

Draft System Design

Group #12 – Botanica
SproutBot

Arun Mistry
Mina Demian
Nicholas Levantis
Usman Minhas

Table 1: Revision History

Date	Developer(s)	Change
December 22, 2023	Arun Mistry, Mina Demian, Nicholas Levantis, & Usman Minhas	Initial Draft System Design

Table of Contents

1. Purpose	5
2. System Overview	5
2.1. Project Scope	5
2.2. Project Constraints	5
3. System Boundary Diagram	6
4. System Component Diagram	7
5. Variables	8
5.1. Notation Overview	8
5.2. Monitored Variables	9
5.3. Controlled Variables	9
5.4. System Constants	10
6. Behaviour Overview	11
7. Components	12
7.1. Robot Movement	12
7.1.1. Purpose	12
7.1.2. Scope	12
7.1.3. Module Guide	12
7.1.4. Scheduling & Timing Constraints	12
7.2. Robot Navigation	13
7.2.1. Purpose	13
7.2.2. Scope	13
7.2.3. Module Guide	13
7.2.4. Scheduling & Timing Constraints	13
7.3. Obstacle Detection & Avoidance	14
7.3.1. Purpose	14
7.3.2. Scope	14
7.3.3. Module Guide	14
7.3.4. Scheduling & Timing Constraints	14
7.4. Watering Arm	15
7.4.1. Purpose	15
7.4.2. Scope	15
7.4.3. Module Guide	15
7.4.4. Scheduling & Timing Constraints	16
7.5. Plant Monitoring	16
7.5.1. Purpose	16
7.5.2. Scope	16
7.5.3. Module Guide	16

7.5.4. Scheduling & Timing Constraints	17
8. Normal Operation	18
9. Undesired Event Handling	18
9.1. No path is available.....	18
9.2. Insufficient water available.....	18
9.3. Foliage in the way	19
9.4. System tips over.....	19
9.5. Loss of communication.....	19
10. Changes	19
10.1 Anticipated Changes	19
10.2 Unlikely Changes	19
11. References.....	19

List of Figures

Figure 1: System Boundary Diagram.....	6
Figure 2: System Components Diagram.....	7
Figure 3: High level connection of modules	11
Figure 4: Robot Navigation Black Box	13
Figure 5: Watering Arm Black Box	15
Figure 6: Soil & Distance Detection System	16
Figure 7: Plant Monitoring Black Box.....	17

List of Tables

Table 1: Notations	8
Table 2: Monitored Variables.....	9
Table 3: Controlled Variables	9
Table 4: System Constants	10

1. Purpose

SproutBot is a plant watering robot that aims to provide a reliable and efficient solution for ensuring the health and well-being of indoor greenery, especially when plant owners are away from home. Its primary objective is to water plants around the house as and when necessary. SproutBot will be considered a success if it is able to move, navigate towards a plant's location, and deliver water to it, repeating this process for other plants as required, and finishing its trip by returning to its base location.

SproutBot is divided into 2 main subsystems, with its movement and navigation system as one, and the watering mechanism as another. These systems will be developed separately and combined in the end. Further details are given under

2. System Overview

2.1. Project Scope

The scope of this project is to have the robot navigate toward plants with minor adaptability (such as stopping & going around obstacles) to various plants around the same height and deliver a specific quantity of water on a schedule basis. As the potential of this project is massive, we plan to limit the scope of the project to features we believe we can implement successfully.

The scope of this project does not include:

- Plant pots of varying heights. The range of plant pot's height will be within [POT HEIGHT RANGE](#).
- Advanced terrain navigation. The robot will not be required to climb any obstacles to navigate towards a plant. Basic movement and obstacle detection, with rudimentary obstacle avoidance will be within scope, and expanded upon as necessary.
- Elevated plants. The robot will only be required to tend to plants located on the floor of the home.
- Plants on different floors of a home. The robot will not be required to travel up and down stairs or ramps.
- Outdoor operation. This robot will be for indoor use only and will not be built resistant to different weather elements such as strong winds and rain or snow.

2.2. Project Constraints

There are 2 main constraints to keep track of when working on this project.

1. Budget. There is a maximum budget of \$750 for the Bill of Materials (BOM) that should not be exceeded.
2. Time. Revision 0 is due by January 26, 2024, which must capture the major components of the project. Revision 1 is due by March 27, 2024, which must build upon Revision 0. The project's ultimate completion date is to be the date of the Final Presentation, March 27, 2024.

3. System Boundary Diagram

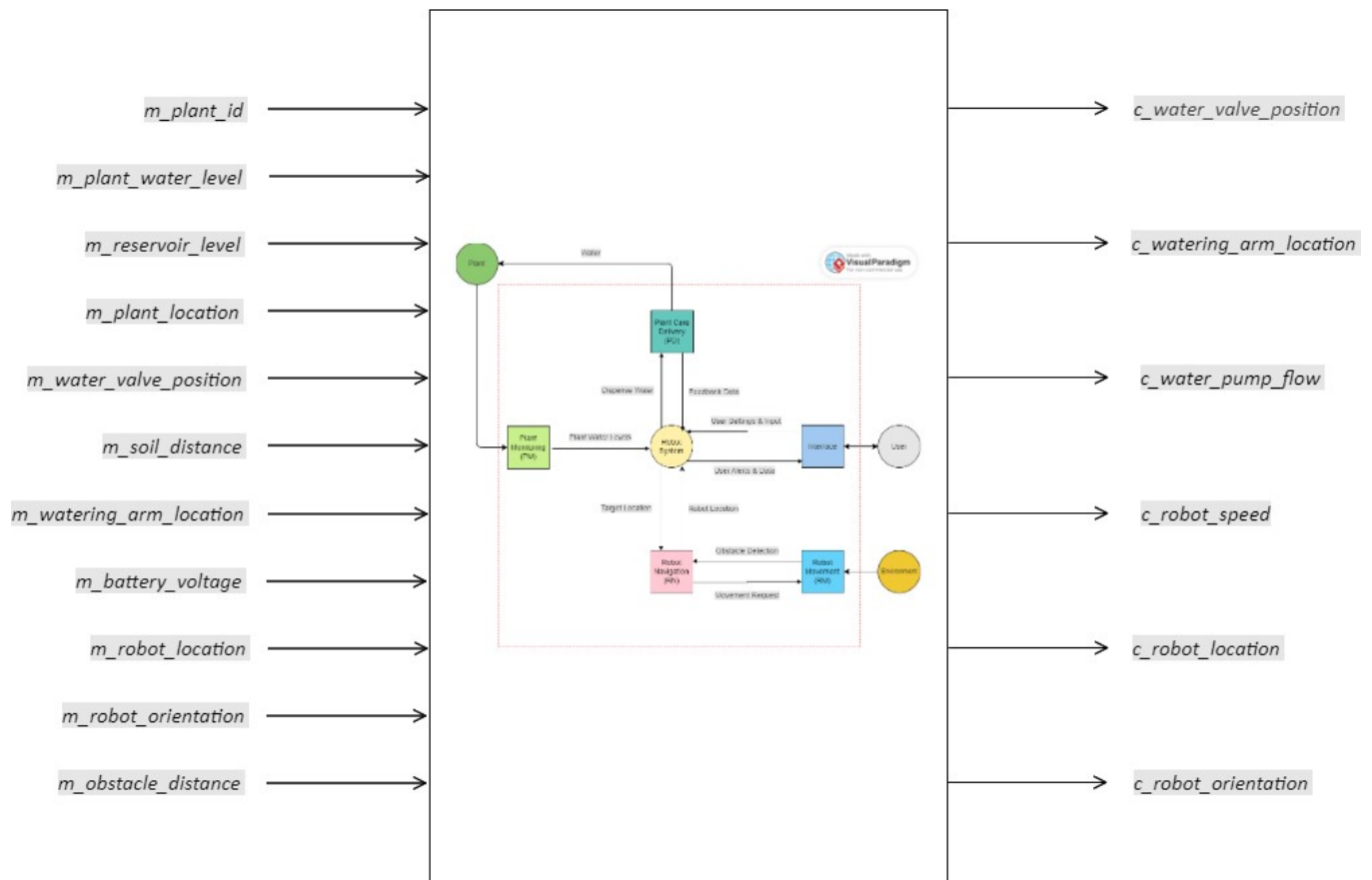


Figure 1: System Boundary Diagram

4. System Component Diagram

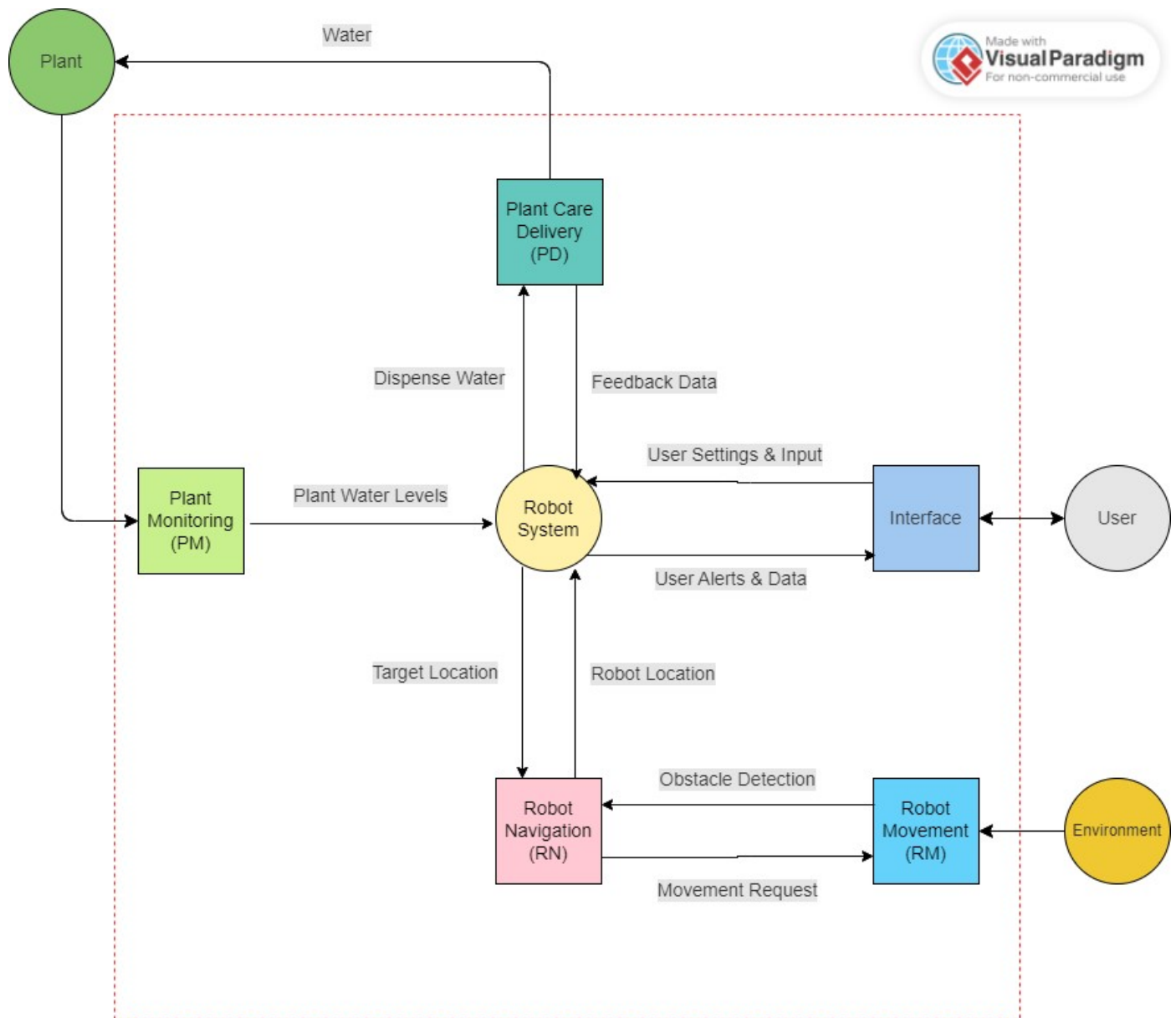


Figure 2: System Components Diagram

5. Variables

5.1. Notation Overview

Notation	Type	Description
m_	Prefix	Denotes monitored variables.
c_	Prefix	Denotes controlled variables.
NAME	Capitalization	Variable names in full uppercase refer to constants.
RM	Acronym	Refers to requirements related to Error! Reference source not found.
RN	Acronym	Refers to requirements related to Error! Reference source not found.
PM	Acronym	Refers to requirements related to Error! Reference source not found.
PD	Acronym	Refers to requirements related to Error! Reference source not found.
E	Acronym	Refers to requirements related to Error! Reference source not found.
H	Acronym	Refers to requirements related to Error! Reference source not found.
SR	Acronym	Refers to Safety Requirements.
SRR	Acronym	Refers to Safety Requirements for Error! Reference source not found.
SRP	Acronym	Refers to Safety Requirements for Error! Reference source not found.
SRE	Acronym	Refers to Safety Requirements for Error! Reference source not found.
SRH	Acronym	Refers to Safety Requirements for Error! Reference source not found.
NF	Acronym	Refers to Error! Reference source not found.

Table 1: Notations

5.2. Monitored Variables

Name	Unit	Description
<i>m_plant_id</i>	-	The plant's identification details in order to control how much water is supplied at specific times.
<i>m_plant_water_level</i>	mL	Measures the moisture level of the plant. Each plant will have its own moisture sensor and value.
<i>m_reservoir_level</i>	mL	Measures the volume of water in the robots on board reservoir.
<i>m_plant_location</i>	Vector in m <x, y, z>	Determines the location of the plant relative to the robot.
<i>m_ir_distance</i>	m	Distance between the IR Emitter at the destination and IR Receivers on the robot
<i>m_water_valve_position</i>	-	Tracks whether the valve for watering mechanism is open or closed.
<i>m_soil_distance</i>	m	Determines if the nozzle of the water mechanism is above soil or not.
<i>m_watering_arm_location</i>	Vector in m <x, y, z>	Location of end effector of robot watering arm, relative to the robot.
<i>m_robot_location</i>	Vector in m <x, y, z>	Location of the robot relative to its environment, such as its initial starting position and the plant.
<i>m_robot_orientation</i>	Vector in degrees < α , β , θ >	The orientation of the robot relative to its starting position.
<i>m_obstacle_distance</i>	m	Location of an obstacle relative to the robot.
<i>m_color_sensor_output</i>	Vector in RGB	The RGB output of the color sensor

Table 2: Monitored Variables

5.3. Controlled Variables

Name	Unit	Description
<i>c_watering_arm_location</i>	Vector in m <x, y, z>	Location of end effector of robot watering arm, relative to the robot.
<i>c_water_pump_flow</i>	mL/s	Speed of the water being pumped.
<i>c_robot_speed</i>	m/s	Overall speed of the robot when it is moving.
<i>c_robot_location</i>	Vector in m <x, y, z>	Location of the robot relative to its environment, such as its initial starting position and the plant.
<i>c_robot_orientation</i>	Vector in degrees < α , β , θ >	The orientation of the robot relative to its starting position.
<i>c_plant_water_needed</i>	mL	The amount of water that each plant needs.

Table 3: Controlled Variables

5.4. System Constants

Name	Unit	Description
<i>NOMINAL_TEMPERATURE</i>	°C	The expected room temperature
<i>NOMINAL_PRESSURE</i>	Atm	The expected air pressure in the room
<i>BATTERY_VOLTAGE</i>	V	The maximum voltage available to supply robot components
<i>MOTOR_TORQUE</i>	Nm	The torque of all motors used for movement will be constant.
<i>POT_HEIGHT_RANGE</i>	m	The range of heights that is allowable for a pot.
<i>RESERVOIR_CAPACITY_MAX</i>	L	The maximum water the on board reservoir can hold.
<i>RESERVOIR_CAPACITY_LOW</i>	L	The threshold for when the water in the reservoir is considered low, for the user to be notified.
<i>PLANT_WATER_CRITICAL</i>	mL	The critical moisture level for any plant that will begin the plant watering process.
<i>MIN_DISTANCE</i>	m	The minimum distance that the robot must leave from all environmental obstacles.
<i>MAX_FORCE</i>	N	The maximum external force that the robot must receive before stopping operations.
<i>MAX_SPEED</i>	m/s	The maximum speed the robot should be allowed to move at.
<i>MAX_ACCELERATION</i>	m/s ²	The maximum acceleration that the robot should be allowed to move at.
<i>BASE_LOCATION</i>	Vector in metre <x, y, z>	The position where the robot stays when it is not required to water plants.
<i>MAX_ATTEMPT_TIME</i>	s	The maximum time the robot can attempt to reach a plant before it must stop.
<i>MAX_RESPONSE_TIME</i>	s	The maximum duration allowable for the user and system's interaction, in either direction.
<i>MAX_RADIUS</i>	m	The maximum radius within which the robot can detect and identify plants to water.
<i>WATER_MONITOR_PERIOD</i>	s	The period to check the amount of water a given plant has.

Table 4: System Constants

6. Behaviour Overview

There are 2 separate systems that work together. System 1 is defined as **Plant**, and holds the Plant Monitoring module. System 2 is defined as **Robot**, and holds the [Robot Navigation](#), [Robot Movement](#), [Obstacle Avoidance](#) and [Watering Arm](#) modules.

The Plant Watering module is responsible for sending a message to the Robot system on the water levels of the plant, upon which the robot activates its Robot Navigation module and Obstacle Avoidance modules. The Robot Navigation module instructs the plant to turn on its IR Emitter, and both modules help Robot Movement in order to navigate to the destination. Once the plant has been reached, the Robot Navigation enables the Watering Arm module to pour water into the plant. This process repeats once the robot goes back to its base station.

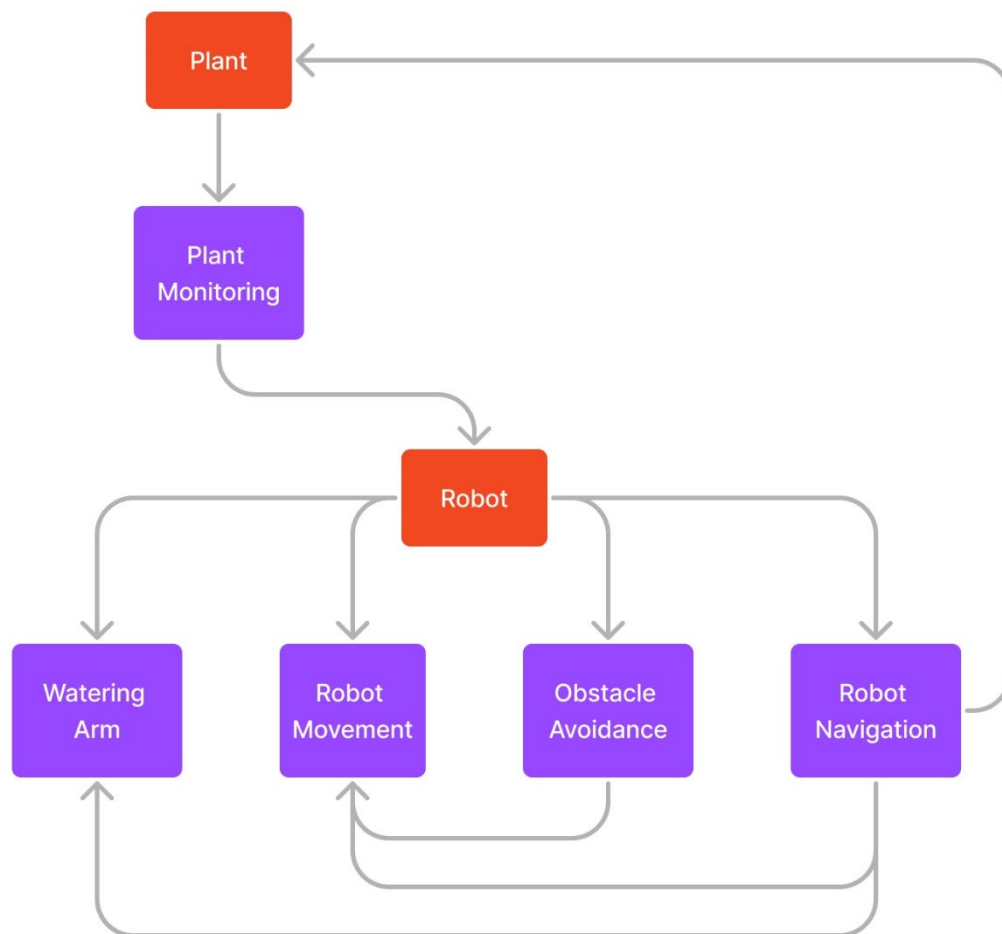


Figure 3: High level connection of modules

7. Components

7.1. Robot Movement

7.1.1. Purpose

This module is responsible for the control of the motors and wheels responsible for the physical movement of the robot, in order to enable it to navigate from the [BASE LOCATION](#) and travel to all the available plants, then return back to the [BASE LOCATION](#).

7.1.2. Scope

The scope of the Robot Movement module is to take the heading and location information determined by the [Robot Navigation](#) module, and use it to control the motors and wheels, in order to navigate to the desired locations accurately.

7.1.3. Module Guide

When the robot receives the signal that it should start moving towards a certain heading, it translates that instruction into the appropriate power for the motors and orientation for the wheels.

7.1.3.1. Module Interface Specifications

There will be 4 motors, one for each wheel of the robot, which will be responsible for supplying each of the wheels with appropriate amount of power to produce the correct linear and rotational movement at the appropriate speed and acceleration to avoid sudden movements and spillages. The wheels of the robot will be big enough to be able to move on surfaces such as hardwood, ceramic, carpets, etc. and not slip or get stuck. The robot is assumed to have arrived once it receives a signal from the [Robot Navigation](#) module that the robot has reached its desired destination (i.e. the distance from the IR Emitters is smaller than a set threshold).

7.1.3.2. Module Internal Design

- Receive the required heading and orientation from the [Robot Navigation](#) module.
- Supply the motors with the appropriate amount of power to allow the wheels to move and produce the correct linear and rotational movement at the appropriate speed and acceleration to avoid sudden movements and spillages.
- Stop the movements once the [Robot Navigation](#) module sends the signal that the robot has reached the desired destination.

7.1.4. Scheduling & Timing Constraints

The Robot Movement module will have some timing constraints, where the robot must reach the desired plant destination in a time lower than the [MAX_ATTEMPT_TIME](#) that we have set.

7.2. Robot Navigation

7.2.1. Purpose

This module is responsible for controlling the robot's heading and ensures that it can appropriately navigate from the [BASE LOCATION](#) and travel to a plant, travel to other plants as necessary, and return to the [BASE LOCATION](#) once all plants have been watered.

7.2.2. Scope

The scope of navigation is limited to providing a heading towards a specific destination for the [Robot Movement](#) module.

7.2.3. Module Guide

When the robot needs to navigate to a specific destination, it communicates with the destination to turn on the IR emitter it has. Navigation towards that destination is then done through aiming the robot towards the IR emitter. The explanation on how this is implemented is given below.

7.2.3.1. Module Interface Specifications

The IR Receivers on the robot are arranged in a horizontal line, aligned in slightly different angles. When the set of IR Receivers detects an IR signal, the robot rotates itself to minimize the distance from the IR Emitter on its middle IR Receiver. The robot is assumed to be correctly oriented towards the plant when the middle IR Receiver has the lowest distance from the IR Emitter, and the ones beside it have the same or slightly larger distance.

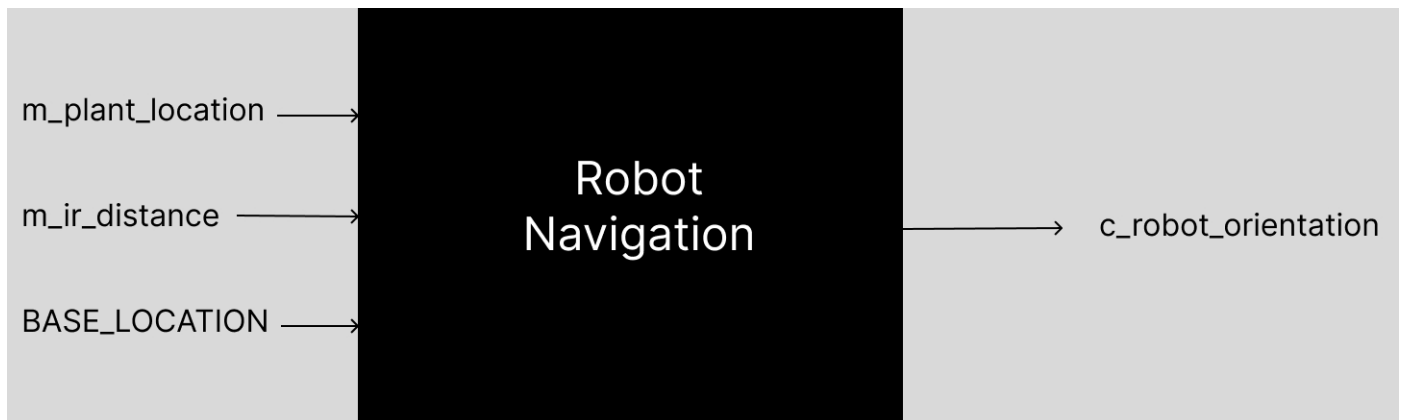


Figure 4: Robot Navigation Black Box

7.2.3.2. Module Internal Design

- Wait until a scheduled time.
- Turn on IR Emitter at the specific destination.
- Rotate robot until the middle most IR receiver has the greatest signal strength, or the same as others.
- Provide the required heading to the [Robot Movement](#) module.
- Turn off IR Emitter once the signal strength is over a certain value, implying the destination is reached.
- Repeat with other destinations.

7.2.4. Scheduling & Timing Constraints

There are no constraints on the timing on the heading to be provided, this should be obtained without any delay

7.3. Obstacle Detection & Avoidance

7.3.1. Purpose

This module helps the robot to detect any obstacles in its path and navigate around it.

7.3.2. Scope

The scope of this module is limited to providing a heading to the [Robot Movement](#) module to navigate around obstacles.

7.3.3. Module Guide

There are 3 scenarios possible when detecting an obstacle:

1. If one specific side (Side1) has a very low distance and the other has a large distance (Side2), the module will instruct the [Robot Movement](#) module to turn the opposite direction and move in the direction of Side2 until Side1 reports a large distance
2. If both sides report a large distance, the robot will choose one side (Side1) and travel in that direction until the other side (Side2) reports a large distance.
3. If both sides report small distance values, the robot will start to reverse. It will keep reversing until either scenarios 1 or 2 are valid, and then follow the valid scenario.

At the end of each scenario, the obstacle will be considered as passed. At this point, the robot will rotate to get back to its initial orientation and track the IR Emitter.

7.3.3.1. Module Interface Specifications

The robot will utilize 3 ultrasonic sensors, 1 at the front, and 2 at the sides. The front sensor will trigger the enabling of this module and the obstacle avoidance algorithms.

7.3.3.2. Module Internal Design

- Wait until the front sensor reports a distance smaller than [MIN_DISTANCE](#).
- Check the left sensor to see if the distance reported is greater than [MIN_DISTANCE](#).
 - If the check is true, rotate the robot until the right sensor reports a distance smaller than [MIN_DISTANCE](#).
 - If the check is false, check the right sensor to see if the distance reported is greater than [MIN_DISTANCE](#).
 - If the check is true, rotate the robot until the left sensor reports a distance smaller than [MIN_DISTANCE](#).
 - If the check is false, reverse the robot until either side reports a distance larger than [MIN_DISTANCE](#), and rotate to that side until the other side reports a distance smaller than [MIN_DISTANCE](#).
- Move forward until the side towards the obstacle reports a distance greater than [MIN_DISTANCE](#).
- Rotate the robot in the opposite direction to which was rotated initially.

7.3.4. Scheduling & Timing Constraints

There are no timing constraints to be considered when detecting an obstacle and instructing the [Robot Movement](#) system to rotate or move.

7.4. Watering Arm

7.4.1. Purpose

Once the robot successfully arrives at a plant, the watering arm attempts to locate soil and avoid foliage. Upon confirmation that soil is in fact detected within a certain distance, it will dispense water onto the soil using a pump.

7.4.2. Scope

The scope of this module is to detect soil within POT HEIGHT RANGE by differentiating between green foliage and brown soil.

7.4.3. Module Guide

After the robot has successfully navigated to the plant, the arm is instructed to attempt to locate soil. At this point, the SEN0101 Color Sensor is powered on and attempts to locate soil while the robot pivots. When it detects the brown color of the soil, the ultrasonic sensor is used to ensure that the soil is within POT HEIGHT RANGE, and if it is the pump is powered on to deliver water from the reservoir.

7.4.3.1. Module Interface Specifications

The ultrasonic and color sensors are attached to the end of the watering arm, just above the output of the nozzle. Both sensors will work together to locate soil color and ensure the correct distance restraints are met. Then the module will power on a pump to dispense water.

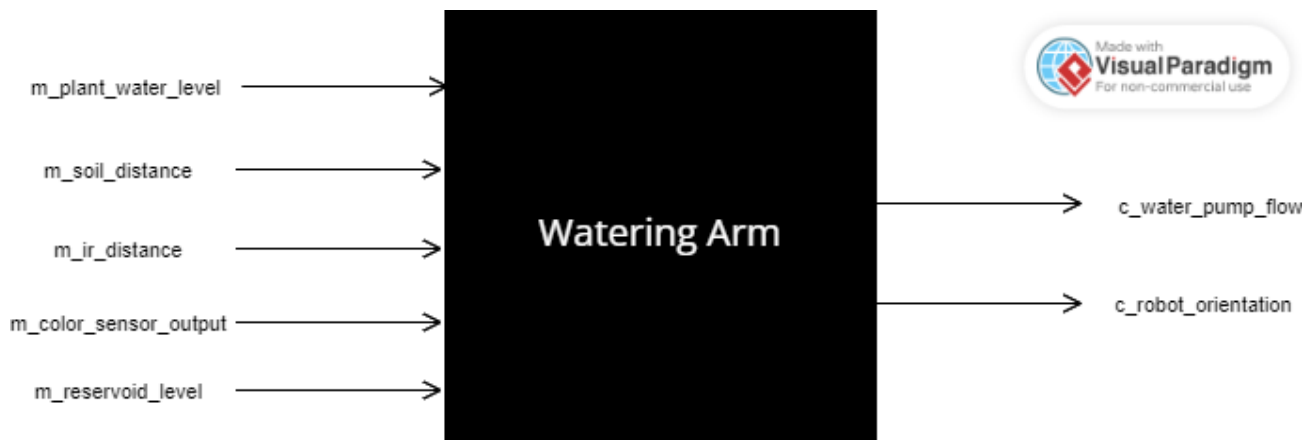


Figure 5: Watering Arm Black Box

7.4.3.2. Module Internal Design

The figure below is a circuit representation of the color sensor and ultrasonic sensor connecting to an Arduino UNO.

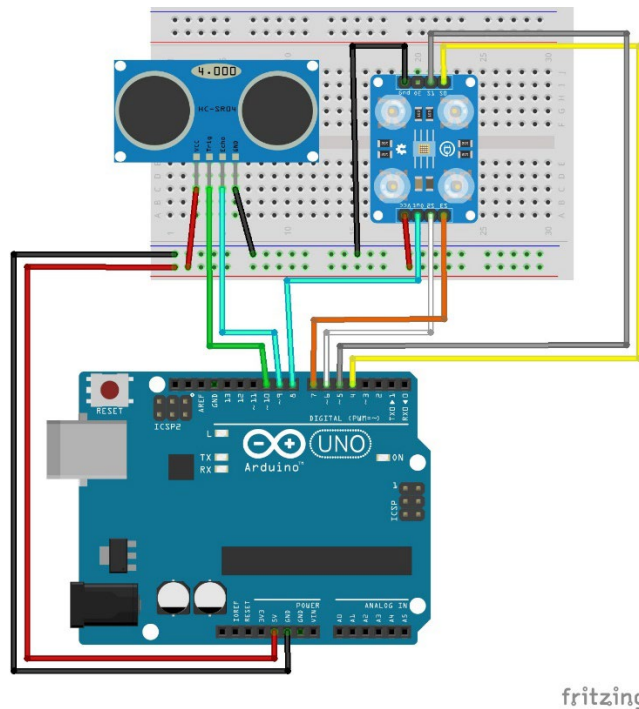


Figure 6: Soil & Distance Detection System

- Wait until the robot arrives at a plant.
- Begin looking for the brown color of soil by pivoting the robot a few degrees.
- Once soil is detected, verify that the distance from the nozzle is an acceptable value.
 - If it is acceptable, power the water pump for a pre-calculated duration corresponding to the plants water level.
 - Otherwise, keep looking for soil.
- Power off pump and sensors.

7.4.4. Scheduling & Timing Constraints

This module has no timing or scheduling requirements.

7.5. Plant Monitoring

7.5.1. Purpose

This module is meant to detect each plants water level and determine when to initiate the watering process, as well as how much water each plant needs.

7.5.2. Scope

The scope of this module includes regular monitoring of the soil moisture level for each plant. Based on this amount, the amount of water required for each plant is determined and when the [PLANT WATER CRITICAL](#) is detected, the watering process is initiated.

7.5.3. Module Guide

This module will make use of soil moisture sensors to get [m plant water level](#) for each plant, and then will use this value to calculate how much water [c plant water needed](#) is required for each plant. Each plant will have an ESP32

microcontroller to transmit the amount of water each plant needs wirelessly to the robot. These values will continuously be compared to [PLANT WATER CRITICAL](#). When the value of any one plant drops below this critical value, the robot begins the watering process.

7.5.3.1. Module Interface Specifications

Each plant will contain its own moisture sensor to obtain water levels and ESP32 microcontroller to transmit information regarding the water levels to the robot. The Arduino UNO will receive this information and output the amount of water required to the watering module.



Figure 7: Plant Monitoring Black Box

7.5.3.2. Module Internal Design

- Wait until [WATER MONITOR PERIOD](#) has elapsed.
- Obtain soil moisture data from the plant.
- Calculate the amount of water (in mL) that would be needed for each plant.
- Transmit this data to the robot.
- If the water level drops below [PLANT WATER CRITICAL](#), initiate the [Robot Navigation](#) module.

7.5.4. Scheduling & Timing Constraints

The Plant Monitoring module should calculate and report [c_plant_water_needed](#) every [WATER MONITOR PERIOD](#).

8. Normal Operation

There are various possibilities to the robot's normal operational scenarios. Here is an ideal operational scenario for when the robot autonomously waters plants around the house, when the user is away.

Pre-Conditions:

- The plants are placed in specific areas and are reachable by the robot.
- A watering schedule has been set by the user.
- The robot has executed the watering operation at least once, successfully.
- The robot is located at a predetermined initial position.
- The robot has enough water to complete its operation.

Scenario:

1. The system reaches a scheduled watering time.
2. The robot starts moving from its initial position, [BASE LOCATION](#).
3. The robot navigates to the first plant to be watered.
4. The robot avoids obstacles along its path when moving.
5. The robot reaches the plant.
6. The robot aligns the watering mechanism with the plant's soil.
7. The system pumps the required amount of water.
8. The system repeats the navigation and watering process with other plants if present.
9. The robot returns to its final position, [BASE LOCATION](#).

Success Post-Conditions:

- Plants that require water according to the schedule have been watered.
- The robot causes none to minimal environmental damage.
- The robot reaches its predetermined final position.

9. Undesired Event Handling

There are various events that could occur, which should not have. A few of these events are given below, along with the desired behaviour, their consequences and how they will be handled.

9.1. No path is available

Under desired conditions, the robot has executed a watering operation at least one successfully to ensure all plants are reachable. However, it is possible that the user could skip a test run or that from the time of this test run to the time of use, new obstacles arise preventing the robot from reaching the plant. If the robot is unable to reach a plant after various rerouting attempts and deems it not possible, a notification will be sent to the user expressing that the device cannot reach its plant.

9.2. Insufficient water available

A condition may arise where the robot has an insufficient amount of water for its scheduled operation. The system should be filled each time the user goes away, however if they are gone long enough it's possible that after the system will have used all the water. If the robot is scheduled to begin a run but does not have the necessary water, it

will complete what it can with the water remaining. When no water remains, it will inform the user that it is unable to complete the task.

9.3. Foliage in the way

When watering the soil, the plant's leaves may obstruct the flow of water from all directions, and there may be a very specific direction from which there is an opening. For this scenario, the robot will try to use a little bit of force to attempt in pushing away the foliage, in attempt to reach the soil. Failing that, the robot will have to skip the plant if it cannot find an angle to the soil from any direction and inform the user.

9.4. System tips over

There could exist a scenario in which the robot tips over. Some reasons for this include a failed collision detection or human/pet interference. If the system is no longer upright, it must inform the user that it fell over and is unable to complete any tasks. The system will also shut down to conserve battery.

9.5. Loss of communication

There is a possibility of the robot failing to communicate with any external components or actors, such as the user. While the user may not be notified of any issues, the system must keep a log of all events that has occurred and inform the user about these events once a connection has been re-established.

10. Changes

10.1 Anticipated Changes

- AC1. Additional modules to allow the user to control the robot may be added in later.
- AC2. The plant monitoring module's scope may be expanded to incorporate assisting in the real time amount of water being added to the plant.
- AC3. The navigation module may be expanded to incorporate other navigational means, such as utilizing ultrasonic sensors for Simultaneous Localization And Mapping (SLAM) techniques.
- AC4. The scope of the movement module might be expanded to include different methods of movement in different environments.
- AC5. A system and module may be created to replace the [BASE LOCATION](#). This may involve IR Emitters to allow the robot to locate itself, more powerful navigational algorithms, and data processing.
- AC6. A module may be created to implement the usage of an IMU and stop all operations if the robot's normal operation is interrupted.

10.2 Unlikely Changes

- The movement module will not be responsible for figuring out where to head, only use the instructions it is provided to proceed at a specific speed.

11. References

No references were used for this document.