CaBot

Team G - Carnegie Malones

Yanda Huang
Jennifer Isaza
Divya Kulkarni
Chris Song
Andrew Wong

Sponsor: Dr. Kris Kitani, IBM

Final Report
May 29, 2018

Abstract

Navigating in an unfamiliar indoor environment can be extremely challenging for the sighted as well as the visually impaired. Blind people have extreme difficulties localizing in any environment and is extremely prone to any danger in their surrounding area, such as sharp objects as well as ledges. With the robot presented in the following report, we hope to not only assist visually impaired with localization, but also guide them to their desired destinations and alert them of their surroundings.

CaBot is a robot capable of guiding visually impaired users through crowded indoor places, such as an airport, a hotel or even a university. It is fitted with various sensors to handle the dynamic environment that we live in today. We use a LIDAR and Kinect to detect obstacles and pedestrians in the user's path. The sensing platform, consisting of the touch sensor and the IMU, can accurately predict CaBot's orientation and the user's proximity. Lastly, CaBot sits on top of a differentially drive platform that allows for good maneuverability.

In the following report, we will be discussing the design process for CaBot in great depth as well as any pertinent project management aspects that kept us on track. Finally, we will end this report with what we have learned as well as any future work that can be done on the platform.

Contents

1. Project Description

CaBot is an assistive robot designed to help visually impaired users navigate through a new environment. The user will use this suitcase-shaped robot to aid them in traversing new indoor environments. Many studies and experiments have shown that visually impaired users struggle with traditional navigation techniques [1]. Users still use guide dogs and canes in order to navigate areas. However, these methods do not provide essential information about the path the person needs to travel or their final destination. There are also communication issues with these type of solutions. For example, guide dogs, although highly trained, are unable to consistently communicate with their users about their intent. The visually impaired users, in turn, have problems relaying their wishes to the guide dog.

One current solution that is being tested is NavCog [2], a phone app based navigation assistant specifically designed for the visually impaired. It has some limitations as it is unable to provide information on obstacles in the path that have not previously been mapped. Visually impaired users also prefer having humans to guide them and having something grounded to trust. We hope to provide a solution for all these shortcomings in the current solutions to guide and assist blind users. CaBot needs to be in a symbiotic relationship with its user to provide an effective platform to guide them through new indoor environments.

We envision the solution to be of no hassle to the user and available to pick up in public areas where they will be necessary. This alleviates the burden of visually impaired people having to carry around and store their canes or take care and support their dogs.

2. Use Case

Maria, a visually impaired woman from Pittsburgh, is on the way to Pittsburgh International Airport to catch a flight home to her daughter's wedding. She currently uses a cane on a daily basis in order to walk around to the destinations she needs to reach, but she has never been to this airport before. Maria is becoming nervous about navigating on her own in this unfamiliar and crowded location. As she leaves her Uber, Maria uses her cane to feel for the curb and enters through the airport's automatic doors. An attendant by the door asks if Maria needs any assistance arriving to her gate, and offers help from their new assistive robot, CaBot, which Maria decides to accept, as depicted in Figure 1.

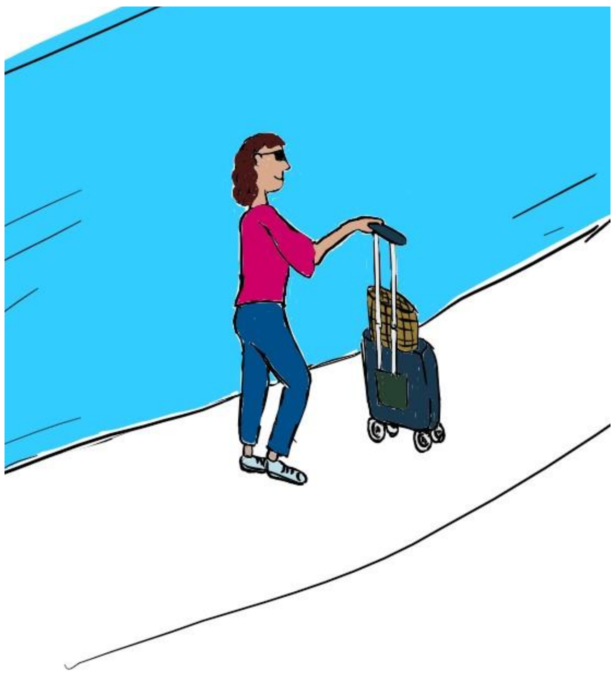


Figure 1 – Artist's depiction of CaBot airport operation.

Pittsburgh International previously mapped out the floorplan of the airport for CaBot's navigation and localization requirements. The attendant helps Maria enter her flight details into the CaBot user interface. The interface alerts the flight attendants to the arrival of Maria, will update CaBot with any new gate changes, and tells Maria which gate they will be navigating to. Maria places her purse and foldable cane into the basket on top of the robot. CaBot asks Maria if she would like a functional overview to describe all of the CaBot features, which she agrees to by verbally stating "Yes". The overview recommends Maria to push CaBot in front of her in order to best detect obstacles. Voice commands that Maria can use are also provided, such as "help desk," "restroom," and "food." Once done, CaBot asks if Maria is ready to proceed to her gate, which she again replies, "Yes." CaBot navigates to the security line available for CaBot users, using motorized wheels, slowing down adapting to the pedestrians in front of the robot. On the way to the security line, there is a suitcase in the path of CaBot, which verbally warns Maria and vibrates the handle once it is about to turn to go around the obstacle. Once the obstacle is out of the path, the handle vibrates again and a "ping" is heard, letting Maria know that the way is clear again.

Once Maria gets through security, she says "time" to CaBot, which responds, "Your flight leaves in 45 minutes, the current time is 4:55PM." Maria decides that there is time for food, and says "food" to CaBot. CaBot states, "The food destinations around you include: "McDonalds," 3 minutes off route, "Chili's ToGo," 5 minutes off route, and "Macaroni Grill," 10 minutes off route. Would you like to hear more options?" Maria responds "no, add stop, Chili's ToGo." CaBot navigates to "Chili's ToGo," using vibrations before each turn and a ping after each turn is complete. CaBot confirms to Maria when she has arrived. After eating, CaBot continues on the route to Maria's gate, bringing her to the information desk. Upon arrival, CaBot says "You have arrived to your gate, your flight will board in 5 minutes." An attendant helps Maria to an empty seat and Maria returns CaBot to an attendant when boarding her flight. The scenario is depicted in figure 2.

Cabot, add stop, Chili's ToGo
Navigating to Chili's ToGo...
10 minutes added to route

Figure 2 – Artist's depiction of CaBot Mid-destination Navigation

3. System-level Requirements

The system-level requirements are derived from the key solutions that will address the problems that we have found through user interviews and research of current navigational assistants for the visually impaired. The performance and nonfunctional requirements are important aspects of the project that must be met in order to provide an effective assistive robotic platform for the users it will serve.

3.1 Mandatory Performance Requirements

Table 1 – Mandatory performance requirements

M.P.1	Localize within 1 meter in a pre-mapped area
Description	The system should localize in an indoor environment that is pre-mapped and equipped with necessary infrastructure
M.P.2	Detect static obstacles 85% of the time
Description	The system must identify obstacles while executing a global route
M.P.3	Notify user within 2 meters of possible collision
Description	CaBot should give the user feedback on unexpected items in the path of the user and with enough time so they are able to avoid it
M.P.4	Travel up to 2 mph unloaded
Description	CaBot travels at a slow walking pace

M.P.5	Classify moving pedestrians 70% of the time
Description	Be able to classify a moving pedestrian so both CaBot and that the visually impaired user have information to behave appropriately

3.2 Mandatory Non-Functional Requirements

Table 2 – mandatory non-functional requirements

M.N.1	Weigh less than 30 lbs
Description	CaBot must be transportable to different areas
M.N.2	Fit within an average carry-on suitcase dimensions: 22 x 35 x 56 cm
Description	The system should be comparable average luggage so it is convenient to roll in large public areas
M.N.3	Take less than 10 min for user to calibration
Description	The user can input parameters according to their preferences: input desired walking speed, notification style, notification rate
M.N.4	Operate on different surfaces
Description	CaBot will be able to effectively on different common indoor surfaces

3.3 Desirable Performance Requirements

Table 3 – Desirable performance requirements

D.P.1	Localize within 0.5 meters
Description	Ideally we can improve the algorithms to localize more effectively in indoor environments
D.P.2	Detect static obstacles 95% of the time
Description	We would like a high rate of detection of obstacles as to provide the user accurate information about their surroundings
D.P.3	Notify user with audio and haptic feedback within 2 meters of an obstacle
Description	The system would be able to provide multiple types of feedback when giving the user relevant information
D.P.4	Predict other pedestrian trajectories with an error <20%
Description	CaBot will have information of other pedestrian paths to effectively develop a plan for its user
D.P.5	Robot's speed and turning radius adapt to user's behavior after 20 minutes of calibration/ familiarization with robot
Description	The robot will be calibrated and preferences will be set after a short familiarization session

	between the robot and the user

3.4 Desirable Non-Functional Requirements.

Table 4 – Desirable non-functional requirements

D.N.1	Weight less than 20 lbs
Description	CaBot will be very easily maneuverable and convenient to hand carry if necessary
D.N.2	The system will be tested in the Pittsburgh airport
Description	The final prototype will be tested in a real environment
D.N.3	Cost of final prototype will be less than $5000
Description	The prototype will be at a cost that will allow businesses to buy it

4. Functional Architecture

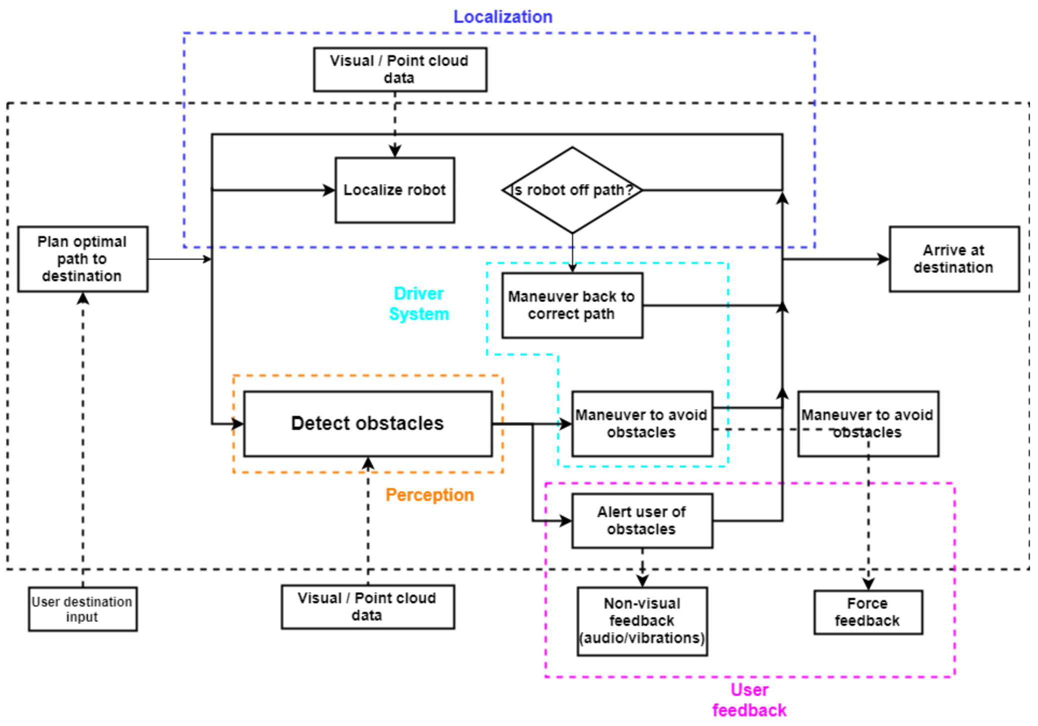

Localization
Visual / Point cloud data
Localize robot
Is robot off path?
Plan optimal path to destination
Arrive at destination
Driver System
Maneuver back to correct path
Detect obstacles
Maneuver to avoid obstacles
Maneuver to avoid obstacles
Perception
Alert user of obstacles
User destination input
Visual / Point cloud data
Non-visual feedback (audio/vibrations)
Force feedback
User feedback

Figure 3 - Functional architecture diagram. Subsystems are in dotted lines.

The whole system consists of four subsystems: indoor localization system, obstacle perception system, driver system and user feedback system.

The user activates the system by inputting their final destination. The system should plan an optimal path to the destination and store the plan. The user holds onto the robot's handle, and CaBot begins moving, guiding the user to his or her destination. On the way, the indoor localization system and obstacle perception system should work simultaneously in a loop, sending commands to the motor system as well as the user feedback system. The indoor localization system uses sensor data to determine CaBot's location, which is used to plan around known obstacles.

The obstacle detection system assists the user to avoid collision with obstacles or pedestrians by analyzing point cloud data or visual data from the LIDAR and stereo cameras. If the system detects a nearby obstacle, the drive system to lead the user away from any obstacles. In the meantime, the user feedback system will also be initialized to alert the user of the existence of obstacles.

The drive system acts as the main actuator of the robot. It applies gentle force to the robot so that the robot can accelerate or steer. These actions indicate where the user should go, thus leading the user to the final destination. The user feedback system alerts the user to the existence of any pedestrian ahead and upcoming turns. The system accomplishes this through voice feedback.

5. System-level Trade Studies

The primary two subsystems of CaBot under analysis are human computer interaction and the drive system. These are two areas that will directly affect the user the most, with the highest number of options available for consideration. Both trade studies are weighted in the favor of the user experience and safety, which are directly associated with the functional and non-functional requirements. In practice, we opted for different design choices than presented in our trade study.

5.1 Motor Movement Trade Study

Originally, our team planned for CaBot to be partially autonomous. Throughout the route, CaBot would be pushed along by the user unless CaBot needed to avoid an obstacle or correct the user's trajectory. This solution avoids complete reliance on localization accuracy while protecting the user from any obstacle that are in the path of the user. Corner correction may also be implemented, because a visually impaired user may not turn at the correct angle even if CaBot indicates there is a turn ahead. Except for these two cases, the user would be able to listen to CaBot for directions and push the robot along.

We decided to switch to full autonomy for two reasons. Chieko, one of our sponsors, indicated that she would be more comfortable with a fully autonomous robot. Additionally, a fully autonomous robot could be mechanically simpler than a partially autonomous robot. The wheels would not have to disengage from the motors while the user is pushing CaBot.

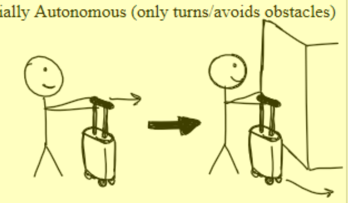
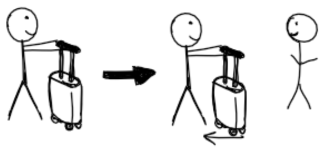
	Is Affordable (10%)	Is Easy to Implement (15%)	Is Portable (15%)	Comfortable for User (30%)	Protects User (40%)	Total	Rank
Push	5	5	5	3	1	3.3	4
Joystick	3	2	3	3	3	3.15	5
Fully Autonomous	1	1	2	4	5	3.75	2
Partially Autonomous (only turns/avoids obstacles) 	1	3	2	4	5	4.05	1
Braking Only 	4	4	4	3	3	3.7	3

Figure 4 - Weighted trade study for the motor drive subsystem

5.2 User Communication Trade Study

Originally, our team also planned to include a microphone in the handle. The user communication trade study seen below in Figure 5 resulted in a decision to incorporate a microphone into the handle of CaBot. Integrating a microphone on the handle will allow voice commands to be the primary form of communication without reliance on hardware that requires sight. We chose to use a touch interface instead of voice recognition, meaning we did not need a microphone in the handle. We chose to use a touch interface since it was easier to implement, and it didn't require us to use external voice recognition libraries. Additionally, voice recognition could be inaccurate, and may not work well in noisy test environments.

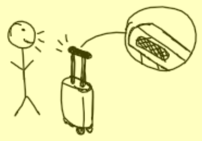
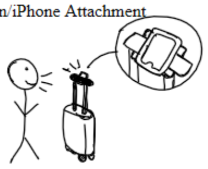
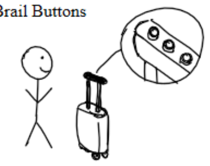
	Is Affordable (10%)	Is Easy to Implement (10%)	Is Portable (20%)	Easy to Use (30%)	Effectively Communicates to CaBot (30%)	Total	Rank
Microphone on User	5	5	2	2	4	3.2	3
Microphone on Handle 	5	4	4	4	3	3.8	1
Touch Screen/iPhone Attachment 	1	2	4	1	4	2.6	4
Brail Buttons 	5	4	5	4	2	3.7	2

Figure 5 – Weighted trade study for the human computer communication interface.

6. Cyberphysical Architecture

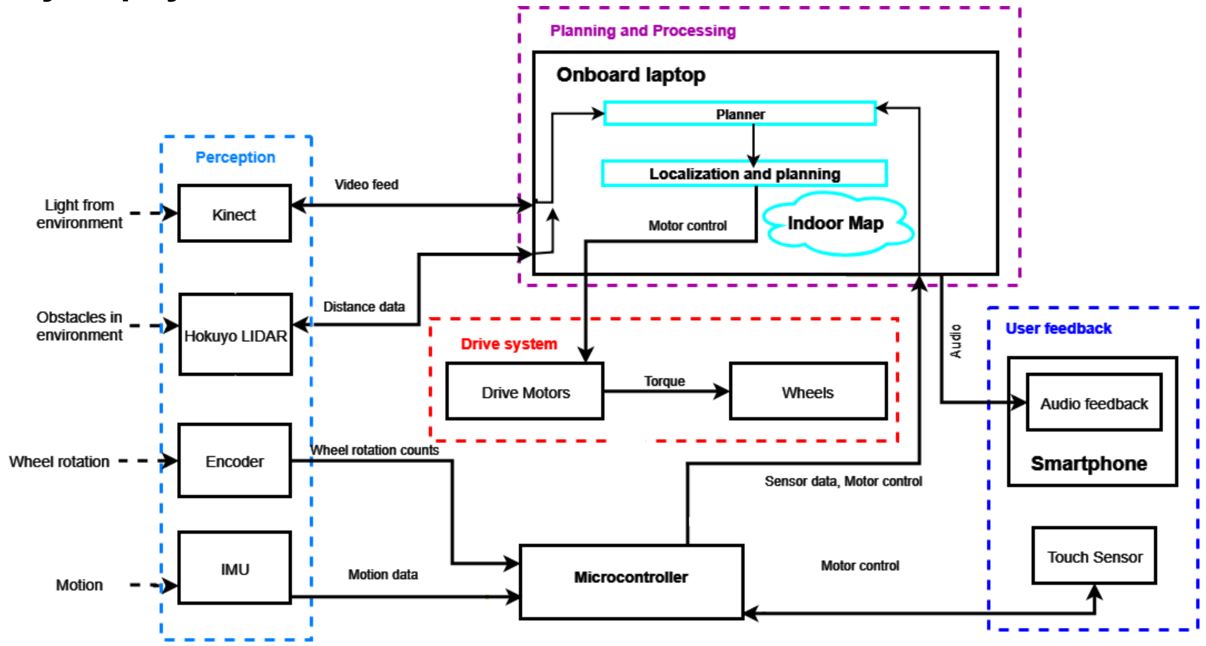


Figure 6 - Cyberphysical architecture. Each subsystem is grouped by dotted lines, and software components are the boxes highlighted in light blue.

The figure above shows our cyber-physical architecture. Our perception subsystem will take the laser scan from the LIDAR to localize within the building as well as detect any obstacle in front of the robot. The Kinect is only being used for the sake of pedestrian detection because there are many available packages for the Kinect. Localization is performed based on the data received from encoder, IMU, and LIDAR. By leveraging the AMCL particle filter, we are capable of fusing those information and accurately localize within the building. Moreover, CaBot was mobilized by our dedicated brushed driver. With a differential drive platform, CaBot can make tighter turns around corners, thus enhancing its mobility. The aforementioned subsystems works very closely with the planning and processing subsystem. Essentially, all the perception information is transferred to the planning system, where a command will be given to the motors, propelling CaBot forward. Finally, the HCI node takes care of any interaction between the user and the robot. The touch sensor will be able to tell whether the user is actually using CaBot and the audio feedback gives prompts regarding the environment, such as obstacles, nearby pedestrians and upcoming turns.

7. System description and evaluation

7.1 Subsystem/system description and evaluation

CaBot comprises of five subsystems: indoor localization, indoor obstacle detection, drive system, audio and haptic user feedback, and path planning and sensor processing. Each subsystem communicates with other subsystems, as depicted in Figure 6.

7.1.1 Indoor Localization

For the first iteration, CaBot used BLE beacons and a pre-generated map of the building to localize itself indoors. Small BLE beacons placed in the hallways of the building and their position recorded in a map were used to localize. Each BLE beacon broadcasts a unique ID, and CaBot measured the Received Signal Strength Indicator (RSSI) of the surrounding BLE beacons to determine its position. CaBot fetched a map of the current building from a map server, which marked the location and floor of every BLE beacon and includes points of interest (POI). CaBot used the BLE ID in conjunction with the map to localize itself. With this method the localization accuracy we got was around 1.5 meters which was not good enough to successfully navigate through the hallways. Our requirements needed CaBot to localize within 0.5 meters so we had to change our localization method.

One alternative we considered to BLE beacons was to use Wi-Fi signal strength. This approach would allow CaBot to enter new buildings without requiring beacons to be placed beforehand. The main drawback of Wi-Fi is that we can't guarantee strong signal strength in all parts of the building [3].

CaBot currently uses the Hokuyo laser scanner to localize in indoor environments. We must map the area where CaBot will operate. The user can use the remote control user interface, that is further described in the HCI section, to move the robot slowly throughout the area. CaBot is compatible with the ROS packages *gmapping* and *hector_mapping* for creating and saving a map. Our system currently uses the *amcl* ROS package to localize. This package uses the map that we created, laser scans and transform messages to accurately estimate pose. The transform comes from the odometry information which includes encoder and IMU information. The rehaul of the localization subsystem helped CaBot become a much more robust, stable platform. It integrated well with the rest of the navigation subsystem as we transitioned to using the ROS navigation stack.

7.1.2. Perception

The perception system's job is to avoid obstacles and detect pedestrians. CaBot uses two different sensors for perception: a laser rangefinder for obstacles detection and Kinect One for pedestrian detection. The Hokuyo laser scanner registers the distance it detects on the costmap. Since the Hokuya has a high scan frequency, it is possible to detect obstacles that appears suddenly. The pedestrian detection is implemented with NiTE toolbox, which is an advanced and robust 3D computer vision middleware. The NiTE algorithm uses RGB data from the Kinect, utilizing the OpenNI2 SDK. NiTE2 extracts the skeleton information to track the skeleton precisely.

7.1.3. Drive System

The drive subsystem consists of two geared DC motors used to guide the user around obstacles. There is one motor on each side of CaBot in a differential driver configuration. Initially we were using stepper motors, however these motors did not get up to the speed we wanted. They were hitting their resonance frequencies and locking up at low speeds. We swapped the stepper motors for brushed DC motors, which provided much more consistent performance. In our first iteration of CaBot, we used an L298 motor driver which lacked overcurrent protection. As a result, we blew out the motor driver several times in both semesters. We decided to switch to a Sabertooth motor driver, which can produce more current than the motors can draw during stall. Additionally, it has over current protection. It has proven to be much more reliable and effective for our platform.

We have designed and integrated a PCB (figure 7) that combines power, sensors, and segments of our drive system on a single board. We went through several iterations of the PCB and settled on a design that was more modular than we had initially planned. Initially, we planned to integrate the motor driver and sensor breakouts for the touch sensor and IMU on a single board. However, we decided to separate the motor driver from the PCB due to multiple motor driver burnouts (as mentioned above) and a design error in the original PCB. Our rationale for making the PCB more modular was to decrease repair times: if the motor driver burned out, we can easily swap it out with a spare motor driver. Our original design had an insufficiently small separation between the traces and the ground plane. As a result, 5V and ground were shorted together. In the final design, we removed the ground plane and the L298 motor driver chip. Additionally, when populating the second PCB design, we made sure to test each section of the board separately before continuing to solder components onto the board.

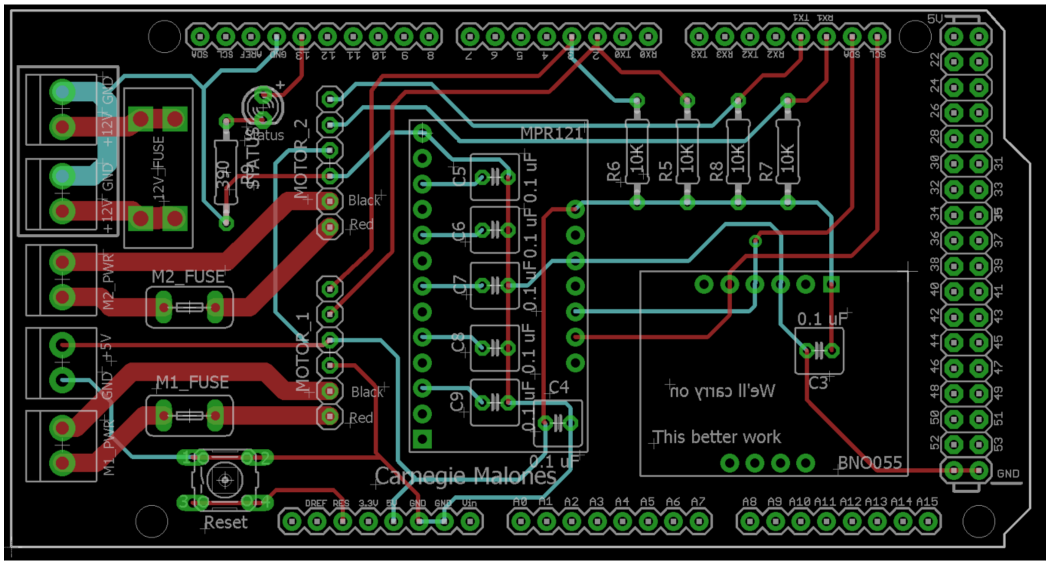


Figure 7. PCB layout

7.1.4. HCI subsystem

As an assistive robot, CaBot must effectively communicate with its user to be valuable in its role. CaBot provides audio directions to the user, such as “turn left in 6 feet.” Additionally, CaBot alerts the user immediately before they need to change their direction, or if there is a pedestrian in the direct path of the user. We are using user’s mobile device to the interface between the user and CaBot. Our team must consider the volume of ambient noise in the operating environment, and whether the user can hear the verbal commands over background noise. We have a web interface in which the user will be able to choose a destination. The sound is also emitted from the phone so that it is close to the user and appropriately loud. The design for this interface was inspired by the assistive touch interface on iOS. The user has access to a dial that will relay the current selected destination through voice. Once the user hears their desired destination they can let go and CaBot will navigate safely to the final destination. Figure 8 below shows the CaBot also is equipped with a touch sensor on the handle. This is currently used to make sure that CaBot is only moving when the user is touching the handle so that it does not run away from its user if they decide to stop.

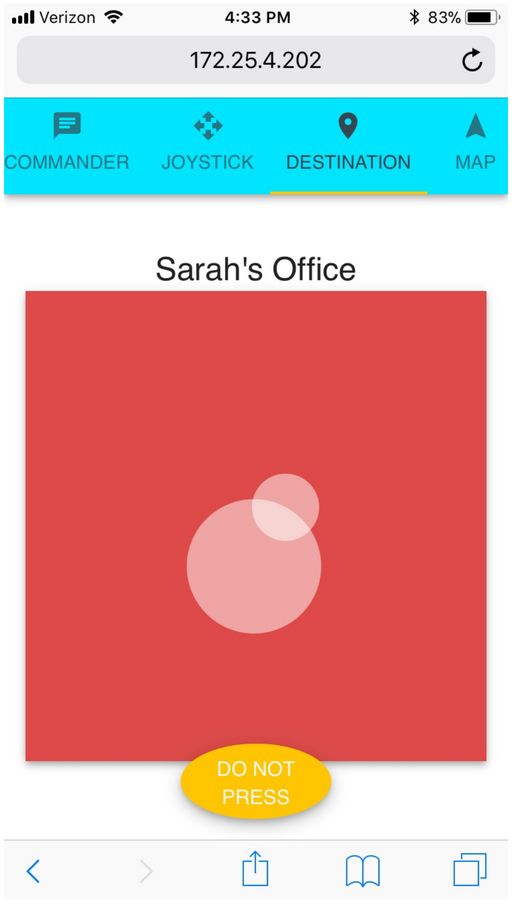


Figure 8 - User interface on smartphone.

7.1.5. Path Planning and obstacle avoidance

The planning subsystem creates a global plan from the current position to the goal position and does local planning to keep on its path and avoid obstacles on the way. This subsystem is tied very closely with the localization system as the planner continually gets estimates for the position to make sure we are able to get to the final destination. For our first prototype, our team integrated the preexisting NavCog app, developed by Dr. Kitani's lab, into our robot to provide a global plan. The phone provided path data that the laptop uses to help decide where the robot should move. We had to modify certain parts of the app to be compatible with CaBot. For instance, the app assumes the user will be walking, so it counts steps (like a pedometer) to assist in measuring how far the user has traveled. We replaced the step counting functionality with a feature that counts wheel rotations using encoders. Our first iteration used a custom built local planner that was basic and did not account for or re-plan for unexpected obstacles in the path. This was a basic PID controller on the trajectory. This was when we were using the bluetooth beacons for localization and therefore had to rely heavily on the encoders to get good estimates of pose.

We completely overhauled the planner during our next iteration to work well with our newly modified LIDAR localization subsystem. This was designed to integrate completely with the ROS navigation stack. We are using the navfn package as the global planner. This is a fast interpolated navigation function that creates plans for CaBot using a costmap from the created map to find a good path from the start to the end point. This package uses Dijkstra's algorithm to find the path. In order to keep the local plan and navigate around unexpected obstacles we are using the dwa local planner package in ROS. This uses a dynamic window approach which samples a possible forward trajectory of the robot and then predicts what would happen in that time frame. It computes the cost of the many of these simulated forward steps and then chooses the lowest cost commands to send to the mobile base. It continues to do this as it makes its way through the global plan. There were a lot of parameters we tuned with respect to these planners. Some of the main parameters we tuned were the maximum and minimum velocities and accelerations to make sure the user would have a smooth path. We also had control over the path distance bias and the goal distance bias. This specified how much CaBot would be allowed to deviate from its original route and how often it would decide to replan if it was too far off from the intended route.

7.2. Performance Evaluation against SVE

Our SVE performance was measured against our SVE test plan. The plan ensured that we fulfilled the relevant requirements for CaBot. Our test plan procedure is shown below in table 5 - CaBot met every performance measure in the test plan (highlighted in green). The SVE's objective was to validate that CaBot could guide a stand-in for a blind user indoors in a simulated use case.

Table 5. - SVE Procedure

Step	Description	Performance Measures
1	A user is deployed with CaBot on NSH 4th floor.	The user is able to start CaBot through a smartphone interface.
2	The user inputs a destination on the 4th floor of NSH.	CaBot receives the destination and begins moving in the correct direction.
3	CaBot localizes itself and begins navigating to destination. We assume that the user holds CaBot gently without leaning on CaBot or picking and placing it.	CaBot localizes itself with an error of 0.5 meters or less.
4	CaBot will avoid slow-moving and stationary obstacles.	CaBot stops or avoids obstacles that are no smaller than 2 ft. tall, 1 ft. wide, 0.5 ft. deep (placed on the ground) that are 4 meters or less from CaBot
5	CaBot will notify user of slow-moving pedestrians in the path and slow down or stop if they are 2 meters or less away. Pedestrians are assumed to be facing forward, approaching CaBot slowly.	CaBot will announce the number of pedestrians in front of it at a range of 2 meters. CaBot will stop for stationary pedestrians less than 1 meter away and will decrease its speed for pedestrians with distance of 2 meters or greater.
6	While navigating, CaBot will notify the user of upcoming turns.	CaBot gives auditory feedback at least 2 meters before sharp (90 degree) turns.
7	Upon arriving at the destination, CaBot indicates that the route is completed.	The Euclidean distance between CaBot's final position and the destination is less than 2 meters.

Verification Criteria
The test is successful if CaBot is given a final destination that is navigable with a safe path, and the user successfully navigates to his/her destination without running into any obstacles in the longitudinal direction.

7.3 Strong and Weak Points

CaBot's strong points include localization accuracy, pedestrian tracking, and human-robot interaction (HRI) features. Using an IMU, encoders, and LIDAR for localization, CaBot can safely navigate through the hallways with an accuracy of less than 0.5 meters. This is a huge improvement over the accuracy of the Bluetooth localization we were using the first semester. Additionally, CaBot has several HRI features that are designed to assist the blind user. For instance, the web interface does not require the user to type their destination. Instead, the user can drag the a circle to their desired destination. When planning around obstacles, CaBot's footprint account for the user standing on the right side of CaBot. Therefore, CaBot should not travel into tight spaces that are wide enough to fit only CaBot and not the blind user. Also, the handle has a touch sensor so CaBot will stop moving if the user lets go of the handle. This features prevents the blind user from losing CaBot if they let go of the handle. Lastly, CaBot is covered in a suitcase shell to better simulate the final product and to conceal its internal electronics, drawing less attention to the blind user.

Weak points include limited computing power, slow replanning times, and difficulty servicing mechanical components. Since the laptop had to fit underneath the suitcase cover, we were forced to use a smaller laptop with less computing power. As a result, the planner was often missed its target running frequency. This contributed to CaBot's slow replanning times when approaching obstacles with barely enough room for CaBot and the user to fit through. To remedy this issue, we can use a more powerful laptop. We will need a better cooling solution for the laptop (such as fans) if CaBot operates with the suitcase shell. Another solution might be to try out other ROS planners. However, it is difficult to balance the strengths and weaknesses of each planner. For instance, DWA planner is better at avoiding new obstacles, while the TEB planner creates a smoother path but is unable to react to new obstacles quickly. Lastly, CaBot's drivetrain and interior components are difficult to access and repair. For instance, to tighten some hard to reach screws, the entire drive train assembly needs to be disassembled. When the suitcase shell is fitted over CaBot, the battery and power buttons are difficult to reach. This can be fixed by fitting the power button and battery disconnect onto the suitcase shell.

8. Project management

8.1 Evaluation of Schedule

Table 6 and table 7 are the proposed work plan for CaBot and associated schedule for each task. The tasks that were not completed by the planned date are highlighted in red. Three iterations of the robot were planned (alpha, beta, gamma), adding complexity and functionality to CaBot. The alpha phase was not motorized, while the gamma phase was an upgraded version of the beta phase, retaining mostly the same hardware. Since we made several significant changes to our schedule after the fall semester, we have included the schedule we actually followed for the spring semester (table 8).

We followed our proposed schedule closely for the fall semester. Major uncompleted tasks in the fall semester included: beginning development of CV algorithms for object detection and avoiding static obstacles during FVE. These tasks were moved to the spring semester. During the spring semester, we deviated from the original schedule that we planned during the fall. The largest changes in the spring semester schedule stemmed from replacing BLE localization with ROS Navigation stack for planning, localization, and mapping. For example, integrating the ROS Navigation stack took less time than we had anticipated. However, tuning parameters for planner and other components took much longer than we had anticipated. Because we didn't use BLE localization in our final project, we didn't need to setup BLE beacons in our test area. Lastly, we held our SVE in NSH instead of the Pittsburgh Airport to make SVE rehearsals easier.

Overall, we did a good job of scheduling tasks during the spring semester. Several tasks scheduled for the fall semester, such as CV and obstacle avoidance were too ambitious. Some team members' course loads were much higher than anticipated, so we scaled down some aspects of our project as appropriate.

Table 6 - Proposed work plan for both semesters (not accurate for spring semester).

MILESTONES	TASKS
1. Defining need statement and requirement	1. Research into a particular problem that needs to be solved with robotics 2. Seek sponsorship (academic/industrial) 3. Formulate need statements 4. Formulate functional requirements 5. Formulate non-functional requirements
2. High Level Hardware Architectural Design	1. Define components within actuation platform 2. Define components within compute platform 3. Define components within sensing platform
3. High Level Software Architectural Design	1. Define software infrastructure 2. Define communication protocol between different sensors 3. Determine algorithm for computer vision, planning and sensor fusion
4. Defining Human – Computer Interaction Interface	1. Research on interactions between robotics and the visually impaired 2. Engage in discussions with experts (professors and postdocs) 3. Prepare a survey/study on the subject
5. Data Collection/Annotation	1. Collection data to be later used for algorithm development 2. Annotate data for machine learning
6. Alpha – phase Chassis Fabrication	1. Chassis Design & determine sensor mount locations 2. Purchase parts to construct a chassis for rapid prototyping 3. Construction of Chassis
7. Alpha – phase Electrical/Sensor Integration	1. Purchase Sensors, breakout boards and microcontrollers 2. Sensor calibration 3. Fit initial sensor suite (iPhone & encoder) onto robot
8. Alpha – phase Software Integration with NavCog	1. Setup communication between ROS and iOS 2. NavCog Integration for localization & HCI 3. Setup encoder input from robot

9. Alpha – phase Validation	1. Receive localization information from Bluetooth beacon 2. Validate HCI with participants 9.3 Ensure global path planner is operational
1. Reflection & Planning	1. Extrapolate lessons learned from previous phase 2. Make changes if needed
11. Beta – phase Electrical/Sensor Integration	1. Integrate a camera and a LIDAR into existing sensing platform 2. Integrate motor drivers and encoders into the system
12. Beta – phase Software implementation	1. Setup communication between existing hardware from alpha and new sensors 2. Develop and integrate computer vision algorithms for object detection 3. Sensor calibration 4. Improve localization algorithm from alpha phase with LIDAR and camera 5. Implement algorithm for local path planning
13. Beta – phase Validation **(FVE)**	1. Ensure the robot is capable of localizing with higher accuracy 2. Ensure the robot can communicate with the user seamlessly 3. Ensure the robot is capable of detect and avoid static obstacles
14. Reflection and Planning	1. Extrapolate lessons learned from previous phase 2. Make changes to hardware or software if needed
15. Gamma – phase Electrical/Sensor Integration	1. Integrate BLE device into the Robot 2. Integrate Barometer into the Robot 3. Integrate IMU into the Robot
16. Gamma – phase Software integration	1. Setup communication between new sensors and ROS 2. Sensor calibration 3. Port localization algorithm from iPhone to ROS 4. Improve algorithm for computer vision, planning, HCI and localization
17. Acquiring Map Data of Pittsburgh airport	1. Attach Bluetooth beacons across Pittsburgh Airport 2. Generate map of Pittsburgh airport 3. Manually assign weights to local paths 4. Generate RSSI fingerprinting information of various locations within the airport
18. Gamma – phase Validation **(SVE)**	1. Ensure the robot is capable of avoiding slow moving pedestrians 2. Ensure the robot can interact with the user seamlessly 3. Ensure the robot is capable of guiding the user to their destination from a voice command

Table 7 – Proposed task scheduling for both semesters (not accurate for spring semester).

DATE	MILESTONE	TASKS				
Sept. 20	1. Defining need statement and requirement	1.1	1.2	1.3	1.4	1.5
Sept. 24	1. High Level Hardware Architectural Design	2.1	2.2	2.3	**CoDR**	
Oct. 1	1. High Level Software Architectural Design	3.1	3.2	3.3		
Oct 10	1. Defining HCI Interface	4.1	4.2	4.3		
Oct 15	1. Data Collection/Annotation	5.1	5.2	**PR1**		
Oct 18	6. Alpha – phase Chassis Fabrication	6.1	6.2	6.3	**PR2**	
Oct 21	1. Alpha – phase Electrical/Sensor Integration	7.1	7.2	7.3	**PDR**	
Oct 30	1. Alpha – phase Software Integration with NavCog	8.1	8.2	8.3		
Nov 3	1. Alpha – phase Validation	9.1	9.2	9.3	**PR3**	

Nov 4	1. Reflection & Planning	10.1	10.2				
Nov 10	1. Beta – phase Electrical/Sensor Integration	11.1	11.2	**PR4**			
Nov 24	1. Beta – phase Software implementation	12.1	12.2	12.3	12.4	12.5	**PR 5**
Nov 30	13. Beta – phase Validation (**FVE**)	13.1	13.2	13.3	**PR6**		
Dec 4	14. Reflection and Planning	14.1	14.2	**CDR**			
Jan 20	15. Gamma – phase Electrical/Sensor Integration	15.1	15.2	15.3			
Mar 15	16. Gamma – phase Software integration	16.1	16.2	16.3	16.4		
Apr 15	17. Acquiring Map Data of Pittsburgh airport	17.1	17.2	17.3	17.4		
Apr 30	18. Gamma – phase Validation (**SVE**)	18.1	18.2	18.3			

Table 8 - Spring Semester Task Scheduling

PR #	**Capability Milestones**	**Associated Tests**	**Associated System Requirements**
8 *(Feb 14)*	CaBot can localize itself down to 0.2 meters. CaBot will travel the same path as the FVE.	**8.1** Localization test	DP1
9 *(Feb 28)*	CaBot can **detect** slow-moving dynamic obstacles, such as pedestrians	**9.1** Pedestrian detection	DP4
10 *(Mar 21)*	CaBot can **avoid** static and slow-moving dynamic obstacles	**10.1** Static obstacle avoidance	DP4
11 *(April 4)*	CaBot's user HCI: ● User can select desired destination from NavCog's interface, and destination is sent to CaBot ● NavCog tells user when to turn	**11.1** HCI test 1 - auditory feedback to user	DP3
12 *(April 16)*	CaBot's pedestrian HCI: ● CaBot warns approaching pedestrians to move out of way	**12.1** HCI test 2 - Pedestrian warning	DP4
SVE *(April 25)*	Full system validation	**SVE** Full System Validation	All
SVE Encore *(Mar 2)*	Full system validation	**SVE** Full System Validation	All

8.2 Evaluation of Budget

CaBot's original hardware (Appendix A) was provided to us by Kitani's lab. Therefore, we spent our MRSD budget mostly to replace the original stepper motors/controller with DC motors/controller and associated hardware. The parts purchased with our MRSD budget is listed in Appendix A. We looked for parts that met our requirements, such as cost, power rating, compatibility with an Arduino, etc. If there were several parts that met our requirements, we looked for the component that was the most reliable and easy to use. For instance, we required that our motor driver be able to drive two motors at 12V, 5A per motor. We narrowed our selection down to two motor drivers, but we felt that the Sabertooth was easier to interface with the Arduino. Therefore, we selected the Sabertooth.

Overall, we were successful with replacing the drive system and electronics with parts we ordered and a custom PCB. However, we could have replaced the L298 motor driver with the Sabertooth earlier in the spring semester. This would have reduced the likelihood of motor driver failures during testing later in the semester. Additionally, if we had nailed down the requirements for our sensors earlier, we could have redesigned and replaced the electronics systems in one go, instead of replacing individual parts as needed.

8.2.1 Parts List

Please refer to Appendix A for the parts list.

8.3 Evaluation of Risk Management

Our final risk mitigation table and risk matrix is shown below in table 9 and figure 9, respectively. The risks highlighted in red are the most consequential and most likely to occur. To mitigate risks, we looked at the most critical parts of our system (sensors, motor driver, time available) and determined how we would best mitigate the risk by finding a work around or solution. Most predicted risks in the table closely follow the actual risks we encountered during the project. However, some risk likelihoods were exaggerated. For example, we had ample time to test before SVE, and pedestrian tracking was able to work. Our biggest scheduling risk was the limited time to work on CaBot due to higher than expected course loads. To mitigate this risk, we scaled back portions of our project. We performed the SVE in NSH instead of Pittsburgh airport, and removed BLE localization from our final project.

Our biggest technical risks were related to either hardware failing or malfunctioning. As mentioned before, the motor driver failed several times throughout both semesters. Additionally, integrating the PCB took longer than we had thought due to design mistakes and changing requirements. Our team could have mitigated the risk of motor driver failure by swapping replacing our motor driver sooner in the project. We could have been more flexible in our hardware design choices, and more willing to try out different components.

Table 9 - Risk likelihood table.

ID	Risk	Requirement	Type	Likelihood	Consequence	Mitigation Strategy
1	Motor drivers break	MP5	Technical schedule	50%	75%	Modularize design of motor driver circuit
2	CV Pedestrian avoidance fails	DP4	Technical	50%	60%	Use LIDAR to detect pedestrians
3	Sensors (LIDAR, camera, etc.) break	MP2, MP5	Budget, schedule	35%	70%	Check if MRSD/faculty has additional similar sensors
4	**Insufficient testing time before SVE**	**DN2**	**Schedule**	**75%**	**80%**	**Incrementally test after reaching each milestone**

5	CaBot takes more than 10 mins to calibrate preferences	MN3	Technical	20%	30%	Research HCI for blind, survey blind people for HCI suggestions
6	Detects less than 80% of static obstacles	MP2	Technical	40%	75%	Collaborate with other MRSD teams/CMU faculty on obstacle avoidance. Use ROS packages where possible
7	**Team member(s) unable to devote sufficient time to project**	**Requirement owned by team member**	**Schedule**	**60%**	**75%**	**Keep all members updated on each other's progress, so members can fill in for absent member**
8	User cannot understand non-visual feedback	DP3	Technical	40%	75%	Research HCI for blind, survey blind people for HCI suggestions

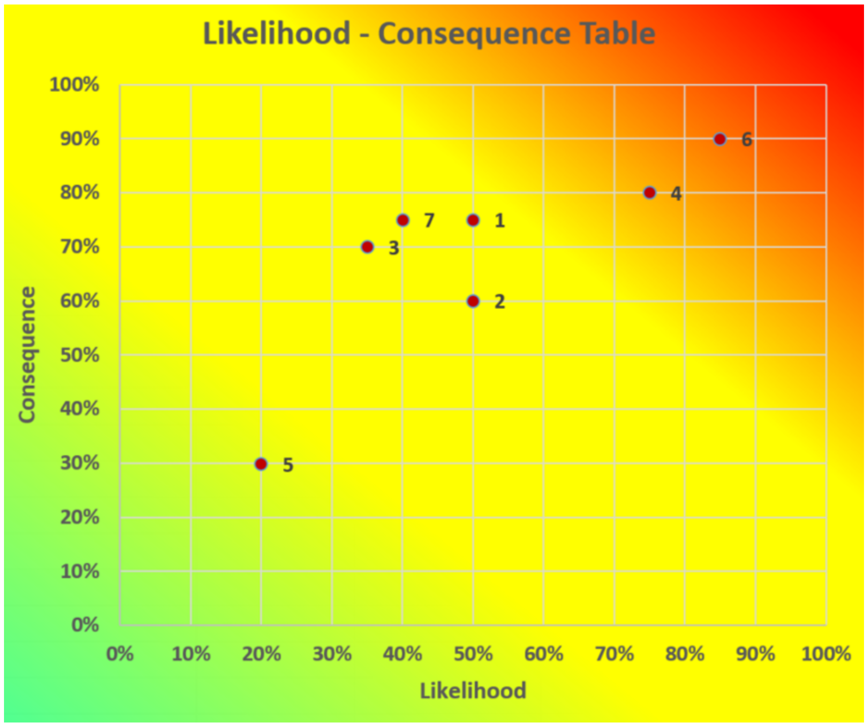


Figure 9. Risk-likelihood matrix.

9. Conclusions

Building CaBot from the ground up was a tremendous experience. We learned many valuable lessons along the way. We will discuss the hardships and what we learned in the section below.

9.1 Lessons Learned

The first lesson we learned is that the simplest solution is often the best solution. In our case, we spent a lot of time adapting NavCog to our application in the first semester when we could have integrated the ROS navigation stack in a matter of weeks. Since localization is required to test all the other subsystems, having inaccurate localization greatly hindered our progress. That precious time could have been allocated to developing other subsystems or spent testing existing ones to make them more robust.

The second lesson we learned was that proper hardware is crucial. We initially started with undersized motors with complex motor controller. The controller's API was difficult to interface with and CaBot was not able to reach human walking speed. To combat this issue, we switched to an L298 motor driver that we used in the sensors and motors lab. Unfortunately, the driver did not have overcurrent protection, leading to repeated failures during test runs. We finally bought a proper motor driver with overcurrent protection, and it has served us well ever since. To reduce wiring clutter and the risk of loose connections, we made our own PCB to be plugged on top of the Arduino as a daughtercard, where the IMU and touch sensor were mounted.

Lastly, the development would have been a lot faster if we started with an off-the-shelf robot and modified the platform to function like a suitcase. A commercial robot platform would reduce the risk of faulty hardware, and would probably include extra features, such as wheel suspension. Additionally, the hardware might be easier to debug. There were many instances in during our project where we struggled to pinpoint the source of a hardware problem. Furthermore, by starting with a pre-made robot, it would make software development and validation go much faster, we could even port everything on a different robot if time permitted.

9.2 Future Work

To start off, the planning aspect of CaBot requires further tuning. As seen from our SVE and SVE Encore, CaBot occasionally had trouble replanning around obstacles in a timely manner. Although CaBot almost always successfully replanned around the obstacle, the local planning should be more streamlined and quicker.

We would also like to add pedestrian prediction to the feature set. Currently, CaBot simply slows down or stops when there are pedestrians ahead, we could accomplish much more if we can predict the trajectory of pedestrians so we can circumvent them in a socially acceptable manner. This will make CaBot act more like a personal helper rather than a robot.

If we are adding more features, we would also need to use a more powerful computing platform. As mentioned before, we would need an active cooling system (such as fans), since the platform is enclosed in a suitcase shell. Since the laptop needed to fit underneath the suitcase shell, our computing platform was a portable ultrabook with limited compute power and no GPU. If we were to switch to a computer with a GPU, we would also have to worry about battery consumption. If the said computer is not a laptop, we would also need to power it from the main battery.

10. References

[1] Enabling Building Service Robots to Guide Blind People. Azenkot, Catherine Feng, and Maya Cakmak. (2016)
http://delivery.acm.org/10.1145/2910000/2906835/p3-azenkot.pdfip=128.237.184.233&id=2906835&acc=ACTIVE%20SERVICE&key=A792924B58C015C1%2E5A12BE0369099858%2E4D4

702B0C3E38B35%2E4D4702B0C3E38B35&CFID=814140114&CFTOKEN=52616633&__acm__=1506542182_6da670e33560e3d77d7b021bb22bafe3

[2] NavCog: A Navigational Cognitive Assistant for the Blind.Dragan Ahmetovic, Cole Gleason, Chengxiong Ruan, Kris Kitani, Hironobu Takagi, and Chieko Asakawa. (2016). In Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16). ACM. *http://www.cs.cmu.edu/~kkitani/pdf/AGKITA-MHCI16.pdf*

[3] WiFi Localization and Navigation for Autonomous Indoor Mobile Robots. Joydeep Biswas and Manuela Veloso. (2010). *http://www.cs.cmu.edu/~mmv/papers/10icra-joydeep.pdf*

Appendix A - Budget

Table 1 - Electrical and Sensing BOM provided by Kitani's lab budget

Component	Manufacturer	Part #	Unit Price	Quant.	Budget	Total Cost
Lidar	Hokuyo	UST-20LX	2855	1	Klab	2855
Stereo camera	Stereolabs	ZED	450	1	Klab	450
Computer	Acer	Predator 15	1654	1	Klab	1654
Stepper Driver	Phidget	1067	95	2	Klab	190
Bipolar Stepper Motor	Phidgets	3322_0-28STH 32	32	2	Klab	64
Battery	hyperion	RB-HYP-15	50	4	Klab	200
Microcontroller	Arduino	Mega 2560	50	1	Klab	50
Encoder	Phidgets	HKT22	25	2	Klab	50
					Total:	5513

Table 2 - Mechanical BOM provided by Kitani's lab budget

Component	Manufacturer	Part #	Unit Price	Quant.	Budget	Total Cost
4 Hole Corner Bracket	80-20	4115	4.05	20	Klab	81
3" Corner Plate 5 Hole	80-20	4151	6.3	8	Klab	50.4
T-nut 1/4-20	80-20	3382	0.21	100	Klab	21
2 Hole Corner Bracket	80-20	4119	2.9	25	Klab	72.5
7x0.375in Timing Belt	Mcmastercarr	70XL037	5	1	Klab	5
8x0.375in timing belt	Mcmastercarr	80xXL037	5	1	Klab	5
2-7/8" wheel	Banebots	T81 Wheel	3.5	2	Klab	7
1/2" Hex hub for 0.25" shafts	Banebots	T81H-RS41	4.5	2	Klab	9
swivel casters	Mcmastercarr	24215T48	1.5	4	Klab	6
Mounting Bracket	Phidgets	3337-0	3	2	Klab	6
					Total:	256

Table 3 - Parts ordered with the MRSD Budget

Part Number	Number Ordered	Description	Subtotal
3531_0	2	Phidget Stepper Encoder HKT22	$50.00
RB-Hyp-15	2	LiPo Batteries	$99.90
MPL3115A2	1	SparkFun Altitude/Pressure Sensor Breakout	$14.95
BNO055	1	Adafruit 9-DOF Absolute Orientation IMU Fusion Breakout	$34.95

460	1	Mini 2-wire Volt Meter (3.2 - 30 VDC)	$7.95
A22E-M-02	1	SWITCH PUSH DPST-NC 10A 110V	$44.64
MF-RG500-2	4	Resettable Fuses - PPTC 5A 16V 0.015ohm Hold 5 Trip 8.5	$2.36
1728	1	LHI XT90 Battery Connector Set for RC Lipo Battery Motor 6 Pairs Yellow ,6 Male Connectors + 6 Female Connectors	$7.99
LYSB00TG1TSUC-ELECTRNCS	1	BNTECHGO 12 Gauge Silicone Wire Ultra Flexible 10 Feet high temp 200 deg C 600V 12 AWG Silicone Wire 680 Strands of Tinned Copper Wire Stranded Wire Model Battery Cable Black and Red Each Color 5 ft	$8.48
2824	2	50:1 Metal Gearmotor 37Dx70L mm with 64 CPR Encoder	$79.90
BNO055	2	Adafruit 9-DOF Absolute Orientation IMU Fusion Breakout	$69.90
B0719H46PF	1	COLCASE Fireproof Explosionproof Lipo Safe Bag for Lipo Battery Storage and Charging	$12.99
9067000172-0	1	Turnigy 5000mAh 4S 40C Lipo Pack with XT90	$43.68
724	1	Terminal Block - 2-pin 3.5mm - pack of 5!	$2.95
1982	1	Adafruit 12-Key Capacitive Touch Sensor Breakout	$7.95
L298N	2	COM-09479	$5.90
60F3W-YT-6SE-6SE	1	Chanzon 60 pcs(6 colors x 10 pcs) 3mm LED Diode Assorted	$6.44
B0756KLNLX	1	Shield Stacking Header set for Arduino MEGA 2560(Pack of 2 Sets)	$8.49
EG5137-ND	2	E-Switch 500SSP1S2M2QEA	$5.44
1N4007DICT-ND	24	1N4007 Diode	$3.84
RG00196	1	Copper tape	$9.99
UHE1H220MDD	10	Nichicon UHE1H220MDD	$3.10
1982	2	Adafruit 12-Key Capacitive Touch Sensor Breakout	$15.90
LVR125S-240	6	Littelfuse Resettable Fuse PPTC LVR 240V 1.25A	$8.22
1725753	2	Phoenix Contact 12P 2.54mm 90DEG	$16.54
B018TW0KN2	2	Electronics-Salon Low Voltage Disconnect Module LVD, 12V 10A, Protect/Prolong Battery Life.	$25.58
B0756KLNLX	2	Shield Stacking Header set for Arduino MEGA 2560(Pack of 2 Sets)	$16.98
9067000172-0	1	Turnigy 5000mAh 4S 40C Lipo Pack with XT90	$43.68

EZ476	1	Dremel EZ476 1 1/2-Inch EZ Lock Rotary Tool Cut-Off Wheels For Plastic - 5 pack	$6.18
B0080X79HG	2	MG Chemicals 835-P Rosin Flux Pen , 1 Pack	$16.60
5952	1	3M Scotch 5952 VHB Tape: 1 in. x 15 ft. (Black)	$17.00
B00CT5CHIC	1	Sabertooth 12A Motor Driver	$79.99
B01ETROGP4	1	Finware 10 Pair XT60 XT-60	$8.45
B01LQ56Z24	1	LHI XT90 Battery Connector	$9.99
B00BWKXTUU	1	DC/DC 24V - 12V Step Down	$18.99
3568	6	Fuse Holder	$5.88
0297003.WXNV	12	3A Automotive Fuse	$3.00
RUSBF110	10	1.1A PTC Fuse	$5.00
1862042	15	2 wire Terminal Block - 45 degrees screwless	$12.45
YO0201500000G	15	2 wire Screw Terminal Block	$10.35
DRV2605L	2	Adafruit DRV2605L Haptic Motor Controller	$15.90
1201	6	Vibrating Mini Motor Disc	$11.70
2824	2	50:1 Metal Gearmotor 37Dx70L mm with 64 CPR Encoder	$79.90
RUSBF160	10	1.6A PTC Fuse	$5.30
0297005.WXNV	15	5A Automotive Fuse	$3.75
		Total	**$969.12**