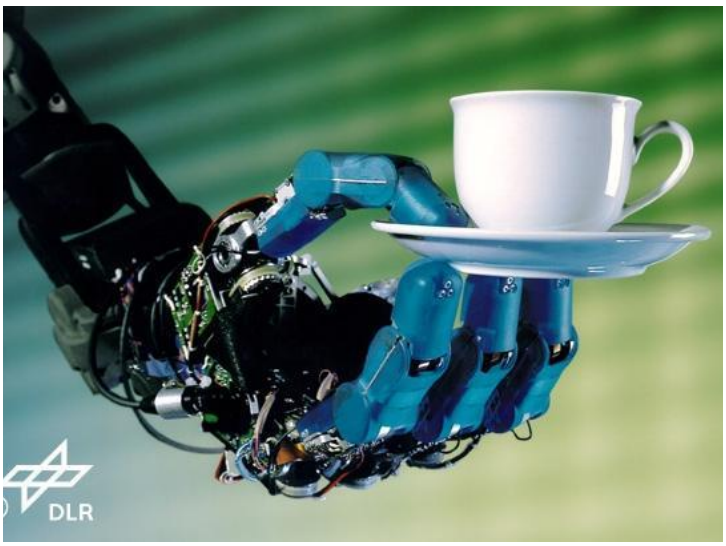
TECHIN 517 Project Documentation
Hydration Robot



[Robot holding teacup]

Authors: James Muir, Sulaiman A. Al-Akbari, Annan Li

Created for: TECHNIN 517 Project: Hydration Robot Team

Hydration Robot

a robot that provides beverages to nursing home residents

1 Introduction

1.1 Purpose

What is the purpose of your document?

This document is a detailed description of our TECHIN 517 Project: Hydration Robot. It summarizes our design process and testing results.

What is the purpose of your project?

The purpose of our project is to design a robot to help keep elderly people in a nursing home properly hydrated. Our project uses a Fetch Mobile Manipulator robot to serve drinks to the residents, freeing up caregiver time for other tasks. Our design goals include safe operation, useful functionality, and otherwise minimal interference with daily life for both residents and caregivers.

1.2 Scope

The scope of this document is to provide [what?]

The document includes seven sections: Introduction, Project Description / Problem Context, Analysis and Conceptual Design, System Requirements and Evaluation Criteria, Evaluation Results, Gap Analysis and Conclusion, and the Appendix.

The Introduction provides a brief overview of our project. The Project Description / Problem Context section goes into more depth about the problem our project is addressing. The Analysis and Conceptual Design section shows our design process and our project design. The System Requirements and Evaluation Criteria section lists the project requirements and the methods we will use to evaluate if our project meets those requirements. The Evaluation Results section includes the summarized results of our project evaluations. The Gap Analysis and Conclusion section contains a discussion of where our project could be improved and an overall reflection on the project as a whole. The Appendix includes more detailed evaluation results.

The scope of this project is [what?]

The goal of the project is to develop a robot to help keep nursing home residents hydrated. To facilitate achieving this goal, we were given access to a Fetch Mobile Manipulator robot and a workspace in GIX, as well as a simulated version of the robot and workspace. Our project requirements mandate that we include manipulation (using the robot arm) and navigation (driving the robot base) tasks. The work for this project consists of designing the behavior of the robot, implementing that behavior in code, testing our code on the robot in simulation and in real life, and documenting our work.

The intent is to [what is the intent?]

The goal of the class (TECHIN 517) is to teach us how to design and program robots. The goal of this project is to provide us with an opportunity to apply what we have learned in class (the labs) to a real-world application and to run our code on a real, physical robot.

A second goal of the project is to give us experience in designing robots that meet human needs. This project involves the creation of a robot that fulfills a real-world use case and interacts with humans in their living spaces. Our design process involves the consideration of human needs, including safety and hydration, and desires, including efficiency and unobtrusiveness.

What will you illustrate with the project and what demonstration will be included in this document?

The final demonstration of our project will include a live demonstration of the robot performing at least one of the set of tasks we set out for it, a video of the robot completing all of the tasks either in real life or in simulation, the source code for our project, and a description of our design in this document.

What specifications are you adhering to?
We will be using ROS as the basis for our code, which will be written in Python. The robot will adhere to all of the safety recommendations laid out by Fetch Robotics.
What kind of system are you documenting?
We are documenting the design of our robot software, the actions the robot will take, and how the robot will interact with its environment.
What type of analysis and what design considerations will you be addressing?
We will be addressing safety, human interaction, utility, and robustness considerations.
What is outside the scope of your document?
The selection of the problem space is outside the scope of this project, as it was assigned to us. Hardware design is also outside the scope of this project, as we were provided with a stock Fetch Mobile Manipulator robot and a preselected environment at GIX.

1.3 Problem Summary

This document applies to the development of [what?]
This document applies to the development of a hydration robot, the function of which is to serve elderly people drinks in a nursing home.
What is the problem you are trying to solve?
Elderly people in nursing homes often get dehydrated. They can have trouble or forget to get themselves a glass of water. Caregivers are often very busy and don't have enough time to make sure that everybody has enough water.
Where does this problem arise?
Mainly in nursing homes, but potentially in other places where elderly or disabled people live.
What are some of the conditions that create or contribute to this problem?
Old or disabled people who are not able to/forget to get themselves water, busy caregivers who forget or do not have enough time to make sure residents are hydrated.
What are some priorities for the solution?
Our solution will prioritize safety and functionality.
What are some subproblems that will require design decisions on the part of your team?
The main function of the robot includes two tasks: 1) autonomous navigation and 2) manipulation ability. For the navigation ability, the robot should conduct the following tasks:

1) Navigate to the kitchen to pick up a can
2) Navigate to a table to give the can to a resident
3) Navigate to back to the kitchen with the empty can
4) Navigate back to the robot's starting/home position

For the manipulation ability, the robot should conduct the following tasks:

1) Pick and tuck: pick up a can and tuck it so the robot can safely navigate
2) Untuck and place: place a can on a tabletop

1.4 Bill of Materials

What are all the parts you need?
Record any shortages and overages.
Amend this Bill of Materials each time you turn in your project documentation, as needed, as your BOM changes.

- Fetch Mobile Manipulator: a differential-drive robot with an attached arm and a gripper end effector, lidar, and a combined color and depth camera.
- Space at GIX: a space at GIX for the robot to operate in, outfitted with furniture and possibly human occupants

- A red can of Coca-Cola: our robot is designed to pick up and move red (we chose red for its visual distinction) coke cans (we chose coke cans because they are red, are a beverage, are readily available, and don't spill easily)

1.5 Glossary

Define any terminology that your documentation readers will require as they read through your project documentation.

- ROS: The Robot Operating System (ROS) is a set of software libraries and tools that help you build robot applications.
- Depth Camera: A camera that captures how far away each pixel is from the camera as well as its color.
- Octomap: a 3D occupancy grid mapping library that enables the robot arm to avoid colliding with perceived objects

2 Project Description / Problem Context

[This section will establish system context and system boundaries.]

2.1 Problem Background and Motivation

[Describe some background information on the project and explain why it is useful.]

Nursing home residents often face dehydration. Reasons include forgetting to get water or the lack of ability to get up, get a glass, fill it with water, and drink it.
Currently, nursing home caregivers are in charge of ensuring that residents are hydrated. However, they are very busy and often overworked, leading to hydration issues.
Building a hydration robot would ensure that residents are always hydrated and would free up time for caregivers to do other tasks.

2.2 Goals and Objectives

What specifically are you trying to achieve?

1. Ensure that nursing home residents are hydrated
2. Free up time for caregivers to other tasks than serving water/beverages

2.3 Stakeholders and Entity Analysis

2.3.1 Stakeholder Definition

Stakeholders:

- Direct:
  - Nursing home residents
  - Nursing home caregivers
- Indirect:
  - Nursing home owners/operators
  - Nursing home residents families

2.3.2 Stakeholder Requirements

- Nursing home residents:
  - minimum: medically adequate hydration, easy to use
  - preferred: beverages delivered whenever they are desired, otherwise non-intrusive robot
- Nursing home caregivers:
  - minimum: medically adequate hydration of residents, reduced amount of time spent hydrating residents
  - preferred: nonintrusive system, easy to use

2.3.3 Entities and Entity Requirements

What other entities play a role in your problem definition?

- Coke Can: We chose a coke can because it is a beverage, it is red, it is easy and cheap to obtain, and it is similar in size and shape to a glass of water, so our algorithms will hopefully generalize to

a glass of water as well
  - One thing to note is that a closed can cannot spill, while a full glass of water can. We do not plan to make the robot work with a full glass of water in this project, because we do not want to accidentally spill water on the robot.
- Furniture: our robot will be designed to pick and place coke cans on tables, which will be located in the GIX environment
  - We are assuming that we will know about the size and layout of the furniture in advance.
  - In the future, perhaps we could add the ability to automatically recognize furniture to the robot.
- Doors: we are assuming that all of the doorways are open for this project.
  - In the future, perhaps we could add the ability to open doors to the robot, but that is out of scope for this project.

2.4 Context (Boundaries and Use Cases)

2.4.1 Context Diagram

Set all the Stakeholders and entities you defined above within the context of your system

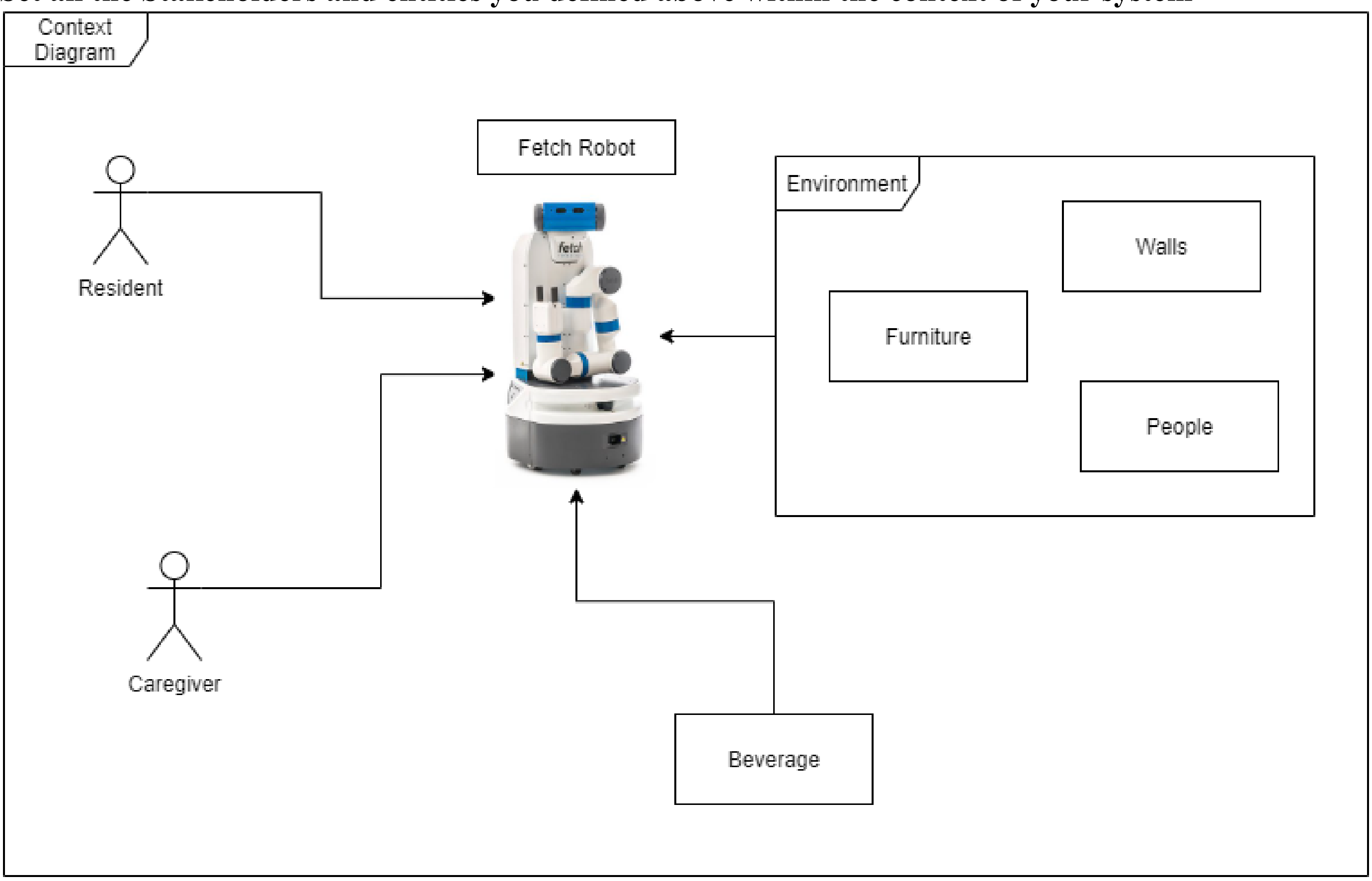


Figure 1: Context diagram

2.4.2 Use Cases

John, a 80-year-old man, is moving into a nursing home. John sometimes forgets things, so his kids are wondering how the staff will take care of him. For example, they want to know who will be responsible for delivering drinks to John everyday to keep him hydrated. One of the staff in the caring house has introduced a serving robot to John's kids. The robot will fetch and deliver drinks to John as scheduled when he is in the dining room. The whole process of the robot is totally autonomous and the robot will keep a certain distance from the elders to avoid collision, so they will be safe with a new partner. In doing so, the schedule of serving drinks will not be wrong as a robot staff takes care of the elders.

2.4.3 Top Level Use Case

The top level use case of our robot is to automatically deliver beverages to nursing home residents.

1. Locate the cup by camera and pick it up
2. Navigate to the area where residents are and avoid collision
3. Place the cup on the table and getting back for waiting

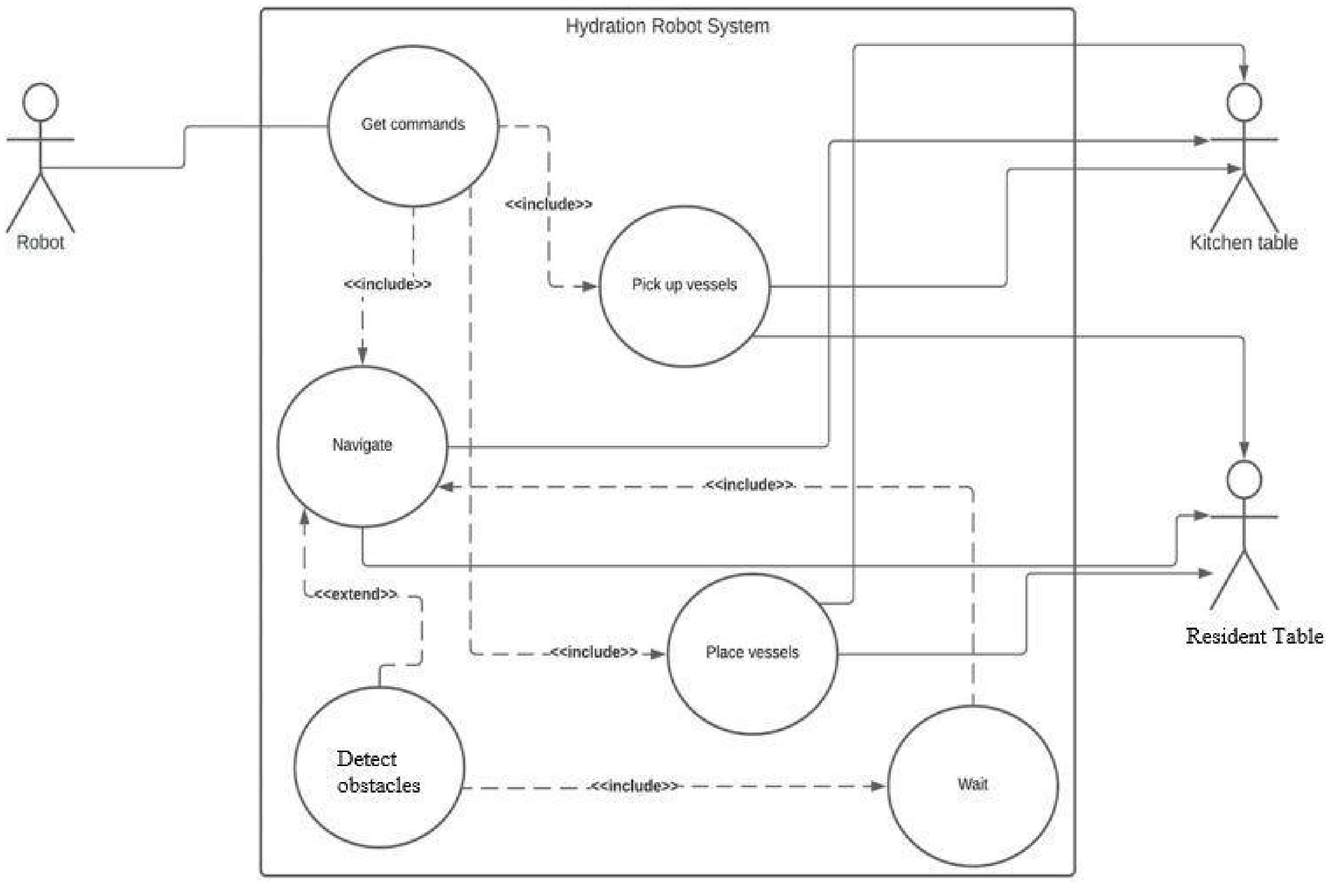


Figure 2: UML Robot, Kitchen & Residents

2.4.4 Lower level Use Cases

[Please describe one additional use case for your robot project, for example, an instance in which one of the residents interferes with the robot. Remember our discussion of runtime flexibility. You have a choice of using an operational or a goal-driven use case. For each additional use case you describe, please describe the use case shortly in words and then provide an appropriate use case diagram, choose the type of diagram from the choices below:]

Diagram - Operational Use Cases

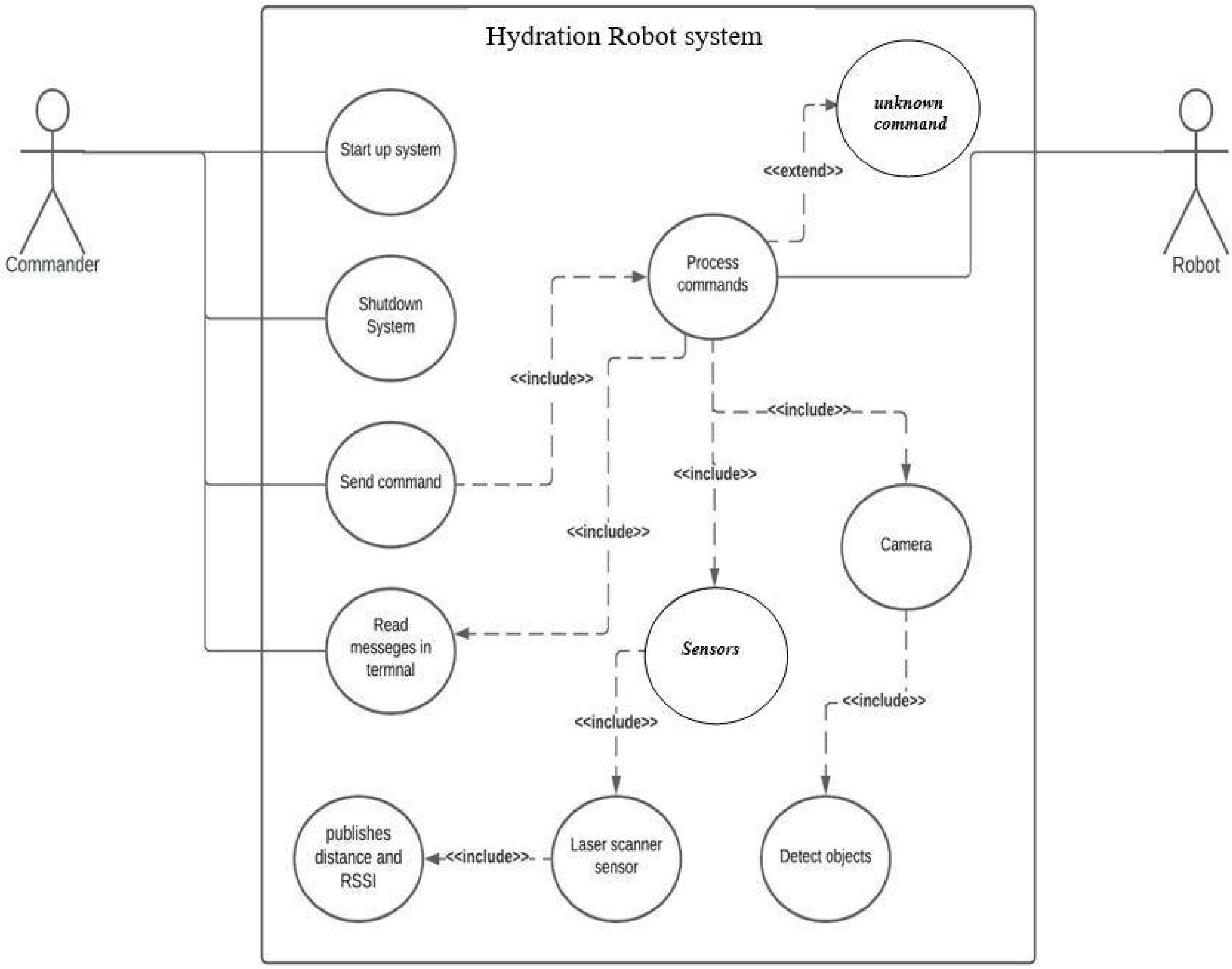


Figure 3: UML Commander & Robot

Diagram – Goal-driven Use Cases

Lower level use-cases include: avoiding interference with other humans in the scene, navigating cluttered environments, navigating through doorways, avoiding obstacles with the arm.

1. avoiding interference with other humans in the scene
   John is waiting for his drinks while the robot is moving to him. Once the robot detects other people in front of it, it will slow down and stop to avoid interference with other people.
2. navigating cluttered environments

The robot is navigating to John, but it finds out the environment is pretty crowded. It will avoid collisions with other obstacles when making plans for moving.

3. navigating through doorway
   The robot is coming through a doorway which is a little narrow for it, so it autonomously adjusts the direction and status to make it through the doorway.
4. avoiding obstacles with the arm
   The robot stops in front of John's table and tries to place the cup on it. But the camera detects there is something on the table, so the robot tries to avoid collisions with the obstacles by planning the arm carefully and moving slowly.

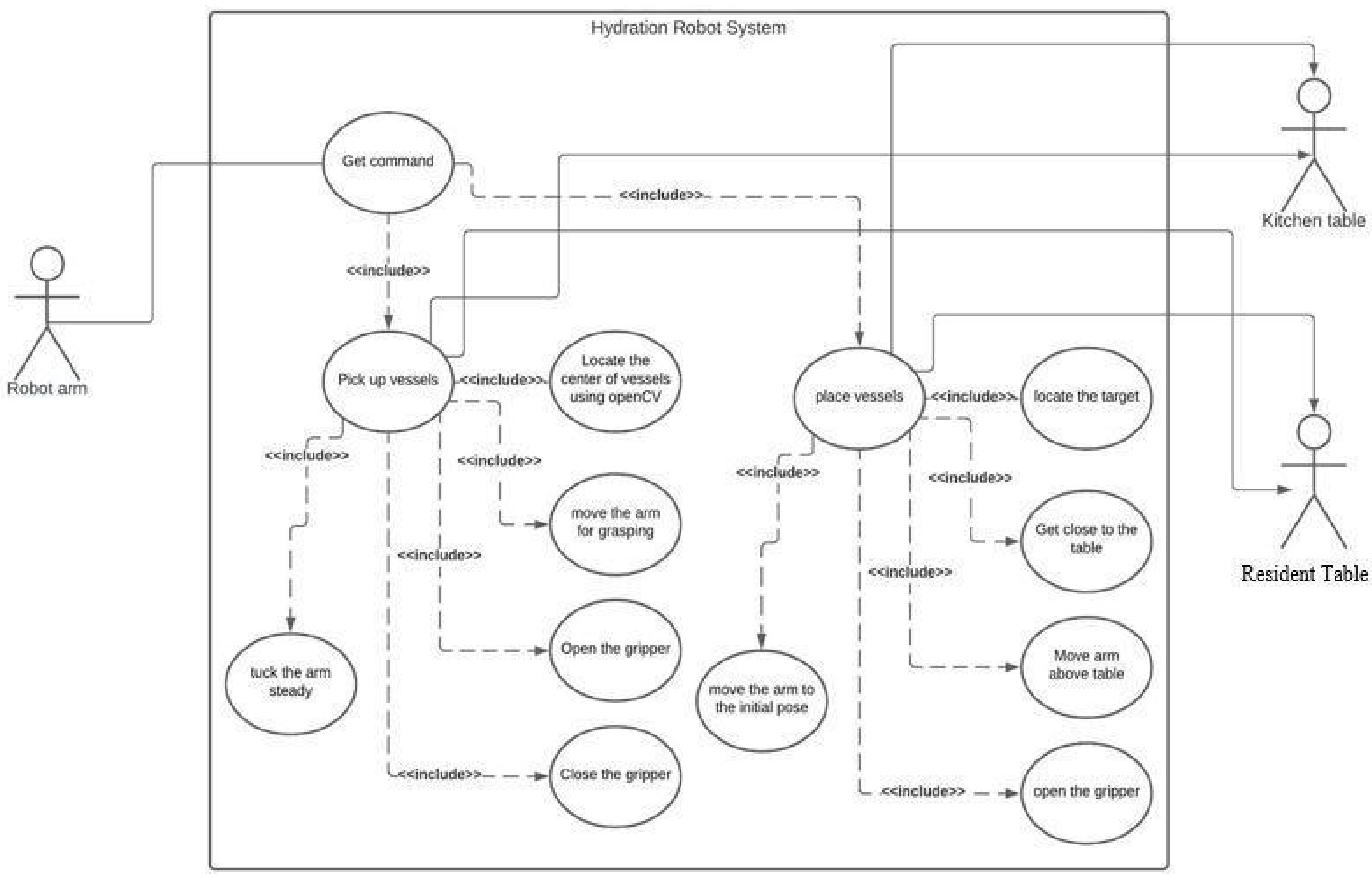


Figure 4: UML Robot's arm, kitchen & residents

3 Analysis and Conceptual Design

[Here you get to elaborate on the functions and activities of your system, performance analyses, structure, and behavior.]

3.1 Elaborating Behavior (Sequence Diagrams and State Diagrams)

The following sections present various ways of describing the robot's intended behavior with diagrams.

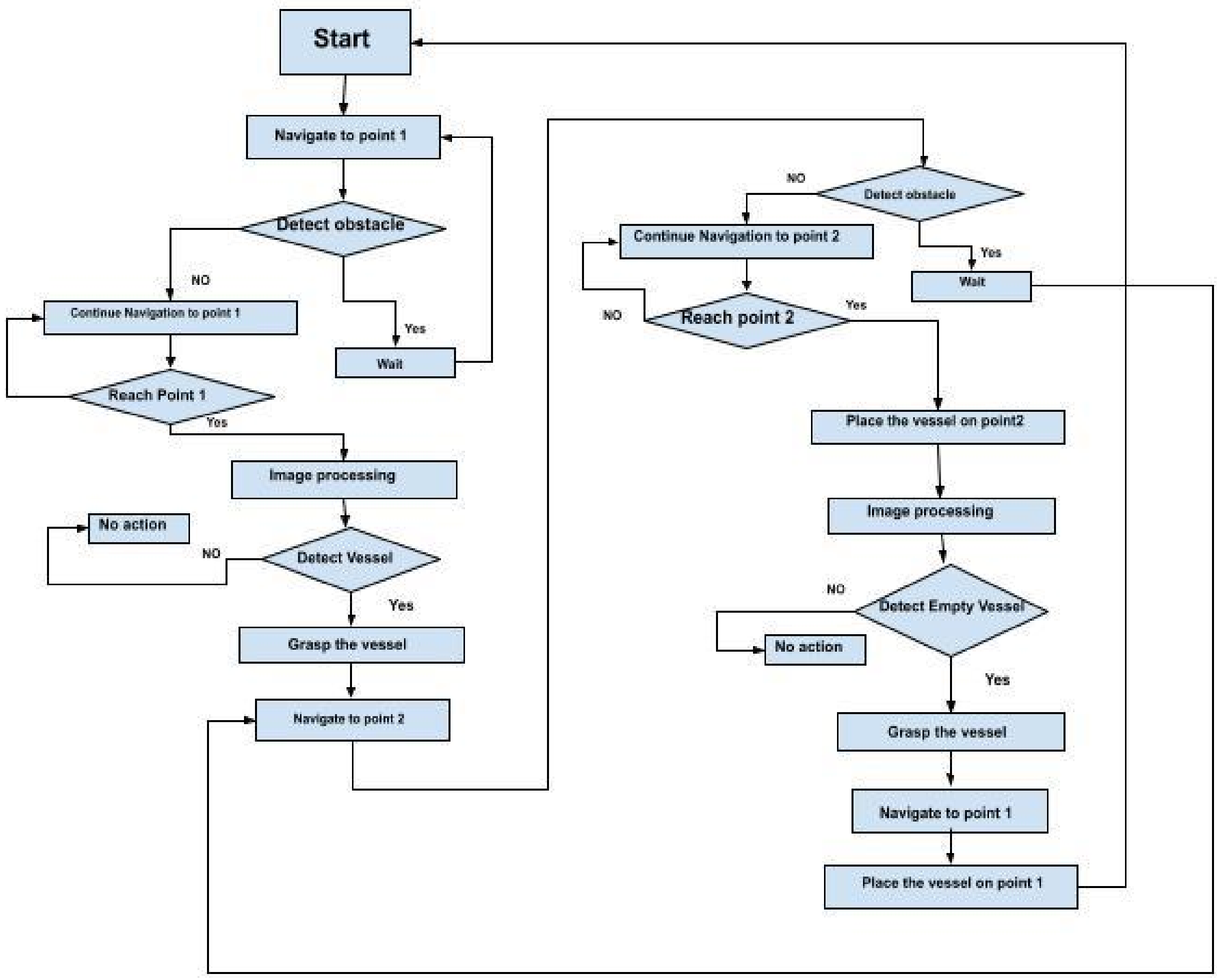
3.2 Sequence Diagram

Our example will provide a sequence diagram for a Drive Black Box

In your example, you will provide a sequence diagram for a "Remove Trash/Laundry" or "Hydrate Residents" black box.

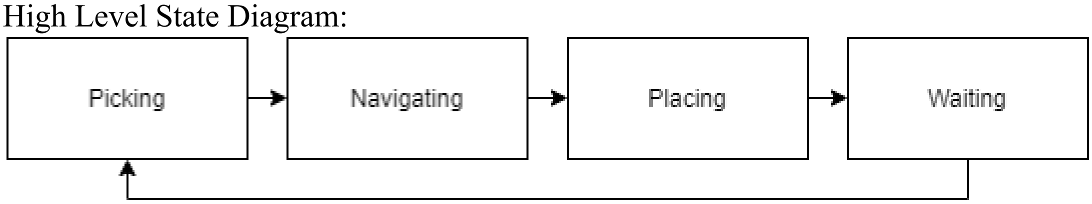
Sequence Diagram:

Flow Chart:

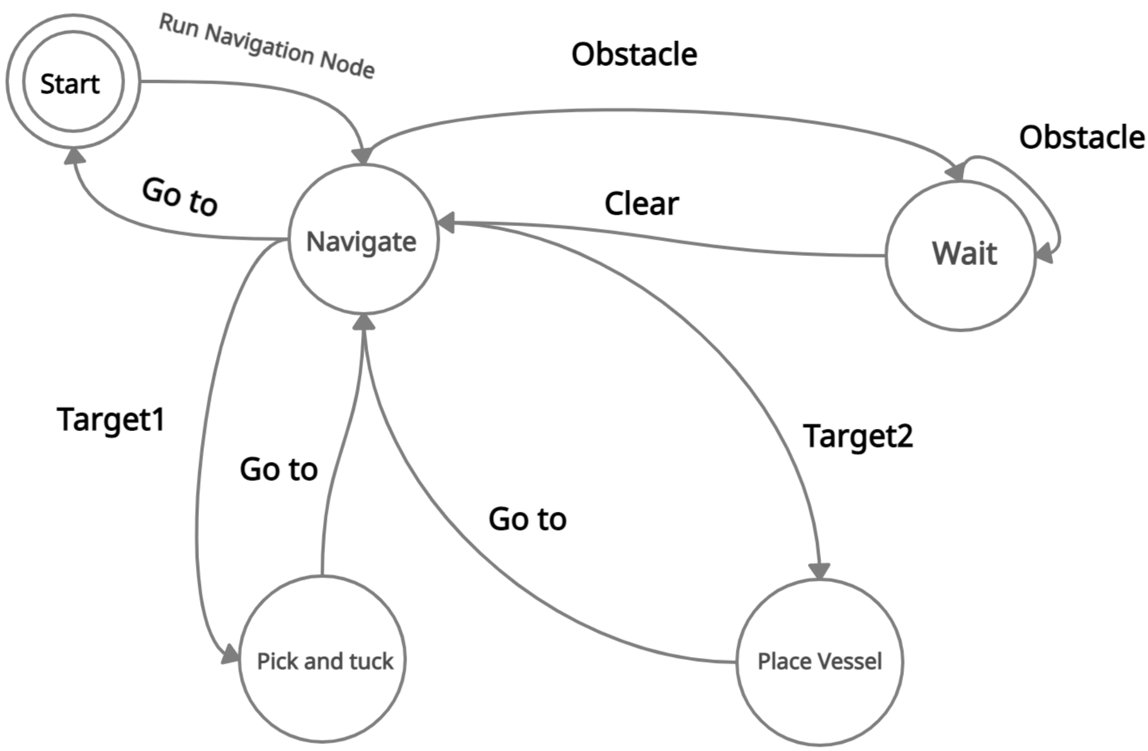


3.3 State Diagram

High Level State Diagram:



The robot will go through this loop twice. The first time it picks up a cup in the kitchen and brings it to the resident, the second time it picks up the empty cup from the resident and returns it to the kitchen.
More Detailed Finite State Machine Diagram:



Please clearly explain why you chose this system architecture?
This high level system architecture represents the four main tasks that the robot will have to engage in, and the flow between them required by our use case.
This system architecture represents the behavior flow of the robot, which is also a mock of normal behavior flow from a human being in real life. If a server wants to give drinks to people, he should first pick it up in the kitchen, then navigate to the area where the guests are, placing the drinks on the table next and waiting for other jobs as a last step. So with this architecture applied to our robot, the serving process won't be strange to other people.

4 System Requirements and Evaluation Criteria

In order to fulfil its design requirements, the robot must perform certain functions. This section describes those functions and the criteria we will use to evaluate whether the robot successfully performs those functions. The main tasks the robot must perform are autonomous navigation (with obstacle avoidance) and picking and placing objects (using perception).

High-Level Requirements:

- Safety
  - Avoid running into obstacles (including humans) with the base
  - Avoid hitting obstacles (including humans) with the arm
- Functionality
  - Keep residents hydrated
    - Autonomously bring beverages to residents
      - Autonomous navigation

- Autonomous picking and placing
  - Perception of drinking vessel

4.1 Derived Requirements

ID	Details
1. Safety	The robot must operate safely around humans.
1.1. Safe Navigation	The robot must not run into obstacles (including humans) when navigating.
1.2. Safe Grasping	The robot must not hit obstacles (including humans) with its arm.
2. Functionality	The robot must keep nursing home residents hydrated.
2.1. Navigation	The robot must autonomously navigate the nursing home without running into any obstacles.
2.1.1. Navigation Obstacle Perception	The robot must be able to detect obstacles with its sensors and avoid running into them.
2.1.2. Localization	The robot must be able to determine its position using sensors.
2.1.3. Mapping	The robot must be able to create a map of its environment to use for navigation.
2.1.4. Navigation to Waypoints	The robot must be able to pathfind to specified waypoints, then navigate to those points.
2.2. Manipulation	The robot must be able to pick up and place the drinking vessel without hitting any obstacles.
2.2.1. Grasping	The robot must be able to pick up the drinking vessel.
2.2.1.1. Drinking Vessel Detection	The robot must be able to identify the location of the drinking vessel on the tabletop using its sensors.
2.2.2. Placing	The robot must be able to place the vessel on a table.
2.2.3. Manipulation Obstacle Perception	The robot must be able to identify obstacles with its sensors and avoid hitting them with the arm.

4.1.1 Human Interaction Requirements

Due to time constraints related to the length of the class and safety constraints related to the COVID-19 pandemic, the scope of this project did not include testing the robot system's effectiveness with real nursing home residents. However, we wanted to practice our human-centered design skills, so we came up with these human robot interaction related requirements.

ID	Details
3. Stakeholder Interaction	The robot must fulfill the needs of the stakeholders.
3.1. Hydrating Residents	The robot must increase the hydration level of the nursing home residents
3.2. Saving Caregiver Time	The robot must reduce the amount of time nursing home staff must spend on hydrating the residents.
3.3. Saving Operator Money	The robot must reduce the overall cost of running the nursing home.

4.2 Establishing Evaluations

This section lays out the criteria we will use to evaluate whether the robot system meets the project requirements.

4.2.1 Safety

Safety	The robot must operate safely around humans.	**Evaluation Procedure:**
Safe Navigation	The robot must not run into obstacles (including humans) when navigating.	1. Send the robot a navigation goal 2. Step in front of the robot 3. Record if the robot runs into you or stops 4. Repeat 5 times 5. Score: # of hits / # of runs
Safe Grasping	The robot must not hit obstacles (including humans) with its arm.	We did not have time to add this functionality to the robot, so we will not test it.

4.2.2 Autonomous Navigation

Navigation	The robot must autonomously navigate the nursing home without running into any obstacles.	**Evaluation Procedure:**
Navigation Obstacle	The robot must be able to detect	See Safe Navigation from the

Perception	obstacles with its sensors and avoid running into them.	Safety section
Localization	The robot must be able to determine its position using sensors.	1. start the navigation stack 2. give the robot a pose estimate 3. teleoperate the robot to a new location 4. measure the distance from the robot to the walls in real life 5. measure the distance of the robot to the walls using rviz 6. repeat 5 times 7. accuracy score: % error of measurements averaged over all attempts
Mapping	The robot must be able to create a map of its environment to use for navigation.	1. run the map building scripts 2. measure distances on the map using RViz 3. measure distances in real life 4. accuracy score: % error of measurements averaged over all measurements
Navigation to Waypoints	The robot must be able to pathfind to specified waypoints, then navigate to those points.	1. mark out a target position and orientation on floor using tape 2. teleoperated robot to point 3. record position using tf_echo 4. teleoperate the robot away 5. send the robot to the waypoint using navigation goal topic 6. measure how far away the robot is from the goal in real life (angle and distance)

		7. Scores: a. accuracy: mean absolute error in distance and angle b. precision: absolute average deviation of distance and angle

4.2.3 Perception and Grasping

Manipulation	The robot must be able to pick up and place the drinking vessel without hitting any obstacles.	**Evaluation Procedure:**
Grasping	The robot must be able to pick up the drinking vessel.	1. position robot in front of table and can on table 2. run grasping code 3. score: # of successful grasps / # of attempted grasps
Drinking Vessel Detection	The robot must be able to identify the location of the drinking vessel on the tabletop using its sensors.	1. position robot in front of table and can on table 2. run perception code 3. score: # of successful can detections / # of attempted perceptions
Placing	The robot must be able to place the vessel on a table.	1. position robot with can in hand in front of table 2. run placing code 3. score: # of successful places / # of total places
Manipulation Obstacle Perception	The robot must be able to identify obstacles with its sensors and avoid hitting them with the arm.	We did not implement this functionality, so we will not test it

4.2.4 Human Interaction

Stakeholder Interaction	The robot must fulfill the needs of the stakeholders.	**Evaluation Procedure:**

Hydrating Residents	The robot must increase the hydration level of the nursing home residents	1. Track the number of glasses of water the residents drink per day before the robot is introduced. 2. Introduce the robot. 3. Track the number of glasses of water the residents drink per day after the robot has been introduced. 4. Compare before and after the robot was introduced.
Saving Caregiver Time	The robot must reduce the amount of time nursing home staff must spend on hydrating the residents.	1. Track the number of hours staff spend each day on hydrating the residents before the robot is introduced. 2. Introduce the robot. 3. Track the number of hours staff spend each day on hydrating the residents after the robot has been introduced. 4. Compare before and after the robot was introduced.
Saving Operator Money	The robot must reduce the overall cost of running the nursing home.	1. Record how much the nursing home spends on hydrating residents (including staff wages) before the robot is introduced. 2. Introduce the robot. 3. Record how much the nursing home spends on hydrating residents (including staff wages) after the robot has been introduced. 4. Compare before and after the robot was introduced.

5 Evaluation Results

See the complete results (detailing each test run and including additional notes/discussion) in the

Appendix.

5.1 Safety

Safety	The robot must operate safely around humans.	**Evaluation Results:**
Safe Navigation	The robot must not run into obstacles (including humans) when navigating.	Score: 5/5 (but was very close to failure on many attempts) The robot did not run into anybody or anything when navigating.

5.2 Autonomous Navigation

Navigation	The robot must autonomously navigate the nursing home without running into any obstacles.	**Evaluation Results:**
Navigation Obstacle Perception	The robot must be able to detect obstacles with its sensors and avoid running into them.	See Safe Navigation from the Safety section
Localization	The robot must be able to determine its position using sensors.	Total: mean percent error: 13.38 % The robot was decent at locating itself. Perhaps it would have worked better with a more accurate map.
Mapping	The robot must be able to create a map of its environment to use for navigation.	Success: the robot was able to create a map of GIX in real life We did not end up using this map very much, simply because the coordinates on this map did not match the ones on the original map created in simulation.
Navigation to Waypoints	The robot must be able to pathfind to specified waypoints, then navigate to those points.	Summary statistics: ● Accuracy: mean absolute offset: ○ X: 0.166 m ○ Y: 0.058 m

		● Precision: mean absolute deviation (MAD) of offsets ○ X: 0.0408 m ○ Y: 0.0104 m The robot's navigation was fairly precise, but it was not very accurate. This means that it returned to roughly the same place each time, but it was not quite the place it was supposed to go. We are not sure exactly why that is. The precision and accuracy were worse in the X direction than in the Y direction. Again, we are not sure why that is the case.

5.3 Perception and Grasping

Manipulation	The robot must be able to pick up and place the drinking vessel without hitting any obstacles.	**Evaluation Results:**
Grasping	The robot must be able to pick up the drinking vessel.	Total: 0/5 The robot failed the grasping task each time during our recorded attempts, but it was able to grasp and tuck the can a few times in other tests.
Drinking Vessel Detection	The robot must be able to identify the location of the drinking vessel on the tabletop using its sensors.	Total: 4/5 Despite the images sometimes being a bit corrupted, the code was able to correctly identify the centroid of the can most of the time.
Placing	The robot must be able to place the vessel on a table.	Total: 5/5 The robot was able to place the can every time.

5.4 Notes

When evaluating the robot performance, we included the tucking action as part of the grasping task. In retrospect, perhaps we should have separated those two actions. The tucking action was much more reliable than the grasping action, and the two actions could be performed independently.
Due to time and COVID-19 related constraints, we did not evaluate the robot on the human-centered (aka stakeholder) requirements.

6 Gap Analysis and Conclusion

6.1 Gap Analysis

We identified many issues that we would address if we had more time to work on this project.
On the object manipulation side of things, one issue we only partially addressed was obstacle perception and avoidance. We only got octomap working for the placing function in our live demo. We would need more time to make it work with the grasping, as the current implementation adds the can to the octomap which prevents the gripper from reaching around the can. We faced a persistent issue with our perception code; the captured camera image would often be corrupted, making can detection impossible. A third manipulation issue we faced is that the robot would often turn the can sideways and upside down when moving it. Perhaps we could fix this issue with cartesian planning or some kind of orientation constraint.
On the navigation side, our robot would often get stuck in narrow spaces such as doorways and hallways. We tried tuning the costmap parameters, but it didn't really help. We also tried writing a custom navigation script for narrow spaces using laser scan messages, but it was buggy and the robot went off track and bumped a wall.
We had some integration problems and we also simply didn't have time to add all of the features we had planned. One large issue we faced was that the robot could not navigate close enough to the table to reach the can with its arm. Other features we planned to add if we had more time included error detection, failed action retry behavior, and a finite state machine architecture to control the robot's behavior.

6.2 Conclusion

The purpose of this project was to learn about robotics development. Of course, we learned how to use ROS and the Fetch Mobile Manipulator robot, but we learned much more than that. We learned human-centered design skills, taking into account multiple stakeholder needs and specifications that aren't purely technical. We practiced the design-build-test cycle, where we made iterative incremental improvements to our design and code. We learned new product evaluation skills, such as writing specifications, writing tests, and understanding precision and accuracy. We learned about integration challenges, the problems that come up when you try to put all of your components together. In the end, we were able to present a slideshow overview and a working live demo of our project, showcasing our work to the rest of the GIX community.

7 Appendix

7.1 Videos

Here is a link to a Google Drive folder containing videos that we took during our robot testing:
https://drive.google.com/drive/folders/1UlgJ5DqclGjuOJGvIJNF2Nv_y3r8-Ui3

7.2 Code

Here is a link to the GitHub repository with all of our project code:
https://github.com/ironmanyu/serving_fluids/tree/james-muir/demo-evaluation

7.3 Project Folder

Here is a link to a Google Drive folder containing all of our project documentation:
https://drive.google.com/drive/folders/1RFBDb50C5QkRsa-b9almIxVHuwiKrRSK

7.4 Full Evaluation Results

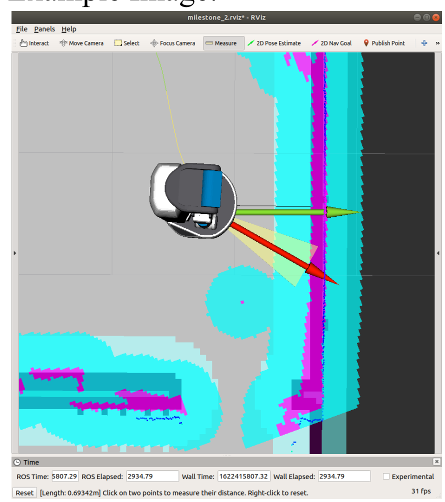
7.4.1 Safety

Safety	The robot must operate safely around humans.	**Evaluation Results:**
Safe Navigation	The robot must not run into obstacles (including humans) when navigating.	1. Success. However, the robot got pretty close to sulaiman, about 1 foot 2. Success? The robot got very very close to sulaiman, almost hit him, only inches away 3. Success? Sulaiman walked in front of the robot as it was moving. The robot did stop, but only inches away. 4. Succes? Robot stopped, but only inches away from Sulaiman, and couldn't find a way past him until he moved out of the way. 5. Success? Robot stopped, but only inches away from Sulaiman, and couldn't find a way past him until he moved out of the way. Notes: The robot drove towards people pretty fast, and only stopped a few inches away from them. If the arm had been extended it would have hit them. It is not aware of the reach of its

		arm when navigating. It also failed to plan a new path around a human, only continuing when the human moved out of the way. Score: 5/5 (but was very close to failure on many attempts)

7.4.2 Autonomous Navigation

Navigation	The robot must autonomously navigate the nursing home without running into any obstacles.	**Evaluation Results:**
Navigation Obstacle Perception	The robot must be able to detect obstacles with its sensors and avoid running into them.	See Safe Navigation from the Safety section
Localization	The robot must be able to determine its position using sensors.	1. Percent Error: 6.8 % a. robot thinks: 0.69 m b. actual: 0.74 m 2. Percent Error: 9.8 % a. robot thinks: 0.46 m b. actual: 0.51 m 3. Percent Error: 17.5 % a. robot thinks: 0.33 m b. actual: 0.40 m 4. Percent Error: 1.5 % a. robot thinks: 0.69 m b. actual: 0.68 m 5. Percent Error: 31.3 % a. robot thinks: 0.22 m b. actual: 0.32 m Total: mean percent error: 13.38 % Limitations: we only used one wall, and only measured distance, not orientation Notes: there was an offset between the laser scan and the map walls in RViz, so it looks

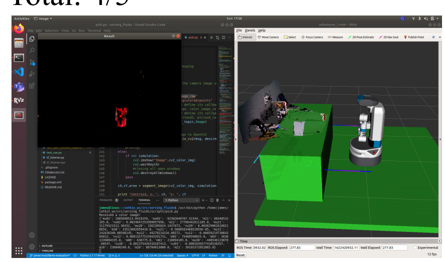
		like the lidar was accurate but the localization was not. Also, we used a map that we generated in simulation based off of a floorplan, not the map we generated at GIX, so that could have impacted the quality of the localization Example Image: 
Mapping	The robot must be able to create a map of its environment to use for navigation.	Success: the robot was able to create a map of GIX in real life
Navigation to Waypoints	The robot must be able to pathfind to specified waypoints, then navigate to those points.	Reported Pose: - Translation: [-3.038, 0.078, 0.000] - Rotation: in Quaternion [0.000, 0.000, -0.021, 1.000] in RPY (radian) [0.000, 0.000, -0.042] in RPY (degree) [0.000, 0.000, -2.426] Results: 1. Trial a. X offset: +0.20 m b. Y offset: +0.04 m 2. Trial a. X offset: +0.20 m b. Y offset: +0.05 m 3. Trial a. X offset: +0.12 m

		b. Y offset: +0.06 m 4. Trial a. X offset: +0.11 m b. Y offset: +0.07 m 5. Trial a. X offset: +0.20 m b. Y offset: +0.07 m Summary statistics: ● Accuracy: mean absolute offset: ○ X: 0.166 m ○ Y: 0.058 m ● precision: mean absolute deviation (MAD) of offsets ○ X: 0.0408 m ○ Y: 0.0104 m Limitations: We didn't record angle offsets because we couldn't come up with a good way to measure them.

7.4.3 Perception and Grasping

Manipulation	The robot must be able to pick up and place the drinking vessel without hitting any obstacles.	**Evaluation Results:**
Grasping	The robot must be able to pick up the drinking vessel.	1. Run a. Notes: Failed to move gripper in front of can, hit can b. Success (Y/N): N 2. Run a. Notes: Failed to move gripper in front of can, gripper moved above can instead of around can b. Success (Y/N): N 3. Run a. Notes: Arm failed

		to move in front of can or around can, but tried to move above can and hit the can instead b. Success (Y/N): N 4. Run a. Notes: Arm failed to move in front of can, so it hit the can when trying to move around it b. Success (Y/N): N 5. Run a. Notes: failed to get can position b. Success (Y/N): N Total: 0/5 Note: Grasping worked occasionally, but more like 1 in 10 times.
Drinking Vessel Detection	The robot must be able to identify the location of the drinking vessel on the tabletop using its sensors.	1. Run a. Notes: Uncorrupted image, successful extraction of red can b. Success (Y/N): Y 2. Run a. Notes: Uncorrupted image, successful extraction of red can, but 3D position might have been off b. Success (Y/N): Y? 3. Run a. Notes: Uncorrupted image, successful extraction of red can

		b. Success (Y/N): Y 4. Run a. Notes: Uncorrupted image, successful extraction of red can, b. Success (Y/N): Y 5. Run a. Notes: [ERROR] [1622422589.173801]: Unable to find the transformation from head_camera_rgb_optical_frame to base_link b. Success (Y/N): N Total: 4/5 
Placing	The robot must be able to place the vessel on a table.	1. Success 2. Success 3. Success 4. Success 5. Success, but robot hit itself with the can Total: 5/5