of the system.
Description of Task:	Properly configure the necessary AI data to send to the user interface, then send it.		
Input:	User input to the Ghost robot for the game session.		
Dependencies:	Finished board, finished robots, finished computer vision, finished path-planning, in-progress user interface.		
Risks:	Level of detail of the sent information may be too high or too low.		

WBS #:	4.1.1	Task:	Build user interface
Est. Level of Effort:	10 hours	Owner:	Nathalie, Esther, Mehar
Resources Needed:	Pac-Man game design	Work Products:	User interface
Description of Task:	Create the overall structure for the user interface in the style of Pac-Man. This includes the home screen, game screen, and start game screen		
Input:	Project design		
Dependencies:	CV		
Risks:	User interface might not be intuitive to targeted audience		

WBS #:	4.1.2	Task:	Create educational content
Est. Level of Effort:	4 hours	Owner:	Nathalie, Esther, Mehar
Resources Needed:	Desired educational content	Work Products:	User interface
Description of Task:	Portray the game details in a way specific to the selected game level.		
Input:	Project design		
Dependencies:	CV and path planning		
Risks:	User interface might not be intuitive to targeted audience		

WBS #:	4.1.3	Task:	Conduct surveys and questionnaires
Est. Level of Effort:	2 hours	Owner:	Mehar
Resources Needed:	Surveys	Work Products:	User interface
Description of Task:	Create a work breakdown structure sheet to outline the tasks and subtasks necessary for the project.		
Input:	Capstone project design and testing and validation documents.		
Dependencies:	Completed user interface		
Risks:	Might not have representative audience for surveys		

WBS #:	4.2.1	Task:	Configure joystick
Est. Level of Effort:	2 hours	Owner:	Nathalie
Resources Needed:	Joystick, jetson	Work Products:	User interface
Description of Task:	Connect the joystick such that a user can operate the ghost robot		
Input:	Project design		
Dependencies:	Router communication		
Risks:	Incompatibility with Jetson		

WBS #:	4.2.2	Task:	Test joystick commands
Est. Level of Effort:	2 hours	Owner:	Nathalie
Resources Needed:	Joystick	Work Products:	User interface
Description of Task:	Make sure the user can control the ghost robot through the joystick		
Input:	Project design		
Dependencies:	Working ghost robot		
Risks:	High latency		

WBS #:	5.1.1	Task:	Create WBS
Est. Level of Effort:	2 hours	Owner:	Mehar
Resources Needed:	Google sheets	Work Products:	WBS
Description of Task:	Create a work breakdown structure sheet to outline the tasks and subtasks necessary for the project.		
Input:	Capstone project design and testing and validation documents.		
Dependencies:			
Risks:			

WBS #:	5.1.2	Task:	WBS Dictionary
Est. Level of Effort:	6 Hours	Owner:	Angel, Esther, Mehar, Nathalie
Resources Needed:	Google sheets	Work Products:	WBS Dictionary
Description of Task:	Designate tasks and subtasks to everyone in the group while giving certain details about these tasks (time, resources, dependencies, etc.). Formalize these tasks in an easily searchable list.		
Input:	WBS chart		
Dependencies:	WBS chart, specializations of members		
Risks:			

WBS #:	5.2.1	Task:	Create Gantt Chart
Est. Level of Effort:	2 Hours	Owner:	Mehar
Resources Needed:	Microsoft Excel	Work Products:	Excel Spreadsheet
Description of Task:	Create a schedule that outlines how long each task will take and all the dependencies between the tasks		
Input:	Gantt Chart Template		
Dependencies:	Detailed Design Documents and Testing Documents completed		
Risks:	-		

WBS #:	5.2.2	Task:	Update Gantt Chart
Est. Level of Effort:	1 Hour	Owner:	Mehar
Resources Needed:	Microsoft Excel	Work Products:	Excel Spreadsheet
Description of Task:	Update the Gantt Chart as the timelines of each tasks changes or gets delayed to see track the progress of the project and determine the finish date of the project.		
Input:	Gantt Chart from Section 5.2.1		
Dependencies:	Gantt Chart from Section 5.2.1 and updated timelines for specified tasks		
Risks:	Meeting the project deadline		

WBS #:	5.3.1	Task:	Design Documents
Est. Level of Effort:	3 Hours	Owner:	Mehar
Resources Needed:	Microsoft Word	Work Products:	Word Documents
Description of Task:	Create documents that outline the technical specifications of each design in the project i.e. Board, LED Grid, etc in order to create a reference for each subcomponent		
Input:	Template		
Dependencies:	Detailed Design completed		
Risks:			

WBS #:	5.3.2	Task:	Update Design Documents
Est. Level of Effort:	3 Hours	Owner:	Mehar
Resources Needed:	Microsoft Word	Work Products:	Word Documents
Description of Task:	Update documents as technical specifications of each design is modified and changed to accommodate future designs.		
Input:	Design Documents from Section 5.3.1		
Dependencies:	Design Documents Completed		
Risks:	Incompatibility with some aspects of the board and project with modified components.		

18. Requirements Traceability Matrix

	SUB-SYSTEMS	Localization	Path Planning	Mobility	Computer Vision	Game Board	User Interface
	SATISFIED REQUIREMENTS						
FUNCTIONAL REQUIREMENTS	R1.1		✓				✓
	R1.2						✓
	R1.3						✓
	R2.1				✓	✓	
	R2.2	✓		✓		✓	✓
	R2.3	✓		✓	✓	✓	✓
	R2.4	✓				✓	
	R3.1						✓
	R3.2						✓
	R3.3				✓		✓
	R3.4						✓
	R4.1					✓	✓
	R4.2					✓	✓
	R5.1					✓	
	R5.2					✓	
	R5.3					✓	
	R5.4					✓	
	R6.1	✓			✓		
	R6.2	✓	✓	✓	✓		
	R6.3	✓			✓		
	R6.4	✓		✓	✓		
	R6.5			✓	✓		
	R6.6			✓	✓	✓	
	R7.1		✓	✓			
	R7.2			✓			
	R7.3					✓	
	R7.4			✓	✓		
	R7.5			✓		✓	✓
	R7.6	✓	✓	✓			
	R8.1		✓	✓	✓		
	R8.2						✓

	R8.3					✓	
	R8.4				✓		✓
	R8.5	✓			✓		
	R9.1						✓
	R9.2				✓		✓
	R9.3						✓
	R9.4						✓
	R10.1	✓	✓				
	R10.2		✓		✓		
NONFUNCTIONAL REQUIREMENTS	R1.4						✓
	R1.5						✓
	R1.6						✓
	R2.6						✓
	R2.7						✓
	R3.5			✓		✓	✓
	R3.6			✓		✓	✓
	R3.7			✓		✓	✓
	R4.3					✓	
	R4.4			✓		✓	
	R4.5			✓		✓	✓
	R6.8			✓	✓		
	R7.7		✓	✓			
	R7.8			✓			
	R8.6			✓			✓
	R8.7		✓	✓	✓		
	R8.8				✓		✓
	R8.9				✓	✓	
	R9.5			✓		✓	✓
	R9.8						✓
	R9.9				✓		
	R9.10				✓		
	R10.3	✓	✓		✓		
NONFUNCTIONAL CONSTRAINTS	R2.5		✓	✓			✓
	R5.5					✓	
	R9.6			✓		✓	✓
	R9.7			✓		✓	

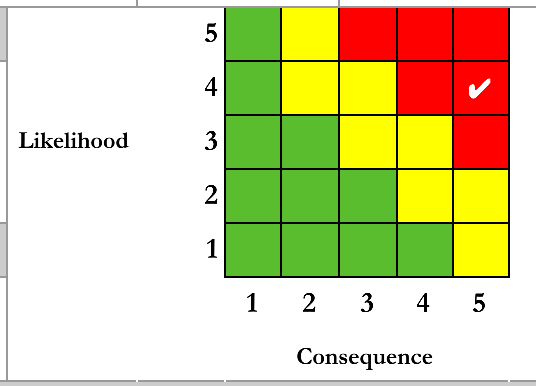
19. Risk Management Plans

MOBILE ROBOTS

Risk Title	Risk Owner	Date Submitted	Date Updated
Ping Sensor Tuning	Nathalie	3/6/18	

Description of Risk	Risk Type
PID tuning of ping sensors can depend on lighting of the environment.	- Technical - Programmatic

Consequence if Risk is Realized
Robots would fail to sense the walls and thus collide into walls.

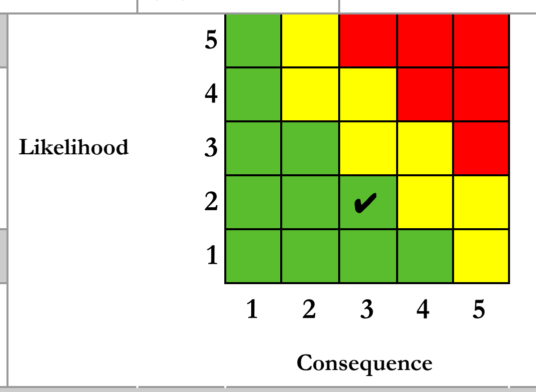


Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Test the sensors where the system will be displayed.	3/14/18 onward	Minimal tweaking of parameters gives same results.	70%	
2. Prepare a separate lighting subsystem.	3/19/18	The light intensity of the subsystem is consistent.	30%	Conducted if Action 1 fails.

Risk Title	Risk Owner	Date Submitted	Date Updated
3D printed pieces might not fit	Esther Lim	3/6/18	

Description of Risk	Risk Type
Robot casings don't fit together nicely	- Technical

Consequence if Risk is Realized
Robots might be too big for the maze. Robots don't hold together the electronics, so high chance of electronics breaking.



Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Create complete CAD model for casings	3/14/18 onward	Everything fits together in the model and once printed	70%	
2. Re-do casings based on physical print	3/19/18	Adjusted casing such that they now fit	30%	Conducted if Action 1 fails.

Risk Title	Risk Owner	Date Submitted	Date Updated
Might notice each individual LED on top of robot	Nathalie Domingo	3/6/18	

Description of Risk	Risk Type
LEDs are not grouped together correctly and not every part of the robot casing is lit.	- Technical

Likelihood	1	2	3	4	5
5					
4		✔			
3					
2					
1					

Consequence

Consequence if Risk is Realized
Users might not understand the animations

Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1.Instal diffusers onto the casings	3/14/18	Can no longer see individual LEDs	10%	

Risk Title	Risk Owner	Date Submitted	Date Updated
LEDs on robot break during operation	Nathalie Domingo	3/6/18	

Description of Risk	Risk Type
LEDs become disconnected or burn out while being used.	- Technical

Likelihood	1	2	3	4	5
5					
4					
3				✔	
2					
1					

Consequence

Consequence if Risk is Realized
No longer have animations, hard to tell the Pac-Man characters. Have to take apart the robots and redo the LED circuit.

Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Conceal exposed wire with electrical tape	4/9/18	No short circuits during testing	20%	
2. Tape down all wires to the top casing	4/9/18	No wires come loose during testing	80%	

Risk Title	Risk Owner	Date Submitted	Date Updated
Unreliable ping sensor readings	Nathalie Domingo	3/6/18	

Description of Risk	Risk Type
Due to large gaps in the maze and the sensors not always being parallel to a wall, sensor readings may be distorted.	- Technical - Programmatic

Consequence if Risk is Realized
Robot will not drive straight in the maze. Might hit walls, hard for user to control and to chase Pac-Man

Likelihood	1	2	3	4	5
5					
4					
3					✔
2					
1					

Consequence

Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Ignore distorted sensor readings	4/12/18	Robot moves relatively straight	20%	
2. Slow down robots	4/12/18 onward	Robot moves straight	80%	Depends on how well the first action works

Risk Title	Risk Owner	Date Submitted	Date Updated
Chosen joystick doesn't work with system	Nathalie Domingo	3/6/18	

Description of Risk	Risk Type
Joystick is meant to work with the Raspberry Pi, so might not work with the Jetson. GPIO pins on Jetson might be more difficult to work with, not worth the trouble	- Technical

Consequence if Risk is Realized
User won't be able to control the ghost robot with a joystick, detracts from the classic aspect of our design

Likelihood	1	2	3	4	5
5					
4					
3					
2		✔			
1					

Consequence

Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Use Jetson GPIO pins, should be similar to RPI	4/1/18	Can control ghost through the joystick	20%	
2. Use computer to control ghost	4/12/18 onward	Can control ghost through the joystick	80%	Only implement if first action doesn't work

Risk Title	Risk Owner	Date Submitted	Date Updated
Joystick control not intuitive	Nathalie Domingo	3/6/18	

Description of Risk	Risk Type
Joystick use might not be as easy to use, depends on joystick configurations	- Technical

Likelihood	1	2	3	4	5
5					
4					
3					
2		✔			
1					

Consequence

Consequence if Risk is Realized
User won't be able to control the ghost robot with a joystick, detracts from the classic aspect of our design

Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Set up joystick so it is intuitive--look at how other joysticks work	4/14/18	Can control ghost through the joystick	20%	
2. Use computer to control ghost	4/14/18 onward	Can control ghost through the joystick	80%	Only implement if first action doesn't work

Risk Title	Risk Owner	Date Submitted	Date Updated
LED Disconnection	Nathalie Domingo	3/7/18	

Description of Risk	Risk Type
Short circuit or soldering fail can cause disconnections of the LEDs under the board.	- Technical

Likelihood	1	2	3	4	5
5					
4				✔	
3					
2					
1					

Consequence

Consequence if Risk is Realized
Some LEDs will not turn on, which can obscure the game rule and confuse the user.

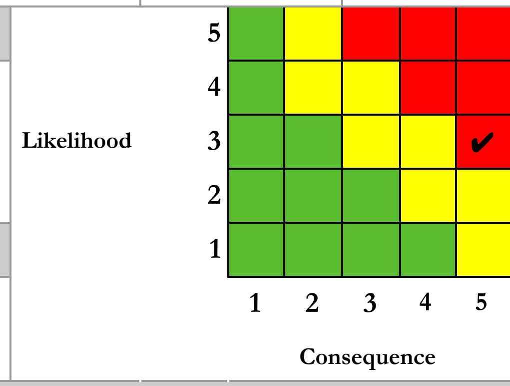
Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Conceal the exposed wires with tape to prevent short circuit.	3/12/18	No short circuit occurs during testing.	20%	
2. Tape the soldered wires together so that the wires stay connected.	3/12/18	No LED disconnections during testing.	80%	

Risk Title	Risk Owner	Date Submitted	Date Updated
Robot Tracking Errors	Angel	3/5/18	

Description of Risk	Risk Type
Robot tracking could issue incorrect coordinates of the robot positions.	- Programmatic

Consequence if Risk is Realized
The game becomes unplayable and the robots could collide with each and/or the walls.

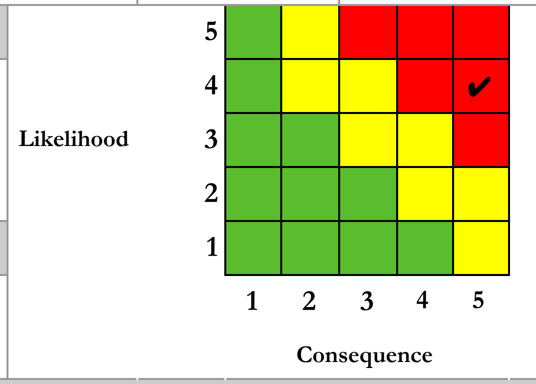


Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Affix camera in a solid and stable position.	3/9/18	Camera feed is extremely stable.	10%	
2. Test under various lighting conditions - find the best range.	3/22/18	Detection of the robots is consistent.	40%	Specifically this is for detecting the robots at all - not necessarily ensuring correctness.
3. Extensively test the code for bugs and best parameters.	3/24/18	Detection of the correct robot positions is consistent.	50%	This milestone is to ensure total correctness.

Risk Title	Risk Owner	Date Submitted	Date Updated
Path Planning/Navigation Errors	Angel, Nathalie	3/5/18	

Description of Risk	Risk Type
The planning might give optimal commands too often, or give incorrect directions. Further, the directions could possibly be followed incorrectly by the robot control code.	- Programmatic

Consequence if Risk is Realized
The game is either too difficult/easy for a person to win against, or the game could be utterly broken with bad commands.



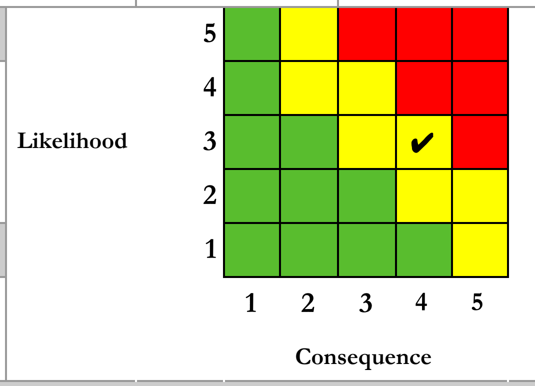
Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Test individual commands from planner to the control.	3/29/18	The Pac-Man robot receives and responds correctly to all commands.	40%	
2. Test multiple map configurations with the planner.	3/29/18	The Pac-Man robot properly navigates the map and optimizes its score.	40%	
3. Test planner with different utility function/edge costs.	3/31/18	The player is able to catch the Pac-Man robot under certain conditions with a certain chance.	20%	

GAME BOARD

Risk Title	Risk Owner	Date Submitted	Date Updated
Hole Mismatch	Esther, Mehar	3/5/18	

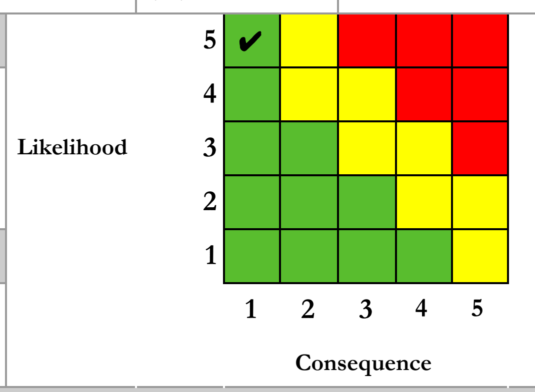
Description of Risk	Risk Type
The holes on the board and the pegs on the walls may not match completely.	- Technical - Cost
Consequence if Risk is Realized	
May have to redo the machining of either the board base or the walls, which requires extra plywood and time.	



Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Triple check the distances between holes before drilling..	2/23/18	The holes on the board matches with the pegs on the walls..	20%	

Risk Title	Risk Owner	Date Submitted	Date Updated
Paint Peeling	Esther, Mehar	3/5/18	

Description of Risk	Risk Type
The paints on the board pieces peel off.	- Technical - Cost
Consequence if Risk is Realized	
The board will not look nice.	

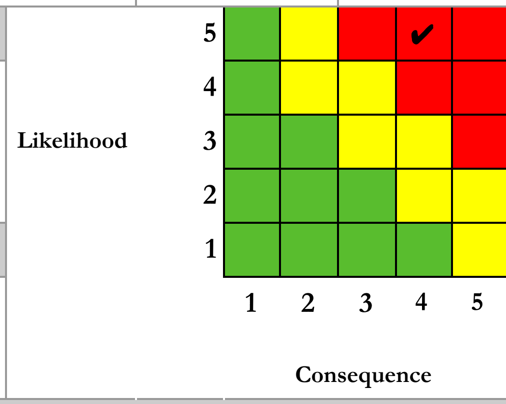


Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Repaint the board pieces when system is complete.	5/1/18	All the peeled off parts are covered by new paint.	50%	
2. Store the board in isolated area.	2/23/18 onward	No peeling of paint due to collision with other objects.	70%	

Risk Title	Risk Owner	Date Submitted	Date Updated
Board Base Pieces Mismatch	Esther, Mehar	3/5/18	

Description of Risk	Risk Type
Due to inaccurate machining of the board base pieces, they do not mesh together when assembled.	- Technical - Cost

Consequence if Risk is Realized
If the connections between two pieces are not level or there is a gap between them, the robot wheels may get stuck in the gap or not be able to drive over the unlevel ground.



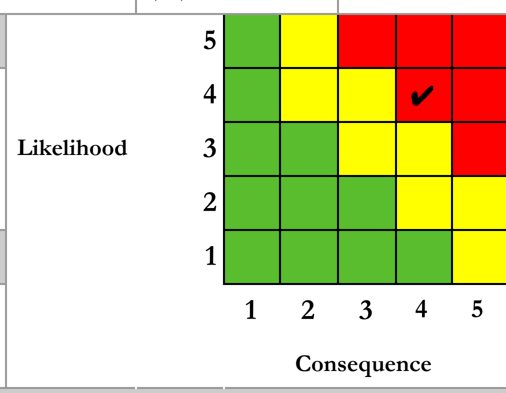
Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Triple check the dimensions before cutting out the pieces.	1/15/18	The dimensions are precise to tight tolerance.	40%	
2. Use precise tools to cut the pieces.	1/15/18	The board base pieces mesh well together.	40%	

Risk Title	Risk Owner	Date Submitted	Date Updated
LED Disconnection	Nathalie Domingo	3/7/18	

Description of Risk	Risk Type
Short circuit or soldering fail can cause disconnections of the LEDs under the board.	- Technical

Consequence if Risk is Realized
Some LEDs will not turn on, which can obscure the game rule and confuse the user.



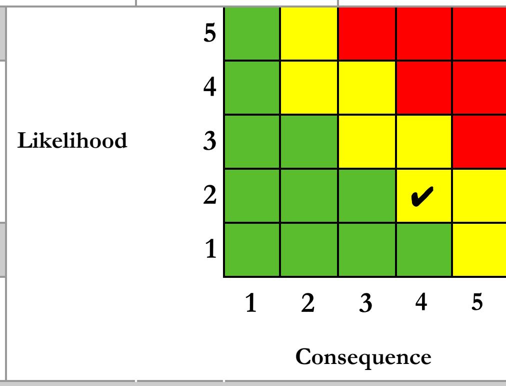
Risk Reduction Plan Summary

Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Conceal the exposed wires with tape to prevent short circuit.	3/12/18	No short circuit occurs during testing.	20%	
2. Tape the soldered wires together so that the wires stay connected.	3/12/18	No LED disconnections during testing.	80%	

Risk Title	Risk Owner	Date Submitted	Date Updated
Wrong Sequence of LED Grid Turning Off	Nathalie	3/5/18	

Description of Risk	Risk Type
The LEDs under the board do not turn off in the correct sequence in correspondence with Pac-Man’s movement..	- Technical

Consequence if Risk is Realized
The game rule will not be enforced correctly and can confuse the player.

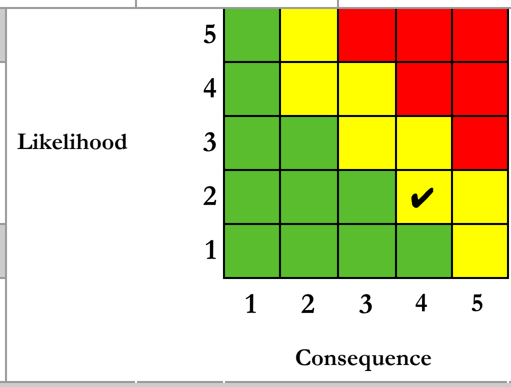


Risk Reduction Plan Summary

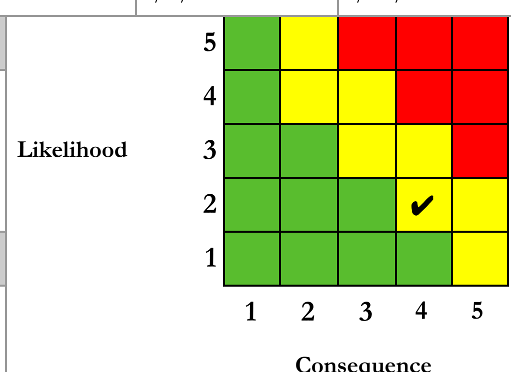
Action/Milestone	Date	Success Criteria	Risk Level	Comments
1. Ensure correct code.	3/5/18	Code sends correct commands.	10%	
2. Ensure LED connections are correct.	2/4/18 - 2/24/18	All LEDs turn on.	40%	
3. Ensure all LEDs are soldered properly.	2/4/18 - 2/24/18	All LEDs turn on.	40%	

USER INTERFACE

Risk Title		Risk Owner	Date Submitted	Date Updated
User Interface is not Intuitive		Esther, Mehar, Nathalie	3/5/18	4/12/18
Description of Risk		**Risk Type**		
The user interface (UI) may not be intuitive and potential users may be confused and unable to play or understand the game		- Technical - Content		
Consequence if Risk is Realized				
May have to re-create educational content and format of the user interface to make it more intuitive to navigate and understand				
Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Conduct surveys to determine benefit of educational content and intuitive-ness of the UI	4/23/18	The UI is deemed intuitive and increases understanding of path-planning by a reasonable amount	30%	
2. Re-iterate through the UI and conduct surveys again	4/30/18	The Pac-Man robot properly navigates the map and optimizes its score.	20%	



Risk Title		Risk Owner	Date Submitted	Date Updated
Representative Audience for Survey		Mehar	3/5/18	4/12/18
Description of Risk		**Risk Type**		
The surveys conducted may not include a wide-variety of audience of various backgrounds and various understanding levels		- Technical		
Consequence if Risk is Realized				
The user interface may be tailored to a specific group of people rather than generalizable				
Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Conduct surveys with college students of different background and children at CMOP	4/23/18	Obtained a wide variety of groups to partake in the survey	30%	

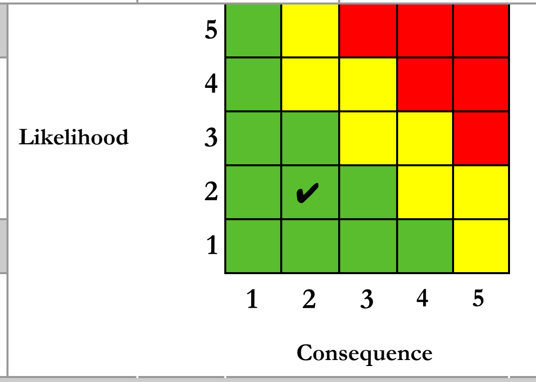


COMMUNICATION

Risk Title	Risk Owner	Date Submitted	Date Updated
Incompatibility with Jetson	Esther, Mehar, Nathalie	3/5/18	4/12/18

Description of Risk	Risk Type
The joystick may be incompatible with the jetson	- Technical

Consequence if Risk is Realized
May have to order a joystick that is compatible with the Jetson. Could set timeline back by a couple of days

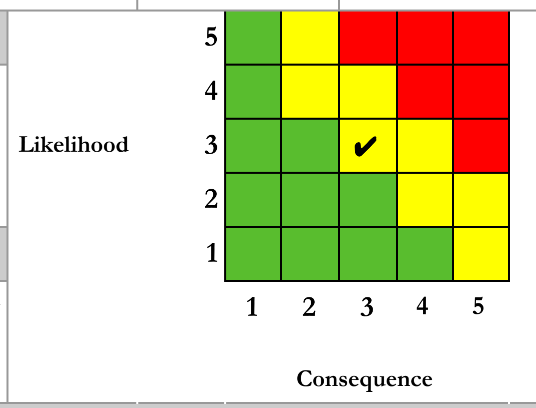


Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Test connectivity of Joystick with Jetson. Otherwise buy another joystick	4/1/18	The joystick successfully connects to the Jetson	20%	

Risk Title	Risk Owner	Date Submitted	Date Updated
High Latency	Esther, Mehar, Nathalie	3/5/18	4/12/18

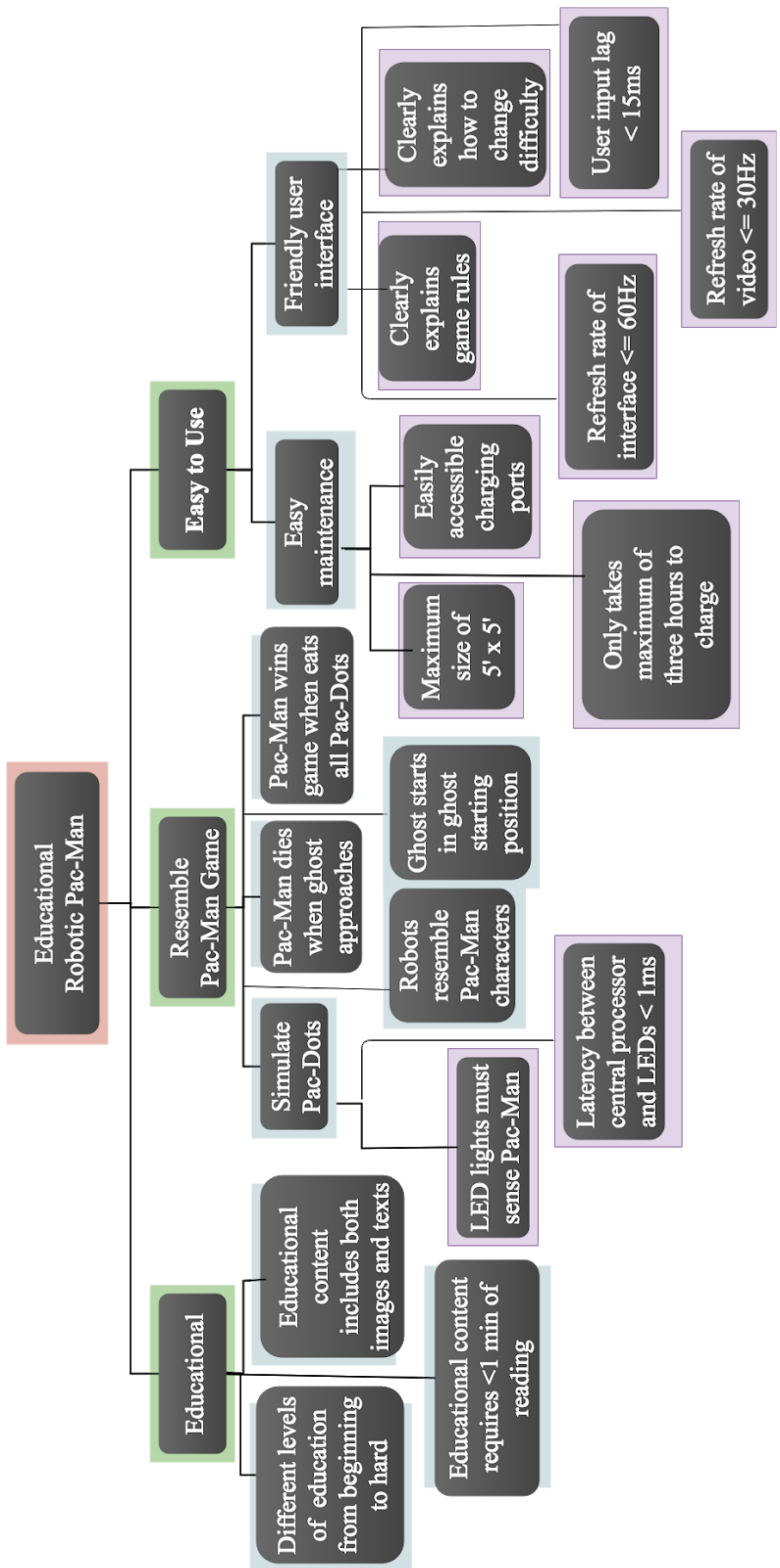
Description of Risk	Risk Type
The user interface (UI) may have a high latency or delay from the actual system.	- Technical

Consequence if Risk is Realized
May have a delay of information from the actual system and the user interface map may not be a completely accurate description of the actual game.



Risk Reduction Plan Summary				
Action/Milestone	**Date**	**Success Criteria**	**Risk Level**	**Comments**
1. Measure latency. If it exceeds our criteria, restructure pipeline to prioritize information sent to the user interface and/or minimize computations per second	4/23/18	There is minimal delay between the user interface, game feed and the actual description	30%	

20. Objectives Tree



21. Test & Evaluation Plan

EDUCATION

Test #1: Education

1. **Purpose**
   Determine whether the education aspects of the user interface are implemented.
2. **What will be tested**
   Different learning levels available, if users learn about path planning, and educational content is well organized.
3. **Requirements**
   R1.1, R1.2, R1.3, R1.4, R1.5, and R1.6
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      The time it takes to go through educational content and the content displayed
   b. **How it will be measured**
      Use a timer and run through game with fully integrated user interface.
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Should only take one minute to go through educational content, different learning levels should be available, and educational content should be easy to understand.
   d. **How measurements will be compared to the ground truth**
      Everything should match ground truth, for timing, can be less than or equal to.
5. **Logistics of testing**
   a. **Timing of the test**
      Week of April 29th
   b. **Where will the test take place**
      FRC High Bay
   c. **Required equipment and supplies**
      Completely built system
   d. **Test personnel and responsibilities**
      Everyone. Minimally one person to time and one person to go through the game.
6. **Post-Test Actions**
   a. **Data analysis**
      Check off a list of contents for the user interface that are required.
   b. **Data reporting**
      Report how many of the required contents are checked off.
   c. **Proposed modifications**
      Change length of content accordingly.

CHARGING

Test #2: System Charging

1. **Purpose**
   Determine how long it takes to charge the entire system.
2. **What will be tested**
   Time it takes for system to be completely charged.
3. **Requirements**
   R4.1 & R4.2
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      Time to charge system.
   b. **How it will be measured**
      Stopwatch
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Maximum of three hours to charge everything. This means three hours to charge the Pac-Man and ghost robots as these are the only components that are not plugged in to an outlet.
   d. **How measurements will be compared to the ground truth**
      Must take less than or equal to ground truth time to charge the entire system.
5. **Logistics of testing**
   a. **Timing of the test**
      Week of April 15th
   b. **Where will the test take place**
      FRC High Bay
   c. **Required equipment and supplies**
      Robots to be fully built
   d. **Test personnel and responsibilities**
      Nathalie
6. **Post-Test Actions**
   a. **Data analysis**
      Add up the time it takes to charge each subsystem.
   b. **Data reporting**
      Total time it took to charge the full system.
   c. **Proposed modifications**
      Replace battery with fast-charging one if necessary.

COMMUNICATION

Test #3: Pac-Man Communication

1. **Purpose**
   Determine whether can communicate to Pac-Man robot.
2. **What will be tested**
   Whether Pac-Man does what is commanded and how long it takes for Pac-Man to carry out the command.
3. **Requirements**
   R8.1 and R8.7
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      Time it takes for Pac-Man to carry out command and whether Pac-Man does right command.
   b. **How it will be measured**
      We know the correct command, so will observe Pac-Man. Timing requirement really small, will just determine by looking at whether it is noticeable.
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Should follow all commands correctly and command latency should not be noticeable.
   d. **How measurements will be compared to the ground truth**
      Should match ground truth.
5. **Logistics of testing**
   a. **Timing of the test**
      Week of April 15th
   b. **Where will the test take place**
      FRC High Bay
   c. **Required equipment and supplies**
      Pac-Man robot, main processing system (Jetson), and board.
   d. **Test personnel and responsibilities**
      Nathalie and Angel
6. **Post-Test Actions**
   a. **Data analysis**
      Analyze the accuracy of Pac-Man's execution of commands.
   b. **Data reporting**
      Report any miscommunications in which Pac-Man fails to follow the command.
   c. **Proposed modifications**
      Implement new method of communication if necessary.

Test #4: Ghost Communication

1. **Purpose**
   Determine whether can communicate to Ghost robot.
2. **What will be tested**
   Whether Ghost does what is commanded and how long it takes for Ghost to carry out the command.
3. **Requirements**
   R8.1, R8.2, and R8.6
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      Time it takes for Ghost to carry out command and whether Ghost does right command.
   b. **How it will be measured**
      We know the correct command, so will observe Ghost. Timing requirement really small, will just determine by looking at whether it is noticeable.
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Should follow all commands correctly and command latency should not be noticeable.
   d. **How measurements will be compared to the ground truth**
      Should match ground truth.
5. **5. Logistics of testing**
   a. **Timing of the test**
      Week of April 15th
   b. **Where will the test take place**
      FRC High Bay
   c. **Required equipment and supplies**
      Ghost robot, main processing system (Jetson), and board.
   d. **Test personnel and responsibilities**
      Nathalie and Angel
6. **Post-Test Actions**
   a. **Data analysis**
      Analyze the accuracy of Ghost's execution of commands.
   b. **Data reporting**
      Report any miscommunications in which Ghost fails to follow the command.
   c. **Proposed modifications**
      Implement new method of communication if necessary.

Test #5: LED Commands

1. **Purpose**
   Determine whether can communicate to LED grid.
2. **What will be tested**
   Whether grid does what is commanded and how long it takes for grid to carry out the command.
3. **Requirements**
   R8.3 and R8.9
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      Time it takes for LED grid to carry out command and whether LED grid does right command.
   b. **How it will be measured**
      We know the correct command, so will observe LED grid. Timing requirement really small, will just determine by looking at whether it is noticeable.
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Should follow all commands correctly and command latency should not be noticeable.
   d. **How measurements will be compared to the ground truth**
      Should match ground truth.
5. **Logistics of testing**
   a. **Timing of the test**
      Week of April 15th
   b. **Where will the test take place**
      FRC High Bay
   c. **Required equipment and supplies**
      LED grid and main processing system (Jetson).
   d. **Test personnel and responsibilities**
      Nathalie and Angel
6. **Post-Test Actions**
   a. **Data analysis**
      Percentage of LEDs turned off at the right command.
   b. **Data reporting**
      Report whether all commands executed correctly.
   c. **Proposed modifications**
      Incorrect execution most likely due to bug with code. Debug code as necessary.

SENSING

Test #6: PID Tuning of Ping Sensors

1. **Purpose**
   To ensure that the ping sensors on the robots are tuned correctly so that the robots move straight along the paths and not collide into walls or other robots.
2. **What will be tested**
   The readings from four ping sensors located on the front, back, left and right of Pac-Man and Ghost robots will be tested.
3. **Requirements**
   R6.2, R6.3, R6.4, R7.1, R7.4, R7.5, R7.6
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      The distance from the robot to the walls will be measured from the ping sensors.
   b. **How it will be measured**
      It will be measured using the ping sensors values and observations to ensure the robots travel in a straight line
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Distance between the ping sensors and the wall when the robot is placed at the center of the pathway. About 1 inch.
   d. **How measurements will be compared to the ground truth**
      The reading from the ping sensor should be no greater than delta value of 0.5 inch from the ground truth.
5. **Logistics of testing**
   a. **Timing of the test**
      Week of April 9
   b. **Where will the test take place**
      FRS High Bay
   c. **Required equipment and supplies**
      Board, Pac-Man and Ghost robots, computer
   d. **Test personnel and responsibilities**
      Nathalie
6. **Post-test actions**
   a. **Data analysis**
      Move the robots around the whole maze and visually inspect that they don't collide into walls or each other.
   b. **Data reporting**
      Report number of collisions in one round of game.
   c. **Proposed modifications**
      PID control using Computer Vision

MAINTENANCE

Test #7: Maintenance of Robots

1. **Purpose**
   To ensure that the maintenance of the robots is simple.
2. **What will be tested**
   Easiness of charging battery and replacing electrical and mechanical components such as feather board, wheels, motors, etc.
3. **Requirements**
   R3.2, R3.4 (simple and short maintenance)
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      Time it takes to replace broken components.
   b. **How it will be measured**
      Measuring time it takes to replace small components like motors, wheels, battery, etc.
   c. **How ground truth will be obtained (could simply be benchmarks)**
      10 minutes.
   d. **How measurements will be compared to the ground truth**
      Maintenance time should be less than 10 minutes.
5. **Logistics of testing**
   a. **Timing of the test**
      First week of May (when the whole system is completed)
   b. **Where will the test take place**
      FRS High Bay
   c. **Required equipment and supplies**
      Pac-Man, Ghost, screwdrivers, new replacing components
   d. **Test personnel and responsibilities**
      Mehar, Esther
6. **Post-Test Actions**
   a. **Data analysis**
      Analyze time data collected to see if average time exceeds 10 minutes.
   b. **Data reporting**
      Report any instances in which time exceeds 10 minutes.
   c. **Proposed modifications**
      Change layout of the internal components if necessary.

BOARD

Test #8: Board Assembly

1. **Purpose**
   To ensure that the assembling of the board can be done in reasonable amount of time.
2. **What will be tested**
   Easiness of assembling.
3. **Requirements**
   R3.4, R5.1
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      Time it takes to assemble the board.
   b. **How it will be measured**
      Measure how long it takes to assemble the board starting from disassembled state.
   c. **How ground truth will be obtained (could simply be benchmarks)**
      15 minutes.
   d. **How measurements will be compared to the ground truth**
      Maintenance time should be less than 15 minutes.
5. **Logistics of testing**
   a. **Timing of the test**
      First week of May (when the whole system is completed)
   b. **Where will the test take place**
      FRS High Bay
   c. **Required equipment and supplies**
      Board, walls
   d. **Test personnel and responsibilities**
      Mehar, Esther
6. **Post-Test Actions**
   a. **Data analysis**
      Analyze time data collected to see if average time exceeds 15 minutes.
   b. **Data reporting**
      Report any instances in which time exceeds 15 minutes.
   c. **Proposed modifications**
      Make wires under the board detachable if they get in the way.

Test #9: Board Modularity

1. **Purpose**
   To ensure that the board is portable as well as configurable.
2. **What will be tested**
   Assembling and Disassembling the Board
   Various Map Configurations
   Safety of Electronics
3. **Requirements**
   R3.2, R5.1, R5.2, R5.3
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      How easily the board is configured and transported.
   b. **How it will be measured**
      Disassemble the board, transport it to a different floor, and re-assemble it.
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Transportation should not be hindered by narrow hallways or doors.
5. **Logistics of testing**
   a. **Timing of the test**
      Last week of February - when the manufacturing of the board is completed
   b. **Where will the test take place**
      FRS High Bay
   c. **Required equipment and supplies**
      Board, walls
   d. **Test personnel and responsibilities**
      Mehar, Esther
6. **Post-Test Actions**
   a. **Data analysis**
      Analyze instances in which transporting was impossible such as the door being too small for parts of the board to fit through.
   b. **Data reporting**
      Report any instances in which transporting was impossible such as the door being too small for parts of the board to fit through.
   c. **Proposed modifications**
      Reduce the size of each module if necessary.

USER INTERFACE

Test #10: User Interface Content

1. **Purpose**
   To ensure that the user interface is intuitive and clearly explains the rules of the game and offers multiple difficulty levels.
2. **What will be tested**
   Educational Content
   User Interface Main Page
3. **Requirements**
   R9.1, R9.2, R9.3, R9.4
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      % Increase in Understand of Path-Planning
   b. **How it will be measured**
      Surveys and Questionnaires
   c. **How ground truth will be obtained (could simply be benchmarks)**
      One group will be presented without the educational content to set our benchmark.
   d. **How measurements will be compared to the ground truth**
      % of Questions answered and correct vs. Benchmark group
5. **Logistics of testing**
   a. **Timing of the test**
      4/23/18
   b. **Where will the test take place**
      FRS High Bay
   c. **Required equipment and supplies**
      Board, User Interface, Robots
   d. **Test personnel and responsibilities**
      Mehar
6. **Post-test actions**
   a. **Data analysis**
      We will test our data to determine whether providing the educational content significantly helped the understand of path-planning for users and observe if the result was statistically significant.
   b. **Data reporting**
      Report instances in which the user's understanding of the content did not improve.
   c. **Proposed modifications**
      Add more details to and clarify the parts that users had most trouble understanding.

Test #11: User Interface Aesthetic

1. **Purpose**
   To ensure that the user interface is aesthetically pleasing and matches the style of the traditional Pac-Man game.
2. **What will be tested**
   User Interface
3. **Requirements**
   R9.4, R9.5, R9.6
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      N/A
   b. **How it will be measured**
      N/A
   c. **How ground truth will be obtained (could simply be benchmarks)**
      Visual Inspection
   d. **How measurements will be compared to the ground truth**
      Visual Inspection
5. **Logistics of testing**
   a. **Timing of the test**
      4/23/18
   b. **Where will the test take place**
      FRS High Bay
   c. **Required equipment and supplies**
      User Interface
   d. **Test personnel and responsibilities**
      Mehar, Nathalie
6. **Post-Test Actions**
   a. **Data analysis**
      N/A
   b. **Data reporting**
      N/A
   c. **Proposed modifications**
      Change color scheme or add more images if necessary.

PATH PLANNING

Test #12: Pacman AI - Full Map Coverage

1. **Purpose**
   To ensure that the Pac-Man robot can create a plan that allows it to autonomously navigate through the entire map and clear all the "dots".
2. **What will be tested**
   Proper execution of movement commands, and real-world location of the robot.
3. **Requirements**
   R2.1, R2.2, R2.3, R2.4, R8.1, R10.1, R10.2, R10.3
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      Coverage of the board by the Pac-Man robot.
   b. **How it will be measured**
      Internal representation of map coverage will be tracked by the central processor.
   c. **How ground truth will be obtained**
      Ground truth is the board itself.
   d. **How measurements will be compared to the ground truth**
      Physically watching how the Pac-Man robot covers the maze.
5. **Logistics of testing**
   a. **Timing of the test**
      Week of April 17
   b. **Where will the test take place**
      FRC High Bay
   c. **Required equipment and supplies**
      Board, Pac-Man and Ghost robots, computer, camera system, router
   d. **Test personnel and responsibilities**
      Angel
6. **Post-Test Actions**
   a. **Data analysis**
      Analyze instances of game playing to ensure Pac-Man always covers the board.
   b. **Data reporting**
      Report instances in which Pac-Man fails to cover the board.
   c. **Proposed modifications**
      Change heuristics function of the multi-goal A* algorithm, debug the path planning code or use alternative path planning algorithm as necessary.

GAME

Test #13: Pacman AI - Different Difficulty Settings

1. **Purpose**
   To ensure that the Pac-Man robot can create a plan that allows it to autonomously navigate through the maze while also allowing for vulnerabilities in that path that allow it to potentially lose.
2. **What will be tested**
   Proper execution of movement commands, and real-world location of the Pacman and ghost robots. Also, the ability for the user to win
3. **Requirements**
   R2.1, R2.2, R2.3, R2.4, R2.5, R8.1, R10.1, R10.2
4. **Data collection and evaluation procedure**
   a. **What will be measured**
      The ratio of win/loss for the user controlling the ghost robot.
   b. **How it will be measured**
      Running the game multiple times under varying conditions in the AI decision making process and gathering the number of wins/losses.
   c. **How ground truth will be obtained**
      Ground truth is the base level AI that is normally uncatchable. In this case, it acts more as a control, than a ground truth, however.
   d. **How measurements will be compared to the ground truth**
      Physically watching and notating how often people win against the Pacman robot with various AI settings.
5. **Logistics of testing**
   a. **Timing of the test**
      Week of April 17
   b. **Where will the test take place**
      FRC High Bay
   c. **Required equipment and supplies**
      Board, Pac-Man and Ghost robots, computer, camera system, router, Player
   d. **Test personnel and responsibilities**
      Angel, Nathalie, Mehar, Esther
6. **Post-Test Actions**
   a. **Data Analysis**
      Win/loss data will be analyzed against people's ages.
   b. **Data Reporting**
      Report cases in which win/loss ratio was zero.
   c. **Proposed Modifications**
      Adjust AI to be as difficult as the various selectable difficulties.

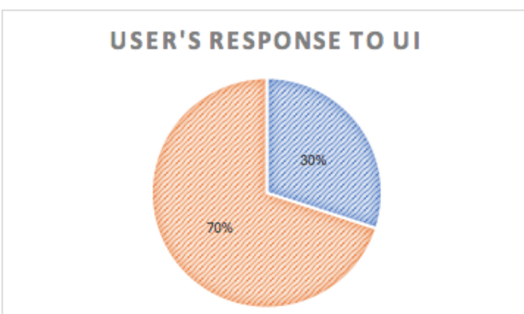
22. Test Results

22.1 User Interface and Educational Content

Tests Satisfied: #1, #10, #11

In order to satisfy our test plans for our user interface requirements, we conducted two tests to determine the user experience and friendliness:

1. **Time Taken:** We measured the time taken for 10 users to read through the user interface. Our goal was to limit the amount of time taken to less than one minute. However, we saw that the it took the users an average time of approximately 45 seconds to read through the instructuctions and the non-technical content and the users approximately 2 minutes to to read through the technical advanced content. The Table below shows the results taken from the 10 users and their time taken to read through the User Interface, Instructions and the Non-Technical and Technical Content.
2. **Aesthetic:** We also asked survey questions about the aesthetic with respect to the original retro Pacman game and received generally great comments. The table and pie chart below shows the users' response of the aesthetic of the interface and the friendliness with respect to the original Pacman game on a scale of 1 (worst) to 5 (best). The pie chart represents the culmination of our results where orange is a score of 5 and blue is a score of 4.



User No.	User Response Score
1	4
2	5
3	5
4	4
5	5
6	4
7	5
8	5
9	5
10	5
Average	**4.7**

User Response to UX and Friendless of UI

User No.	Non-Technical Content	Advanced Technical Content
1	41	138
2	39	125
3	50	105
4	44	112
5	47	160
6	45	110
7	43	118
8	42	112
9	48	107
10	45	121
Average	**44.4**	**120.8**

Time Taken to Read through UI (in seconds)

Unfortunately, due to time constraints, we were unable to conduct surveys to test the the impact of the educational content to increase understanding of path planning.

22.2 Communication with Robots

Tests Satisfied: #3, #4

We also tested the accuracy and reliability of our communication with the robots. We found that the connection was strong the communication with the robots was very robust and the robots followed the commands accurately. We also tried to measure the input lag between the users' input to the response of the robots and found that the lag was unnoticeable or negligible as the robot moved simultaneously as the command was sent

22.3 Communication with LED Grid

Tests Satisfied: #5

We tested the connection of our LED Grid and the accuracy of turning the correct LEDs on and off depending on the position of Pacman. In order to do this, we split up our tests into two phases. Our first step was to send commands to the LED Grid to test our communication to turn on and off certain LEDs based on arbitrary positions. Once we got the functionality and the pipeline of the communication set up, we tested moving the Pacman robot and our Computer Vision algorithm sending commands to the LED Grid to turn off LEDs to represent our game mechanics.

Our results for this were also based on visual inspection and found that the system was accurate and reliable

22.4 Maintenance of Robots

Tests Satisfied: #8

We tested the maintenance time to fix potential issues within robot such as replacing the batteries or replacing the motors and wheels. Our table shows the results we collected for each maintenance category and the time taken to replace them. We set our benchmark in our test plan as 10 minutes. However, we were able to comfortably replace all these components within that time. Our casings that were screwed on made it very simple to unscrew the relevant section and replace the component. The times are represented in minutes and seconds in the table below.

Trial	Replace Battery	Replace Wheels	Replace Motors
1	3:26	1:05	4:12
2	3:39	1:02	4:23
3	3:15	0:59	4:20
4	3:20	0:53	4:38
5	3:31	1:02	4:28

Time Taken to Replace Components

22.5 Path Planning Completeness

Tests Satisfied: #12

We tested the completeness of out path planning algorithm as well as it is crucial as part of the game. Again, we tested this in two separate steps. We tested our algorithm extensively on various map configurations to ensure that Pacman attempts to conduct a full map coverage and eat all of the dots. Our first step was to create various mazes virtually to ensure the path planning algorithm works as desired. We then tested a couple of mazes on the physical game board to ensure the commands are sent properly to Pacman.

22.6 Board Assembly and Modularity

Tests Satisfied: #8, #9

Lastly, we tested our how long it takes to assembly and transport our boards as outlined in our Test Plans. We measured the time taken to assembly our board from the unassembled state. Our goal was to ensure this time was less than 15 minutes. However, we were able to assemble the board within 12 minutes on average from 10 tests from it's unassembled state. In terms of portability, the boards are easily transportable with a trolley to carry the boards as it allows to stack the boards up and transport them. We also tested how long it takes us to configure the mazes in the board. Our goal was to take 5 minutes to set up a certain maze configuration. However, it took us closer to 7.5 minutes to set up each configuration. This was due to tolerances in the holes being too tight so it was slightly harder to fit each wall.

23. Future Work & Commercialization

23.1 Safety

In order to make the system safe, it should be enclosed and there should be strict speed limits on the robot. Currently the board is completely exposed, both in terms of no cover over the top of the system and no cover for the LED circuit underneath the board. In the current state, anyone can move the walls and remove the robots during game play. This is unsafe as the Pac-Man robot will continue to try to move, regardless of interference. One way to make the system safer in this regard would be to place a clear glass cover on top of the board. This would prevent users from interfering with the system while it is on. The board is also currently unsafe due to the exposed circuit under the board. Although this is not necessarily unsafe for users, as they will not be shocked by playing with the circuit, the system itself is in danger of being destroyed. Since there are a lot of exposed wires, it is very easy for a user to pull one of them, thus damaging the LED grid. One way to prevent this would be to enclose the LED grid so there are no wires exposed.

23.2 Reliability

In order to make the system more reliable, better wiring for the LED grid and robot circuits should be done and error correcting code memory implemented. The LED grid was hand soldered and the connections taped over with electrical tape, which leads to the possibility of connections coming loose. If connections come loose, LEDs will not turn on, thus making the LED aspect of the system unreliable. In order to fix this, all the connections should be heat shrinked, and possibly done using solder sleeves. This will ensure that even with some tugging the connections will not come undone, thus making it more reliable. Similarly, with the robot circuits, namely those for the LED animations, the same upgrades can be done to ensure the animations are reliable. Lastly, in order to make the path planning and robot detection more reliable, error correcting code memory can be implemented. This will ensure that the data saved on where the robots are and have been is not corrupt, thus making the path planning more accurate.

23.3 Maintainability

In order to make the system more maintainable, it should be made more modular and it should be easy to access all the parts. Although we made the system modular by being able to take apart the board, with the addition of the LED grid, it is quite difficult to take apart all of the board pieces without risk of breaking some of the LED connections. This can be fixed by having each of the board pieces controlled separately, rather than the current method of central control. Additionally, this would also make it easier to access the LED grid as there would no longer be a large risk in breaking connections when maintaining the board.

23.4 Sustainability

In order to make the system more sustainable, biodegradable components can be used.

23.5 Affordability

In order to make the system more affordable, non-toxic cheap materials can be used, or the entire system can be miniaturized. Cheaper alternatives for the board, robot casings, and camera holder can be found and used. Additionally, making the entire system smaller will make it significantly more affordable.

23.6 Marketability

To make the system marketable, one could highlight the systems game and educational aspects, while using the fact that Pac-Man tends to bring on nostalgia. The fact that the system is a game means it should be marketed as an entertainment system. Additionally, considering that it does have an educational aspect, where it's supposed to teach users about how path planning works and expose them to robotics, it can be marketed as an educational game. Since Pac-Man is a retro arcade game, marketing strategies can utilize nostalgia to make the system more appealing.

23.7 Ethical Issues

Two ethical issues that can be associated with this system is the ability to win and appropriate language. Since this system is meant to be a museum exhibit, there may be some issues with users thinking it is impossible to win the game. In order to minimize this, the game should be tested with users from varying age groups to make sure they don't think the game is too difficult to win. The goal would be to make the game relatively easy to win, but hard enough so users both enjoy the game and consider the path planning that Pac-Man has to do. Similarly, the user interface must contain appropriate language according to the targeted age group.

23.8 Social Issues

One social issue that can be associated with this system is that it only encourages playing. Since it is meant to be in a museum, it should encourage the users to learn something. This can be addressed by making sure the user interface contains valid path planning information, and that it is thought of as an important aspect of the system.

24. Final System Functions

24.1 Board

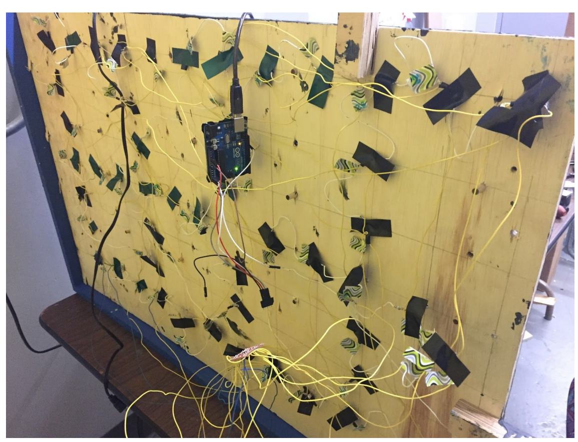
Satisfied Requirements: R5.1, R5.3, R5.4

The board is modular and easy to transport and assemble. The walls are user configurable. The size of the board is 5ft by 5ft. We also created a stand for which the camera to rest on for our Computer Vision code to detect the robots.


24.2 LED Grid

Satisfied Requirements: R2.1, R3.2, R6.5, R7.3

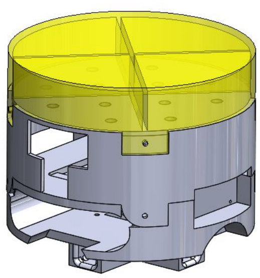
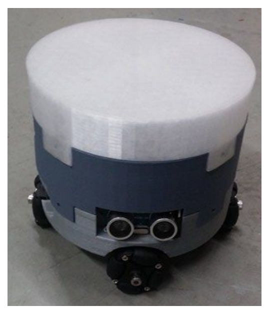
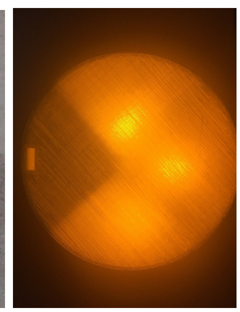
The LED grid can simulate dots being eaten by Pac-Man by turning off each LED when Pac-Man passes by. It does this by receiving commands from the program doing path planning. It sends a location between (0,0) and (1000,1000) which represents where Pac-Man is on the maze. Then, the program controlling the LEDs turns that into the corresponding LEDs. The wires are easily accessible for regular maintenance. The LEDs are flush with the bottom of the maze.



24.3 Pac-Man Robot

Satisfied Requirements: R6.2, R7.2, R7.6, R7.7, R7.8, R10.1, R10.3

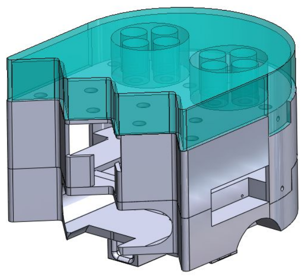
Pac-Man can autonomously navigate within the maze while avoiding the Ghost using path planning algorithm. Pac-Man always chooses the path that optimizes its scores. The omni-wheels make Pac-Man change directions smoothly. Pac-Man has two degree-of-freedom and can move up/down and left/right. The LED animation inside Pac-Man can simulate Pac-Man's mouth opening and closing to eat the dots as it moves. The LED animation is responsive to the direction of movement.

24.4 Ghost Robot

Satisfied Requirements: R2.4, R7.2, R7.8, R8.2

The Ghost always starts from the designated starting region which is located at the center of the board. The omni-wheels make Ghost change directions smoothly. Ghost has two degree-of-freedom and can move up/down and left/right. The LED animation inside Ghost can simulate Ghost's eyes moving up/down, left/right with the direction of movement. User can control the movement of the robot wirelessly.

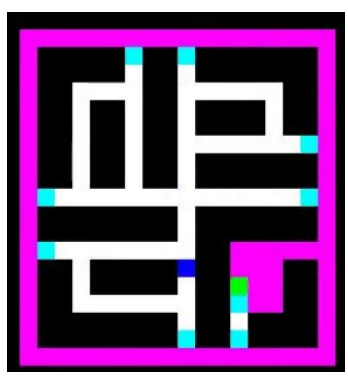


24.5 Path Planning

Satisfied Requirements: R2.5, R6.2, R6.5, R8.1, R8.7, R8.9

Our A.I. and Path Planning is completed and has been deployed onto the Pacman robot. We have created a couple of difficulty levels in our path planning algorithm that the users can choose. We also modified our algorithm so that it gives the users a chance to win - even with the 'hard' difficulty level - which was an issue with an older version of our algorithm. We have also connected and created a pipeline that connects our Computer Vision, LED Grid and robot control all into one smooth functioning pipeline.

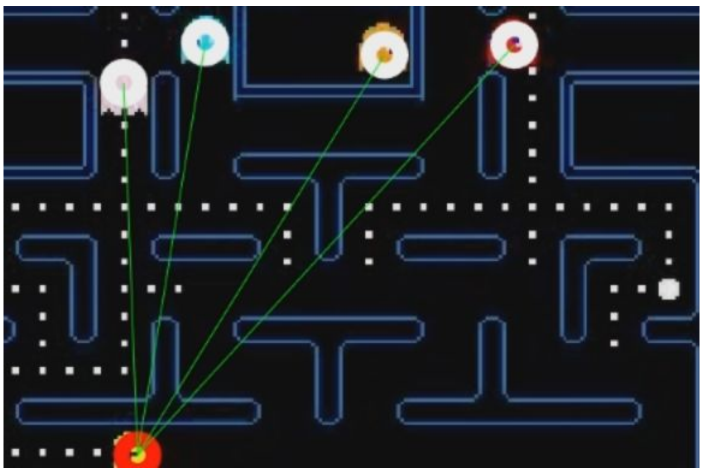
The following screenshot shows how the Pacman AI (the green dot) plans and covers the digitized board map against the ghost (the blue dot). The pink/purple areas are the visited places, white areas are unvisited, and cyan are frontiers.

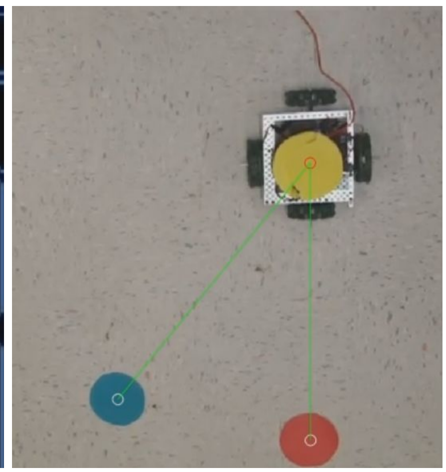


24.6 Computer Vision

Satisfied Requirements: R6.3, R2.1, R5.2

The computer vision system is able to track the locations of the robots on the map and provide coordinates for both. These coordinates can indicate whether the pac-man is too close to the ghost and loses. This subsystem is part of the planning pipeline by providing it the location of the ghost so that the Pac-man robot can avoid it.



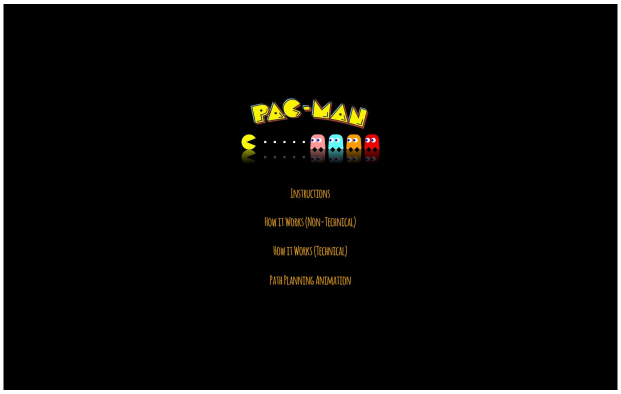


24.7 User Interface

Satisfied Requirements: R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R9.3, R9.4, R9.5, R9.6

The user interface provides a complete lesson on the path planning algorithm of Pac-Man to facilitate user's understanding of the overall concept. It provides technical and non-technical levels of education from beginning to advanced for literate users. The educational content contains both images and texts for enhanced quality. The educational content can be visibly displayed to both the user and the bystanders. The educational content is well organized into main menu and multiple pages for easy interface. The length of the educational content does not exceed 2 minute of reading time in order to balance education and entertainment.

The user interface clearly explains the rule of the game. It has friendly aesthetics and the color scheme matches that of the Pac-Man game.



24.8 Communication

Satisfied Requirements: R8.1, R8.2, R8.3, R8.4, R8.5

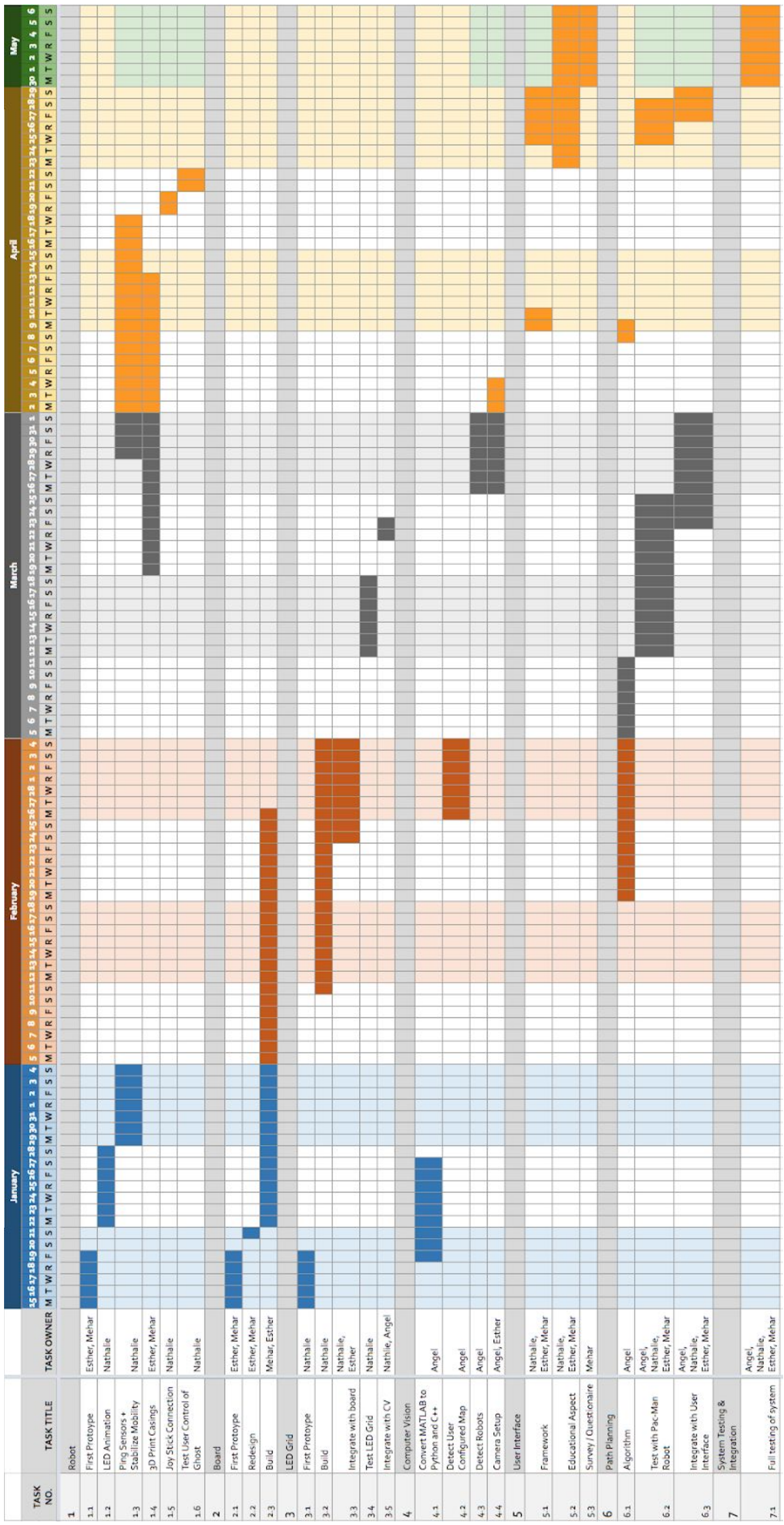
The central processor can send motion commands to the robots. User can communicate movement commands to the Ghost through the central processor. The central processor is able to command the LED array to shut off specific LEDs. The visual feed from a camera overlooking the maze can be communicated to the user. The locations of the center of the robots are communicated to the central processor via camera.

24.9 System Integration

We created a pipeline that allows our Computer Vision and Path Planning systems to interface with one another to send commands to the Pacman robot based on our Computer Vision telling us where the user controlled Ghost robot is. We also created a IP address to communicate between our planner and the LED Grid to instruct the LEDs to turn off once Pacman has driven over it similar to the original game. Our system integration incorporated all of the game mechanics that we set out to. However, we encountered some issues with our ping sensors. The sensors gave our erratic values with a lot of noise that was hard to remove and tune the sensors to ensure the robots go straight. For our final design, we instead utilized our computer vision to create a line following system for the robots to move straight as opposed to the ping sensors.

25. Gantt Chart

Our Gantt Chart for the entire semester is shown in the page below - for easier viewing. It is updated to showcase all the tasks that we conducted over the course of the semester and how long each one took.


January
February
March
April
May
TASK NO.
TASK TITLE
TASK OWNER
1 Robot
1.1 First Protoype Esther, Mehar
1.2 LED Animation Nathalie
1.3 Ping Sensors + Stabilize Mobility Nathalie
1.4 3D Print Casings Esther, Mehar
1.5 Joy Stick Connection Nathalie
1.6 Test User Control of Ghost Nathalie
2 Board
2.1 First Protoype Esther, Mehar
2.2 Redesign Esther, Mehar
2.3 Build Mehar, Esther
3 LED Grid
3.1 First Prototype Nathalie
3.2 Build Nathalie
3.3 Integrate with board Nathalie, Esther
3.4 Test LED Grid Nathalie
3.5 Integrate with CV Nathlie, Angel
4 Computer Vision
4.1 Convert MATLAB to Python and C++ Angel
4.2 Detect User Configured Map Angel
4.3 Detect Robots Angel
4.4 Camera Setup Angel, Esther
5 User Interface
5.1 Framework Nathalie, Esther, Mehar
5.2 Educational Aspect Nathalie, Esther, Mehar
5.3 Survey / Questionaire Mehar
6 Path Planning
6.1 Algorithm Angel
6.2 Test with Pac-Man Robot Angel, Nathalie, Esther, Mehar
6.3 Integrate with User Interface Angel, Nathalie, Esther, Mehar
7 System Testing & Integration
7.1 Full testing of system Angel, Nathalie, Esther, Mehar