41068 Robotics Studio 1

Spot Search and Rescue

Sprint Number: 2

Version Number: 1

Date: 11-09-2024

Team K-10

|  |  |  |
| --- | --- | --- |
| Student Number | Full Name | Email |
| 24443691 | James Silsby | james.silsby@student.uts.edu.au |
| 14163885 | Joseph Tarbath | joseph.a.tarbath@hotmail.com |
| 24459594 | Connor Williams | connor.a.williams@student.uts.edu.au |
| 24502825 | Jet Webb | jet.t.webb@student.uts.edu.au |
| 24439287 | Tamsyn Crangle | tamsyn.m.crangle@student.uts.edu.au |

Contents

[Project Description 4](#_Toc176913575)

[Sprint 1 5](#_Toc176913576)

[SLO 1.1 Communicating with the stakeholder 5](#_Toc176913577)

[SLO 1.2 Identifying and agreeing on the priorities, goals, and system requirements 5](#_Toc176913578)

[SLO 2.1 Problem Statement 6](#_Toc176913579)

[SLO 2.2 Functional requirements and design parameters of project 6](#_Toc176913580)

[SLO 2.3 Technical Statement 6](#_Toc176913581)

[SLO 2.4 Design Objectives 7](#_Toc176913582)

[SLO 2.5 Evaluation Criteria 8](#_Toc176913583)

[SLO 2.6 Timeline for the Sprint 1 8](#_Toc176913584)

[SLO 3.1 Configuring the system 8](#_Toc176913585)

[SLO 3.2 Setting up a simple indoor office environment in Gazebo 8](#_Toc176913586)

[SLO 3.3 Design and Develop a Subscriber/Publisher 9](#_Toc176913587)

[SLO 4.1 Time Management 10](#_Toc176913588)

[SLO 4.2 Communication Skills 10](#_Toc176913589)

[Sprint 2 11](#_Toc176913590)

[SLO 1.1 11](#_Toc176913591)

[SLO 1.2 15](#_Toc176913592)

[SLO 2.1 16](#_Toc176913593)

[Top Level: 16](#_Toc176913594)

[Localisation: 17](#_Toc176913595)

[Mapping: 18](#_Toc176913596)

[Audio Processing 19](#_Toc176913597)

[SLO 2.2 22](#_Toc176913598)

[SLO 3.1 22](#_Toc176913599)

[SLO 3.2 23](#_Toc176913600)

[SLO 3.3 23](#_Toc176913601)

[SLO 3.4 23](#_Toc176913602)

[SLO 3.5 24](#_Toc176913603)

[SLO 3.6 (IN LAB) 25](#_Toc176913604)

[SLO 3.7 25](#_Toc176913605)

[SLO 4.1 (IN LAB) 25](#_Toc176913606)

[SLO 4.2 25](#_Toc176913607)

[Appendix 26](#_Toc176913608)

[Client Meeting 1 Minutes 26](#_Toc176913609)

# Project Description

The aim of this project is to develop an autonomous search and rescue mapping system, using Spot, the robot dog, to assist first responders in hazardous urban earthquake zones by identifying gas hazards as well as key areas of interest for the responders. The primary goal is to map a safe and efficient route to be taken by responders that leads to areas of interest where potential casualties may be. We plan to identify hazardous gasses commonly found in disaster zones (CO, CO2 and CH4), using a gas sensor and a combination of infrared, sound and RGB-D sensors to identify potential people trapped through usage of thermal, visual and audio data. Spot will autonomously plan and execute two stage room sweeps to enhance the accuracy of the identification systems.

The project will utilise Gazebo simulation software and a TurtleBot 3 to simulate the actions of Spot. We will use a simulated urban earthquake disaster zone with areas of each gas type and two hidden people to test our systems capability. Through this testing, we aim to test the robot on a TurtleBot 3 as the end goal for this project.

# Sprint 1

## SLO 1.1 Communicating with the stakeholder

Meeting date: 16/08/2024

Stakeholder name: Sangmin Song

Discussed the project situation, types of sensors that will be used, and how we are going to simulate the code.

Also discussed goals for the project which are listed below

1. Gas leak situation

2. Sound Sensing

3. Finding a hidden person

4. Report back a safe path

More details can be found in Appendix A – Client Meeting Minutes

## SLO 1.2 Identifying and agreeing on the priorities, goals, and system requirements

Our team wanted to design a robot that could detect and save people from a variety of natural disasters. We wanted this to be versatile so that we could help with floods, fires, earthquakes, and others. After discussions with our stakeholder, we decided to prioritise the most feasible of these situations, being post-earthquake support. We also determined that spot is incapable of digging out trapped people, hence we settled on the detection of points of interest. With these things in mind, our primary goal is to have our search and rescue robot, spot, explore earthquake affected areas and determine locations it believes to have trapped people.

Our priorities to achieve this goal are to incorporate navigation and mapping using SLAM, implement sensors including audio, hazardous gas, thermal, and an integrated system to combine, process and determine areas that are most likely to contain trapped people. These areas will then be transmitted to the master control panel for further investigation by rescue teams. These have been determined to be vital functions to provide a successful search and rescue robot, with other functionalities and features being wanted but not essential at this stage. As the project continues our current priority and focus will shift, however, these points will remain the most important.

## SLO 2.1 Problem Statement

In the aftermath of an earthquake in an urban area, first responders are exposed to a highly dangerous environment including hazards that cannot be seen with the naked eye such as natural gas, this slows down the response time reach to casualties, whilst they make the area safer for the responders. As the casualties may be critically injured, this delay can be the difference between life and death. This opens a design opportunity to develop a system using the Spot robot to enter these hazardous areas and create a map for the first responders, highlight key areas such as pockets of natural gas, and potential casualties through the use of visuals, infrared, and sound, using this data to create a map for the first responders to locate and rescue the casualties.

## SLO 2.2 Functional requirements and design parameters of project

|  |  |
| --- | --- |
| **Functional Requirements** | **Design Parameters** |
| The simulation must have gas leak detection | The system must be able to detect when, CO, CO2, CH4, reach a level that is unsafe for humans. It also must detect the location of this leak within 5 minutes. |
| The simulation must cover four separate rooms | The system must be able to navigate four separate rooms with a success rate of 90% |
| The simulation must investigate blind spots | The system must investigate over 90% of blind spots. |
| The simulation must differentiate between a person and the environment | The system must be able to differentiate between a person and the environment with 85% certainty. |
| The simulation must identify sound from a person | The system must be able to distinguish between sound from a person and the environment with 80% accuracy. |
| The simulation must identify safe route to casualty | The system must be able to create a path to a casualty that is possible to be taken by the system. It must not be a path that significantly damages the system. |

## SLO 2.3 Technical Statement

The team has decided to simulate an urban earthquake environment in Gazebo because of the flexibility in creating 3D environments. Custom models can be created to simulate a variety of areas, obstacles, and challenges. Moreover, the team is familiar with using the software gazebo, having used it in previous subjects. This knowledge will help the team to make an intricate and well-crafted simulation. Furthermore, UTS only has access to one spot robot across all engineering classes which makes accessing the robot difficult. Therefore, a simulation using a turtle bot will be a faster and more efficient way to test and run code.

The choice of sensors was first discussed within the team and then discussed with the client. The main sensors being used are an RGB-D camera, microphone, gas sensor and thermal camera. The RGB-D camera was chosen because it is part of the spot robot. The microphone, gas sensor and thermal sensors have been chosen because they can be added to the gazebo simulation. Furthermore, they will be required to complete the tasks set by the client.

## SLO 2.4 Design Objectives

|  |  |
| --- | --- |
| Gas leak detection | The sensor will be able to detect specific gases CO, CO2 and CH4 (natural gas). The concentration of these gases will be reported back to the user and location marked. The goal of gas detection will be able to successfully locate areas of hazards gas. |
| Simulation area of 4 rooms | The area will consist of 4 different rooms with different sizes and obstacles. In one of the rooms there will be a hidden person which the turtle bot will identify. |
| Investigate blind spots | The turtle bot can navigate the terrain and map its location as it travels into the disaster area. |
| Differentiate between a person and an object | The turtle bot will use a range of sensors including an RGB-D camera, thermal camera, and microphone. Using a combination of those sensors the turtle bot will be able to identify a hidden person. It will also be able to differentiate between a person and the surrounding area. Once a person has been found the location will be flagged and sent back to the user. |
| Identify sound from a person | The turtle bot will utilise a microphone to detect and identify human-generated sounds, such as speech, footsteps, or other vocalisations. This capability will enable the system to distinguish these sounds from ambient noise, ensuring accurate identification of a person's presence through auditory cues. |
| Identify safe route to casualty | During the autonomous navigation of the turtle bot a safe route will be calculated and sent back to a user. The route will be a series of waypoints. Using the RGB-D camera, pictures will be sent back to the user to visualise the waypoints. The goal of the design is to be able to share the safest route to a user. |
| Demonstrate hardware | The information received from the sensor in the system should send the gas data, sound data, identification data and waypoints back to a terminal. The data should be displayed in an easy-to-read way. |

## SLO 2.5 Evaluation Criteria

|  |  |
| --- | --- |
| Criteria | Description |
| Gas Leak Detection | The TurtleBot should be able to detect the following gasses within an error margin of ±10%:   * CO * CO2 * Natural Gas |
| Simulation Area | The TurtleBot should be able to investigate at least 4 different rooms in their entirety.   * Blind spots should be identified and investigated without additional user input. |
| Human Detection | The TurtleBot should be able to positively identify a human with an accuracy of 75%. |
| Sound Detection | The TurtleBot should use sound to identify potential hazards and humans. |
| Safe Routing | The TurtleBot should identify safe routes for casualty extraction, minimising risk to self and humans. |

## SLO 2.6 Timeline for the Sprint 1

A screenshot of a computer

Description automatically generated

## SLO 3.1 Configuring the system

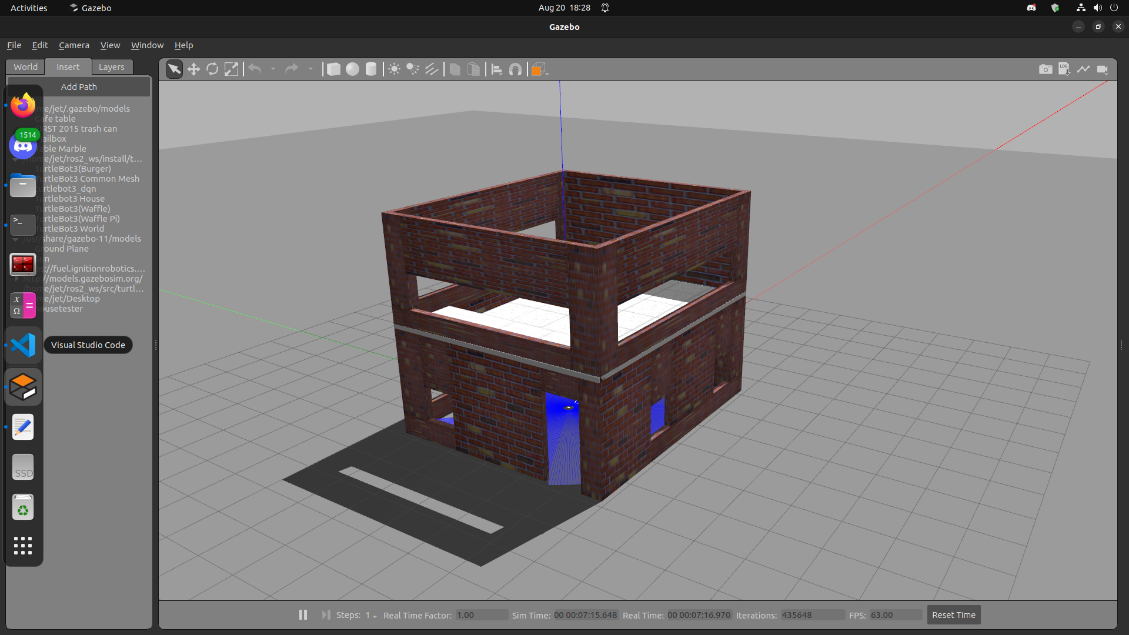
I decided to set up my Linux environment as a bootable external SSD as I had a small SSD that was not in use, and it will allow me to more easily complete work and troubleshoot problems as I am familiar with Ubuntu’s graphical user interface. I was considering using a virtual machine, however, as my laptop is relatively old, running Linux through a virtual machine on windows causes significant stuttering and crashes. To configure my system I followed the guide on canvas and made minor quality of life adjustments to my Linux install such as disabling mouse acceleration, installing terminator, and a variety of other convenient programs and settings.

## SLO 3.2 Setting up a simple indoor office environment in Gazebo

Initially I was having trouble running the house world on Gazebo. I attempted rebuilding the ROS workspace using colcon build, however, it did not fix it. After some research I found that the issue was that it simply needed a long time to load everything in on the first run. To resolve this issue, I left Gazebo loading on my desktop instead of my laptop for the first run after removing the build folder and doing a full rebuild.

I also initially had trouble creating a package as I was getting many errors while trying to run my code that didn’t seem relevant to the code I had written. After some research online I found out that the issue was ROS has other configurable files that need to be manually configured after creating the package such as package.xml and setup.py. After I made the changes suggested online to these files such as adding the name of my code file, the ros2 run command begun to work as expected.

Below is a simple indoor office environment setup I created within Gazebo that includes the TurtleBot.



## SLO 3.3 Design and Develop a Subscriber/Publisher

I used NumPy’s array slicing through *scan\_msg.ranges[::self.n]* in order to only keep every nth value of the array and remove the rest, I placed these values within a new LaserScan message. The code then multiplies the angle between each scan by n so that the scans are placed in the correct location, this needs to be done as through removing a group of points, the angle between the remaining points will increase. In addition, time\_increment is multiplied by the same amount so that the time between each scan is also corrected.

In order to change the value of n without having to restart the simulation and manually modify the code, I created a subscriber that listen for an Int8 and changes n to whatever value is read. To utilise this feature, an integer can simply be published to the topic set\_n using ros2 topic pub through the terminal in order to change the sequence of scans.

A diagram of a software company

Description automatically generated with medium confidence

## SLO 4.1 Time Management

The below image shows my groups Trello board displaying how we have organised and assigned each SLO to a member of the group who will take lead on that specific SLO. All tasks within sprint 1 have been transferred to the Done section.

A screenshot of a computer

Description automatically generated

## A screenshot of a phone Description automatically generatedSLO 4.2 Communication Skills

In order to effectively communicate we have been using a group teams chat to stay up to date with the other members of the team. We have especially been using the polling function within teams in order to decide on directions we want to head as a team, for example, the poll below shows how we decided on which project to follow. We also had a group discussion both before and after this poll to ensure everyone was happy with the outcome.

# Sprint 2

## SLO 1.1

**List of Constraints**

**Time Constraints**:

* The project must be completed within the 12-week course deadline. This is a constraint because the client is only working with for the duration of the course. Therefore, we must complete the task before the end of the 12-week course. To combat this constraint project deadlines and project management techniques are used to make sure the team complete the work on time.
* Limited time available for development, trouble shooting and code testing due to conflicting schedules. This is a constraint because team members rely on other work being completed to continue their task. To combat this Trello boards and team check ins are used to keep everyone on track.

**Budget Constraints**:

* There are no budget constraints for this project as the project is all simulated.

**Resource Constraints**:

* Availability of skilled personnel or team members. Each team member has different level of skill. Therefore, we are constrained by each other's technical abilities. Each team member needs to complete their own research of each part of the code.

**Scope Constraints**:

* The project must stay within the predefined objectives, goals, and deliverables which were discussed with the client. For example, the code we produce does not have to tell the difference between sounds, rather it must mark locations of sound.
* Another example would be gas detection does not need to identify all gas types but a select few.
* Lastly, we must complete all the goals and deliverables defined at the start of the project.

**Technology Constraints**:

* Dependence on certain software like azure being available.
* Compatibility issues between different systems or languages for example half the team is working in python while the others are working in C++. Therefore, if the code is not compatible together then we have a software constraint.
* The team is not allowed to use the spot robot to test the code so we are constrained by the simulation and the testing we can do in gazebo.

**Quality Constraints**:

* Specific quality standards or performance benchmarks the project must meet.
* Customer or stakeholder expectations for the final deliverable must be within the defined quality.

**Dependency Constraints**:

* The project relies on the completion of other tasks, projects, or external entities before proceeding.
* Delays in other projects or processes may impact timelines and deliverables.

**List of Risks and Mitigation Strategies**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activity** | **Associated Hazards** | **Inherent Risk** | **Existing Control Measures** | **Proposed Control Measures** | **Residual Risk Level** |
| Traversing areas filled with gas | * Explosion/ fire * Oxygen deprivation (human operators) | * Burns (human operators) * Damage to sensors/ electronics/ hardware (robot) |  | * Use gas detection equipment onboard to mark areas of high gas concentration * Ensure autonomous operation of robot * Insulate robotic components to prevent spark ignition | Low |
| Traversing unstable buildings/ infrastructure | * Impacts from debris * Fine dust | * Impediment of motor/ sensor/ joint function * Impediment to breathing/ sight * Crushed by debris |  | * Ensure autonomous operation of robot * Maintain smooth and consistent motion * Keep robot lightweight and mobile | Low |
| Electrical Hazards | * Electrocution | * Electrocution * Sensor interference | * Cut power to disaster zones | * Ensure autonomous operation of robot * Avoid areas filled with water | Low |
| Communication Interruptions | * Exposing human operators to hazards | * Robot becomes trapped in disaster area and requires humans to enter the area to rescue |  | * Ensure autonomous operation of robot * Create a safe path that leads to the robot’s current position | Low |
| Human interactions with the robot | * Hesitancy about autonomous systems | * People may be reluctant to be rescued by an autonomous robot and this may induce panic in casualties |  | * Clearly indicate on robot that it is a search and rescue robot * Provide training to operators on robot interactions | Low |
| Data Security | * Sensitive/ personal data exposure | * Unsecure data poses risk of releasing data about sensitive sites (e.g. military bases) and casualties |  | * Implement encryption and secure data transmission protocols to protect sensitive information gathered by the robot | Low |

## SLO 1.2

**Ethical:**

Many ethical concerns need to be considered when designing and deploying our search and rescue robot. The most prominent being if Spot fails to detect a person in danger, which could lead to harm, or death of the person involved that may have been avoidable if the algorithm was better. If this occurs, are we the designers and manufacturers of the product to be held responsible for the robot's indirect harm to a person.

In addition, the algorithm will be required to make decisions without human input, such as deciding the priority and order of people to be saved. This order could be sorted in many ways, such as from severity of injuries, safest to reach and extract for the first responders, or through the time required to retrieve each person. We must decide on one of these sorting systems in order to have an operational robot, however, any decision will result in some people in danger being disadvantaged and directly reduce their chances of survival.

Finally, we must ensure that protections are in place that enforce ethical use of our recorded sensor data, especially relating to the camera feed and microphone. Spot will be collecting possibly sensitive information that could lead to a breach of individuals privacy if misused. However, this must be done while ensuring efficiency and safety is not affected by the added layers of security.

**Legislative:**

No specific legislative restrictions are in place for robot products, but general health and safety practices, and ‘common sense’ should be considered during project design. Automated, moving components are a hazard and algorithms should fail safe prevent harm to persons or property. Recommendations from government organisations, such as the NSW Centre for Work Health and Safety, for collaborative robots should be followed to minimise these risks. Beyond harm caused by the robot and its operations, the K-10 project will need to be aware of safety limits for gases, heat and other environmental parameters it will come across in its operation so that it can mark areas as being safe or not. Being safety oriented, most, if not all, of these will be publicly accessible from government resources.

Outside of operating in a safe manor, robotics can come into legislative challenges related to the sensor packages used. The K-10 project does not contain any restricted sensors in its current form so will not currently need to be concerned with this, but any future sensor upgrades will need to be aware.

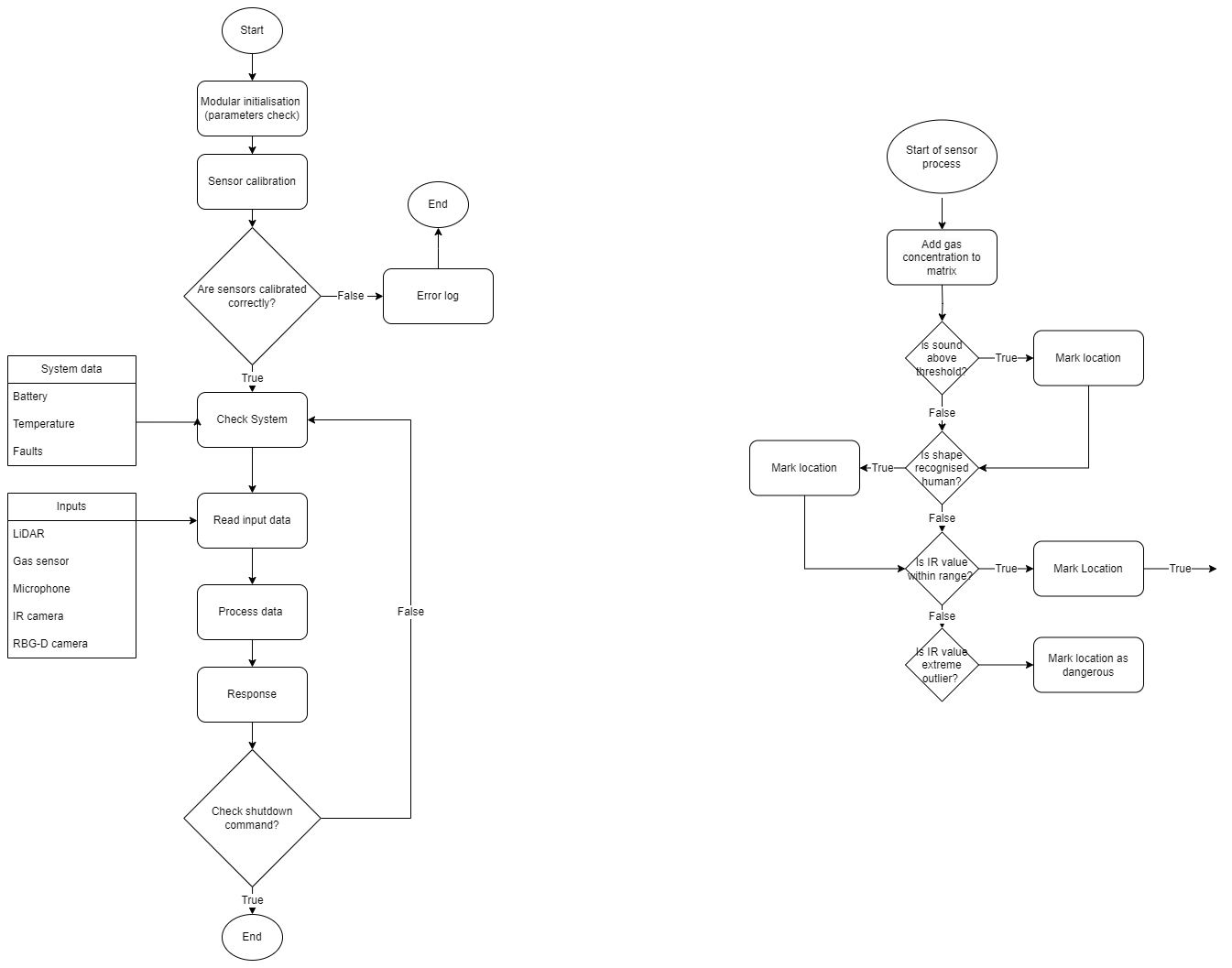
**Economic Perspective:**

K-10s application as a search and rescue robot provide significant economic benefits, both to the safety of first responders, and through its cost-effective design. K-10 is designed to alleviate the need for first responders to put themselves in life threatening situations, which often leads to injuries or preventable deaths. These occurrences place a significant economic burden on rescue operations, with hazard pay and expensive medical operations taking place. By significantly reducing the risk to these humans’ lives, K-10 is a much more cost-effective solution.

Additionally, K-10 is designed with danger to itself in mind, sporting a robust design and commercially available components. These design choices keep the overall cost of K-10 low, and maintenance is also easy to perform. By acknowledging that K-10 will experience damage, wear, and potential destruction, a lower cost of replacement and maintenance helps ensure that K-10 is as economic as possible, and that the downtime between missions can be minimised.

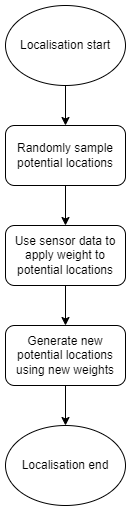
## SLO 2.1

### Top Level:



The system operates within a primary function loop, which takes in inputs, parses them to respective subsystems and functions to be processed and update variables, then refreshing its information to determine navigation and detections. This main operations loop is initialised after variable setup and will continuously run and provide feedback to the central control unit. The central control unit can be used to provide goals or areas to search, otherwise the spot will autonomously begin its operations based upon sensor data.

### Localisation:



Pseudocode:

START

GET sensor data

while (random\_sample\_list < desired\_samples) {

sample = Generate random number from 0-potential\_list.size();

random\_sample\_list.push\_back(sample);

}

Generate weights from sensor data

Remove low weight samples

while (potential\_list.size() < desired\_locations) {

randomly select sample;

new\_location = valid location within distance of sample;

Potential\_list.push\_back(new\_location);

}

END

Mapping:



Mapping Pseudocode:

START

GET sensor\_data

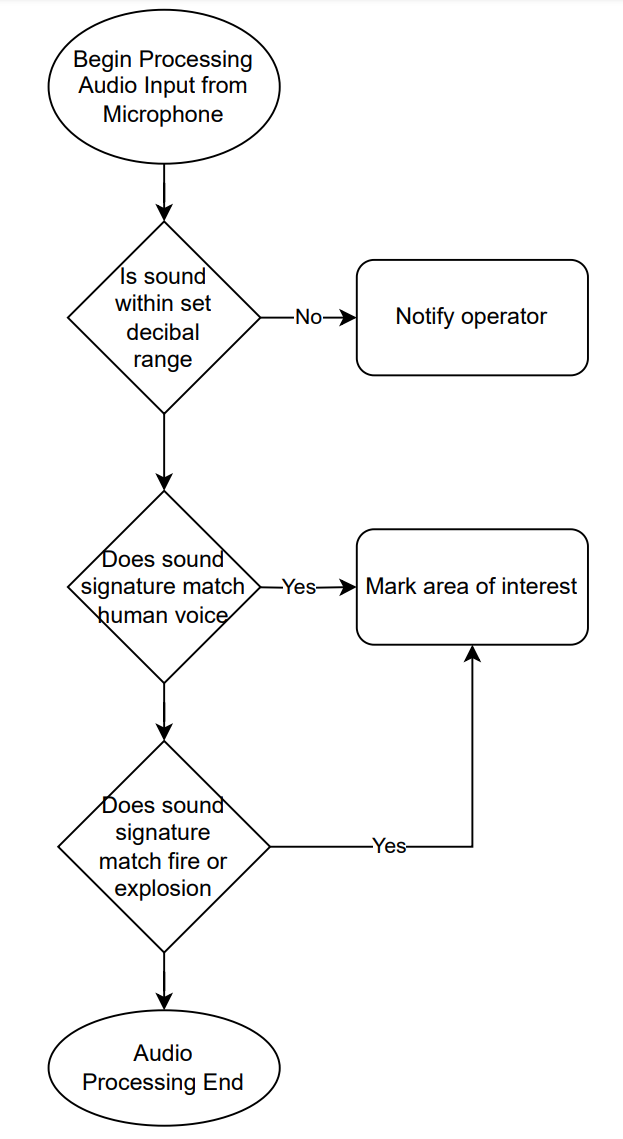
GET camera\_data

new\_map = MAP\_PROCESSING(sensor\_data, camera\_data)

UPDATE\_MAP(new\_map)

END

### Audio Processing



**Pseudocode:**

START

RECORD audio\_data FOR 5s

IF MAX\_VOLUME(audio\_data) > max\_vol\_threshold

NOTIFY\_OPERATOR()

IF MATCHES\_SIGNATURE(audio\_data, human\_voice)

MARK\_AREA\_INTEREST(“HumanVoice”)

IF MATCHES\_SIGNATURE(audio\_data, fire\_audio) OR MATCHES\_SIGNATURE(audio\_data, explosion\_audio)

MARK\_AREA\_INTEREST(“Danger”)

END

A diagram of a flowchart

Description automatically generatedCasualty Detection

**Pseudocode**:

START

OBSERVE DATA FROM IR, RGB-D AND AUDIO

IF TEMP > (temp min) && TEMP < (temp max):

IF SHAPE MATCHES PERSON:

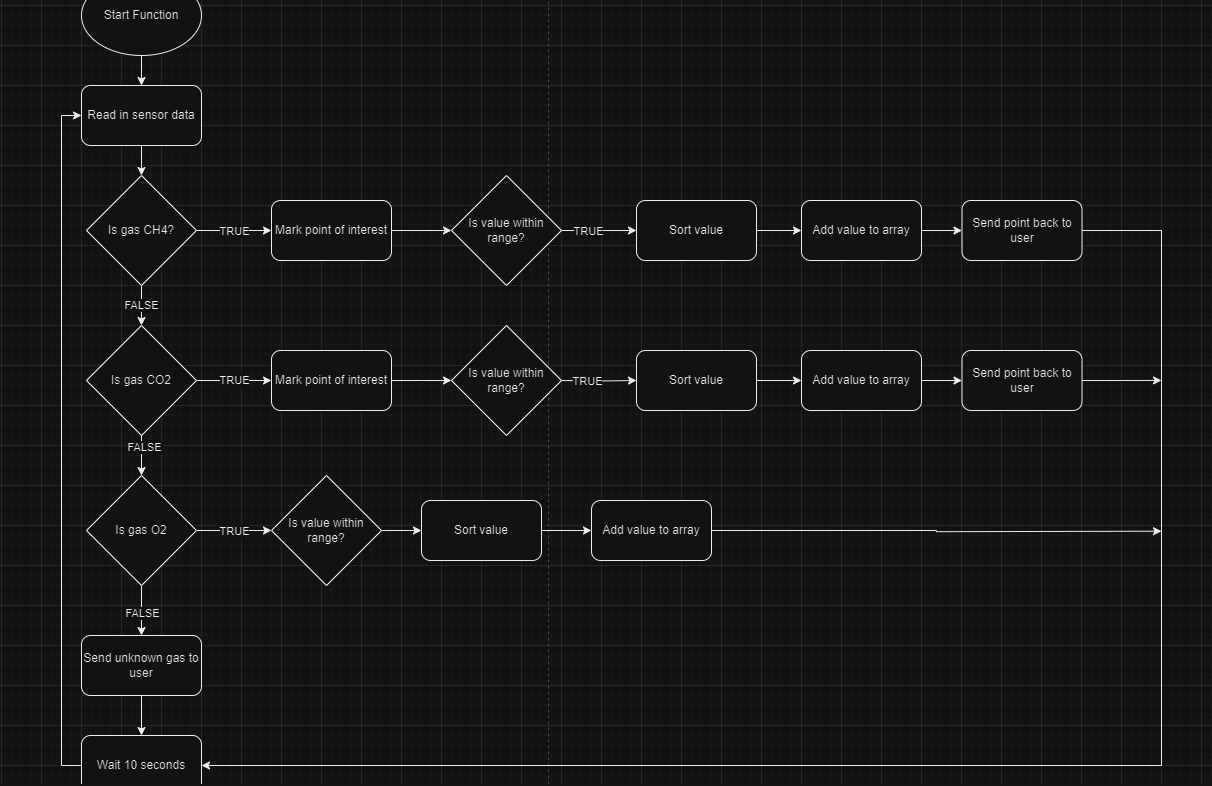
SET FLAG -> “POI”

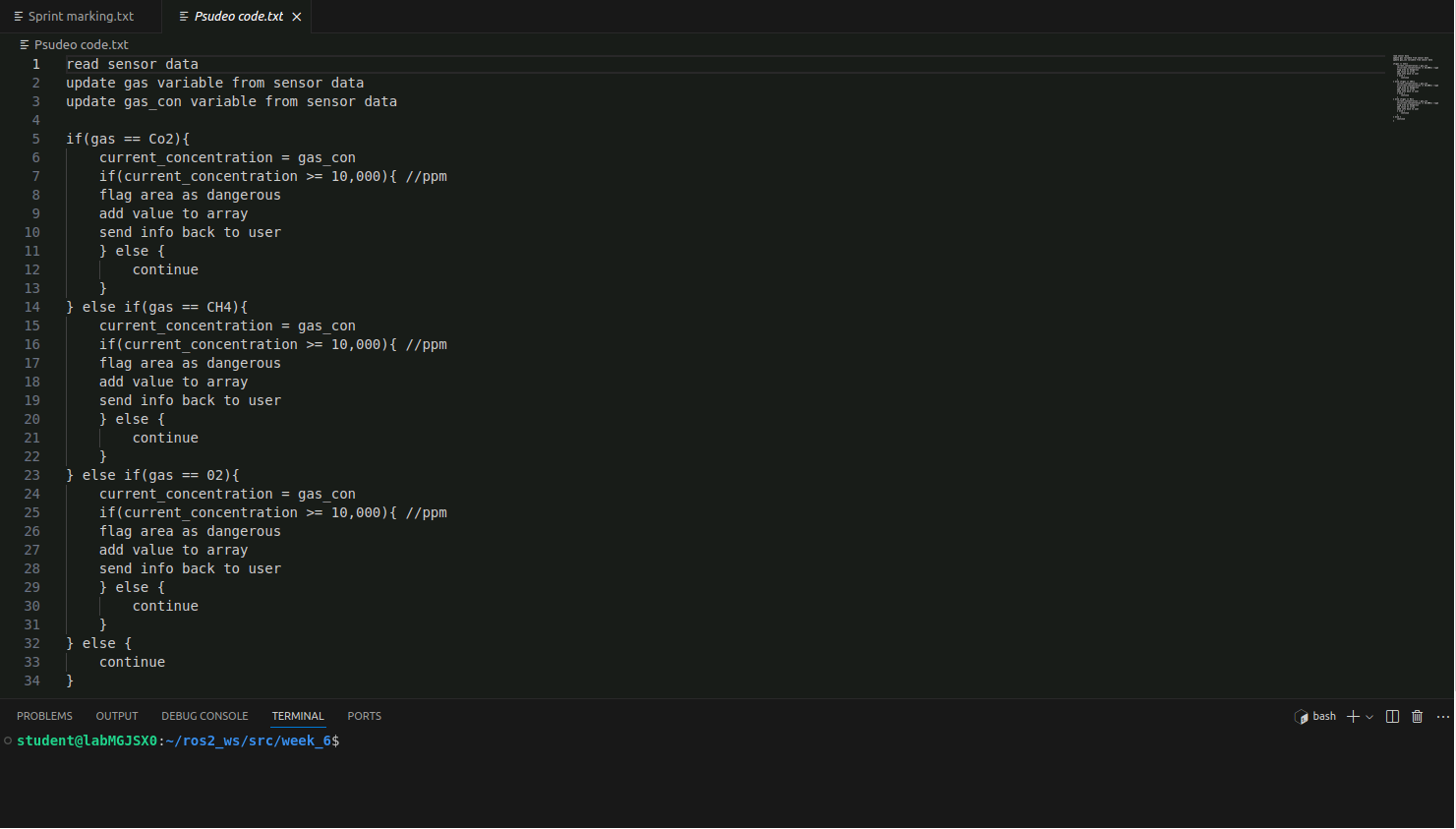
IF AUDIO MATCHES PERSON:

SET FLAG -> “CASUALTY”

ELSE

ELSE

ELSE



## SLO 2.2

Challenges:

Due to my slow laptop and the complex geometry of our world, it took a long time during the first time loading it up. Any launches after this were significantly faster. As this was my second time creating a world environment within Gazebo it was significantly easier as I already knew exactly what needed to be done and I could just move the newly created files into the relevant ros workspace folders and colcon build, with it working the first time I ran it. Below shows an image of my groups project world taken from within Gazebo.

A screenshot of a computer game

Description automatically generated

## SLO 3.1

Mapping is the process of creating a representation of an environment (a map) from collected data of the surroundings.

Many sensors can be used for mapping including:

A LiDAR, which is a sensor that uses pulsed laser light in order to measure ranges. This data can be used to create a matrix of points to estimate the surrounding environment.

An RGB-D sensor, which can gather red, green, blue, and depth information from the environment it is looking at. This can be used to create a textured 3D model of the environment.

A SONAR sensor, which emits high frequency sound waves and measures the time delay between sending them and receiving the sound reflections from objects in the environment. The distance can be calculated by the time delay between sending and receiving and is often used for underwater mapping.

All sensors that can be used in mapping have their advantages and disadvantages, so the best sensor is different for each application. While LiDAR is extremely accurate and works well in low light scenario’s, it doesn’t provide as much fidelity or colour as an RGB-D sensor would. However, RGB-D sensors are less accurate and require a certain amount of light. While accurate, SONAR requires sound to be used which can harm or irritate nearby animals.

Any mapping task will encounter some level of noise which must be filtered out in order to have useful data. In addition, mapping methods where the robot must move, such as LiDAR require algorithms to move the robot efficiently so it can scan other areas without wasting time.

## SLO 3.2

When setting up my mapping solution I was unable to see any topics within rqt\_plot, even after entering the topics into the graph nothing appeared. I am not sure what the cause for this issue was, however, restarting my computer fixed the problem and the next time I started everything up, all topics were displaying correctly in rqt\_plot. In addition, I had some errors when initially trying to save my maps saying it had failed to spin map subscription. I solved this error through ensuring I was running the command within a suitable location as it did not work in the HOME directory or my desktop, I created a folder called Maps and navigated to that folder so I don’t clutter my ros2 workspace.

## SLO 3.3

A screenshot of a computer

Description automatically generated

## SLO 3.4

Localisation is the process of determining a robot’s position and orientation with respect to its environment through using a previously generated map. This can be done through matching similar features based on sensor readings with features from the existing map.

An IMU can be used in combination with other sensors to determine the robot’s orientation.

A camera can be used to match features that can be seen with an existing image or map to determine location.

A LiDAR can be used to find similar patterns in the points that are read.

Localisation has many challenges including fusion of sensor data, taking into account and correction of noise within the sensor data, and processing power requirements to do the localisation in real time.

These challenges can be reduced through the use of a variety of techniques such as Kalman Filtering or particle filtering.

## SLO 3.5

When I was trying to set up localisation I had issues running my world file, no matter what world I ran, it always opened the same one that I had open previously and the TurtleBot would not reset its position. I figured out I had an instance of Gazebo stuck open in the background through checking ros2 topic list and seeing that there were many topics available. After this, I ran htop and saw gzserver from ros2\_ws was running and taking a significant portion of my CPU. To resolve this issue, I SIGKILL’ed the task to fully close it and when I reran my command to open my gazebo world it worked correctly, with the correct world opening, and the TurtleBot’s position being reset.

A colorful image of a hexagon with many circles

Description automatically generated

Robot Localisation in Simulation Environment

## SLO 3.6 (IN LAB)

## SLO 3.7

A screenshot of a computer

Description automatically generated

## SLO 4.1 (IN LAB)

## SLO 4.2

A graph with green and orange lines

Description automatically generated

Above is a visual comparison of the localisation systems implemented by three of our group members. While mine and Connor’s constantly change, Tamsyn’s stays more consistent, however, at a larger inaccuracy. All the data used in the graph above was collected while the TurtleBot was moving around within our project’s simulation environment.

# Appendix

### Client Meeting 1 Minutes

|  |  |
| --- | --- |
| **Client Name** | Sangmim Song |
| **Team Name** | Team K - 10 |
| **Meeting Date/Time** | 16/08/2024, 10am |
| **Attendees** | James Silsby, Tamsyn Crangle |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Discussion Items** | **Agenda** | **Notes** | | |
|  | Team Introduction | Only two team members were present  Others were working (Jet, Joseph, and Connor) | | |
|  | Project Overview | Urban earthquake environment using spot robot. Simulated in gazebo with a turtle bot. | | |
|  | Discussed Sensors | RGBD camera  Sound sensor (microphone)  IR camera  Gas sensor | | |
|  | Discussed Deliverables | * 1 area of gas leak * 1 area with a hidden person (behind a wall, etc) * Differentiate between a person and an object * 1 scenario with sound from a person (any sound) * Send a safe route back to the users * Put into real hardware | | |
|  | Clarified with Client | Clarified the deliverables and meeting / discussion options with the client. 2 more meetings this semester (week 6 and week 11). Best form of contact emails or teams. | | |
|  | Discussed sprint | What is going to be in the sprint  Clarified face to face demonstration of pubs and subs | | |
| **Action Items** | **Action** | | **Owner(s)** | **Due by** |
|  | Share Problem statement | | James | 16/08/2024 |
|  | Prepare for the Sprint | | Everyone | 21/08/2024 |
|  | Relay information to the rest of the team | | Tamsyn | 16/08/2024 |