Khanh Kim Vu

IW10: Robot Construction

Professor David August

Coffee Delivery Robot:
Collision Avoidance, Motion Fine-Tuning, and Fine-Grained Localization

I. Introduction:

Coffee is essential in the workplace, and the Computer Science Department of Princeton University is no exception. Coffee and tea are provided in the Tea Room on the second floor of the CS Building, but walking from your office to the lounge and brewing the coffee may take some time. In order to maximize productivity in the Computer Science Department, our team is proposing to build a coffee delivery robot whose domain of operation will be the second and third floors of this building, where all the CS Department offices are located.

Additionally, this robot could be a valuable source of entertainment. It would enliven the halls and stimulate productivity of staff and faculty. Following examples of similar projects on other campuses, such as University of California's Kiwibot, and George Mason University's Starship, both food delivery robots, our project will take autonomous mobile robots another step closer to college communities, making them more accessible and popular.

Our robot will take orders through an online form and deliver coffee to user-specified rooms. As an office service robot, in order to be effective, it must maintain safety for not only itself, but also the humans and their properties. This includes not running into people and causing injuries, not colliding with objects and ruining properties, and, of course, not destroying itself, as this is not in the interest of its human creators. In order to do this, the robot must be able to react

quickly to changes in the environment, ranging from moving humans to static obstacles. The robot must be able to perceive the environment with a continuous feedback loop instead of completely relying on static environment mapping, react to unexpected obstacles and difficulties in navigation, and plan or re-plan the path flexibly in order to still achieve its goal without constant human supervision and guidance [2].

II. Related Work:

Collision avoidance in automated robotics have been a well-researched topic, as it is one of the most important steps for robots to reach full autonomy. According to Gageik et al. 2015 [5], up to now, there are two main categories for collision avoidance solutions. The first category executes a path based solely on environmental feedback it. This means the robot may not be aware of an overarching goal and may retrace its steps, undoing its own work to get there. The second category saves a map of the environment, and clearly tracks and navigates the robot from origin to destination using pre-calculated instructions that may take a lot of time and memory overhead, but is ensured to reach the goal. These two categories can also be respectively referred to as local planning, which has high resolution and quick response, and global planning, which is low resolution and slow response [4]. Global path planning alone is often called *static* path planning.

Fox et al 1997 [2] proposed a hybrid approach, using both sensor feedback (local) and map of the environment (global), called *dynamic window approach* (DWA), which is optimal because it relies on environment mapping to achieve accuracy, but still reacts flexibly to unexpected obstacles [4]. Our robot hopes to test this dynamic approach using modern

localization tags, for global planning, and distance sensors, for local planning, to increase efficiency.

In the specific field of anti-collision using sensors, there are two main types of sensors: sound-based and light-based [5]. Light-based sensors often fail in poor lighting conditions, or in detecting glass materials, whereas sound-based sensors cannot accurately detect sound-absorbing materials such as cloth [5]. Adarsh et al 2016 [8] performed an experiment to compare the effectiveness of infrared sensors and ultrasonic ones. They have found that ultrasonic is better for sponge, wood, plastic, and tile materials, whereas infrared is better for paper, which transmit sounds. A combination of both sensor types is optimal for cardboard and rubber. We chose infrared sensors because the walls in the CS Building are made of paper-like materials.

In terms of the design for its working process, according to [12], there are three phases that comprise the motion of an autonomous mobile robot: localization, path planning, and path execution. During the first step of localization, sensor readings are used to deduce the robot's position and orientation in the environment. The second step, path planning, is to use an environment map to determine an optimal path between the robot's current location and its destination. The most well-known path planning tool is the global position system (GPS). However, GPS is expensive [13]. Its accuracy is also limited, so when fine positional control is needed, it can prove ineffective. GPS is further limited by the fact that it will only work in outdoor environments [12]. Therefore, in autonomous robots, path planning is instead often done by a network of sensors, whose positions in the environment are pre-defined. In order to be accurate, sensor networks often consist of a large number of sensors, which can also become expensive. The third step is to execute that path, when error is most likely to occur. Poor estimates in position during path execution require more frequent localizations to be made,

incurring extra overhead and possibly slowing the movement of the robot. It is therefore important to use a more accurate method to minimize positional errors during the path execution phase. Lee & Song 2004 [12] have suggested a method using the extended Kalman filter to estimate the robot's position, instead of making various localization measurements. Our approach builds upon theirs by using both localization tags and distance sensors to locate the robot, cross-checking to reduce errors without having to use Kalman filter.

Within these three main phases of autonomous robot mobility, distance sensors have been used in many other functions besides anti-collision. Ibisch et al. 2013 [11] proposed using a sensor network of LIDARs embedded within the infrastructure environment, which was a parking garage in their experiment, to estimate and track vehicles' position and orientation, in the garage's coordinates. They installed stationary sensors within the environment because it was a public garage, and they couldn't install sensors on every driver's car. Instead, in order to track a moving car from one sensor to another, they first separated sensor data points into *static* and *active* points. Active points, i.e. of moving objects, are used in a series of probability calculations to infer the most likely position of the car within the garage. They had to take care with their calculations to avoid multiple detections of the same object and handle occlusions and intersections of moving objects across different sensors. Their proposed solution to indoor tracking and navigation is insightful and relevant to our experiment of delivery within an office environment, but it can be further optimized. In their context, the environment is static, but the vehicles are not, while in our experiment, both are static and defined. Therefore, we can install tracking devices on both in order to avoid uncertainty and inefficiencies in vehicle detection.

III. Approach:

In my experiment, in order to maximize the efficiency of each of the robot's components, the distance sensors will be used for (1) anti-collision, as well as (2) robot's path alignment and (3) office door detection.

A. Collision Avoidance:

For the collision avoidance module, I wrote an algorithm that constantly maintains a "private sphere" around the robot of a predefined radius. The sensors will be used to report distances to objects surrounding the robot, and if the data points are lower than the radius threshold, the robot should halt. When the robot begins to move initially, the signal is set to "stop". Only until the sensors have confirmed no obstacle within this "sphere" would this signal be changed to "go". As the robot moves, similarly, if any obstacle is detected within range, the signal will change to "stop" again.

B. Angle Correction:

The robot was built from scratch with different kinds of motors for the two front wheels, so it is challenging for it to keep a perfectly straight line. Since the sensor readings should be used to keep the robot to the right of the hallway anyway, we can also use them to ensure the robot moves straight, by aligning it with the wall. In this context, the sensors function as an additional source of information verification for the robot, which is especially necessary if its other angle measurements are affected by noise in order to fine-tune the robot's motions, ensuring it moves in a smooth straight line.

C. Fine-grained Localization:

The robot's navigation system relies upon a network of wireless location tags.[1] However, processing this data only gives an approximation of the robot's location in relation to everything else on the map. However, in order for our coffee delivery mission to be successful, the robot should deliver to the user's office door. We currently use the side sensors to align with the walls, but they can also distinguish between flat wall and door without difficulty. Thus, sensors can be used to guide the robot to right outside the office, given that the tags can lead the robot to within a certain distance of the door.

IV. Implementation:

Within our team of five, each of us worked on a different component of the robot to make it not only mobile but also autonomous. The work is divided into five parts: (1) mobility and stabilization by Collins Metto, (2) global localization and navigation directions by Yashodhar Govil, (3) human interaction by Janet Lee, (4) elevator navigation using computer vision by Sandun Bambarandage, and (5) collision avoidance, localization and motion fine-tuning. I worked on the last part, in close association with Collins Metto's part (1), and Yashodhar Govil's part (2), which uses a network of sensor tags.

I used VL53L1X Time-of-Flight distance sensors, which detect objects by emitting an infrared laser and timing the reflection from the target. They have a range of 40mm to 4m, with +/-5mm accuracy. The field of view is fairly narrow, only 15 to 27 degrees, but with a reading rate of up to 50Hz.[2] In order to use multipled of these sensors on the same Raspberry Pi, I connected them through the TCA9548A 8-channel mutiplexor.[3] At an overall glance not

[1] https://www.decawave.com/product/dwm1001-module/
[2] https://www.sparkfun.com/products/14722
[3] https://www.sparkfun.com/products/14685

accounting for corner cases, the robot will generally need sensors in the front for collision avoidance, and on the sides for calculating angles and detecting doors. As the robot moves, if it keeps a straight line as desired, then sparse sensor configurations at the front may not pick up certain small, static objects, such as chair legs. On the other hand, all sensors on the sides will eventually pass the door or the particular point on the wall we need them to detect, so their placement does not matter as much. Therefore, I deduced that more sensors are needed on the front than the sides. I had four front sensors, evenly spaced out at distance of 115mm from one another, and two on each side, 277mm apart. I labeled them 0 to 7, from farthest back right to farthest back left, like an arc. The sensors are mounted on the lower shelf of the robot, about 250mm off the ground.

The domain of operation is the second and third floor the CS building, where most of the offices are located (see Figure 1). The robot is currently restricted to only the main hallway, between the elevator and rooms 201 or 302, for second and third floor, respectively. This hallway is 1700mm wide, while the robot is 510mm wide and 610mm long.

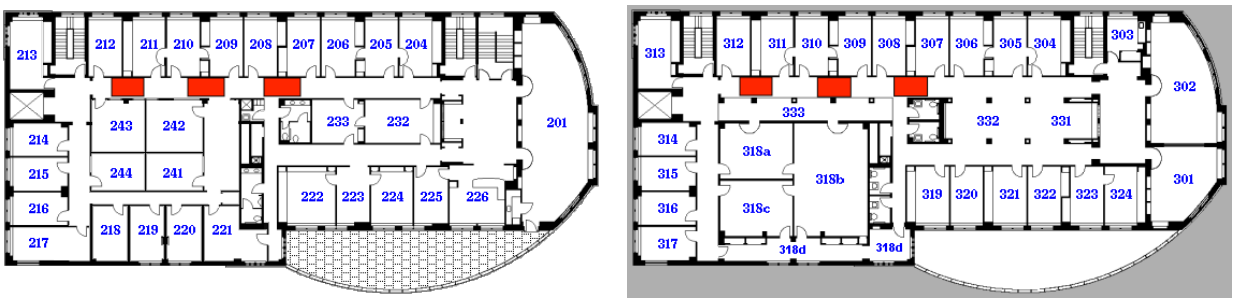

Figure 1. The floor plans for the CS Building's second (left) and third (right) floors. The robot is restricted to the main hallway between room 201 and 213 on the second floor, and room 302 and 313 on the third floor. The elevator is located right next to rooms 213 and 313. The areas marked red are proposed to give good angle measurements.

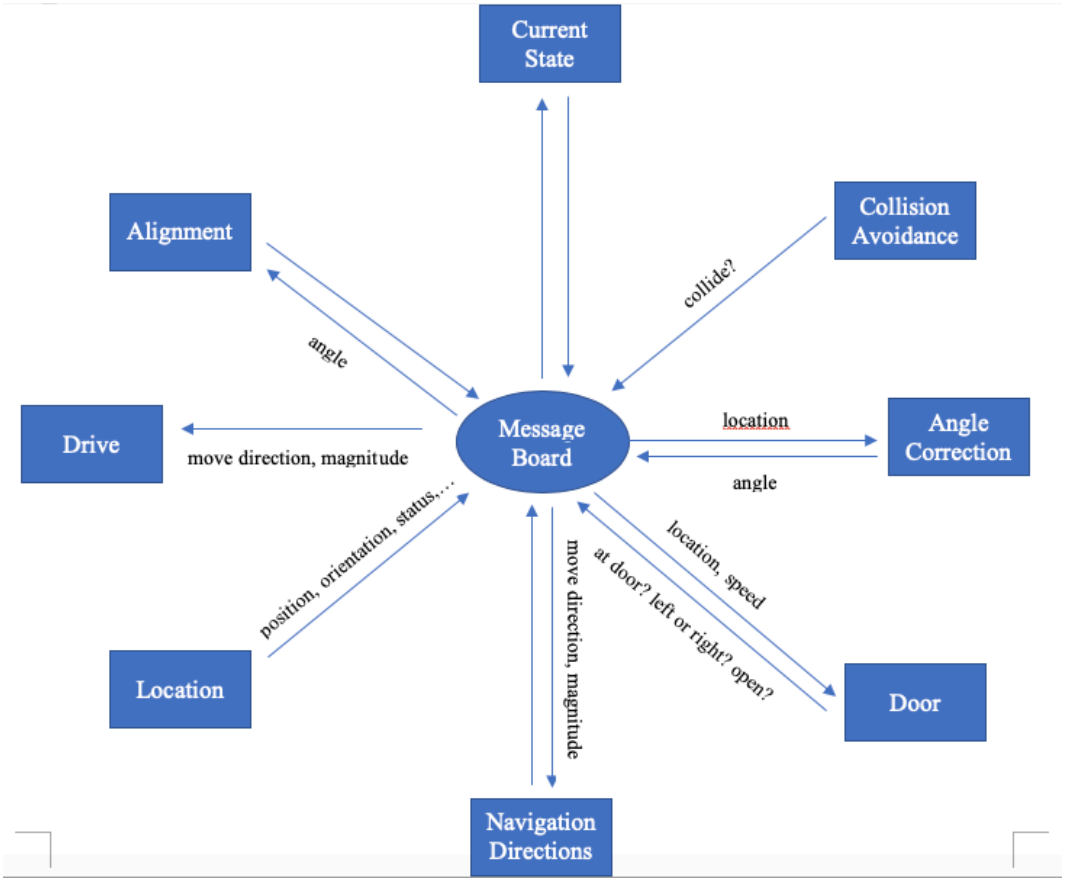

Figure 2. Diagram of how various modules interact with the message board and thus indirectly with one other. Arrows towards the message board indicate what each module post as outputs for other modules to see. Arrows away from the message board indicate what each modules grab as inputs from other modules' outputs.

These five parts of the robot need a common communication means. My part has 3 subparts, called modules, that all use sensors as inputs but serve different purposes. The first module is "collision avoidance", which only takes sensors as inputs and sends a boolean of whether there will be collision to the message board as output. The second module is "angle correction", which takes from the board the robot's current location in order to check whether it's a good place for angle measurement, then outputs the new angle, if any. The third module is "door location", which also takes the location as input, and outputs whether the robot is at the office entrance and other information about the door. Each of the other parts is comprised of at

least one module that, similarly, will maintain a connection with the message board to communicate with other modules (refer to Figure 1 for more details). However, the message board is only a means of communication. It does not process or compile the pieces of information from all the modules in any way. Thus, we created the "current state" module, which reads messages primarily from "location", "angle correction", and "navigation" modules to provide the following most crucial information about the robot's current functioning state: position, orientation, floor, angle from straight line, move instructions, and delivery status. The "current state" module ensures that these information are the most updated based on all the other modules. It provides other modules with convenient access to the most recent information on the robot.

A. Collision Avoidance:

For collision avoidance, setting the radius was crucial to the robot's operation. Initially, the radius was set to 100mm for all the sensors, but as we needed to stick to the right of the hallway due to courtesy, this sensed the right wall as an obstacle and never allowed the robot to move from its initial stop position. However, if we lower the radius to 50mm, the robot couldn't decelerate in time before running into the obstacle. A solution to this problem would be to define separate radii for the front and the sides. For all the three faces (front, right, and left), we would all need to maintain a minimum distance from the nearest object, but this distance can vary. For the sides, we know that we might want to stay close to the walls under certain situations, so we need to maintain a smaller minimum distance. Whereas for the front, we need a larger minimum to buffer for the deceleration. Through trial and error, we found that the most appropriate radii were 200mm for the front and 50mm for the sides. If these protective radii are violated, we will

post a notification to warn the motors to halt. Once stopped, if the radii are cleared up again (i.e. object is moved), then we send another notification.

B. Angle Correction:

We want to calculate the angle, which is how many degrees the robot has veered off the straight line in the hallway, by the difference in distance to the wall between the two sensors on each side. However, we also need to take into account whether the robot is close to anything that is not flat wall, such as a door, closed or open, which can mess up the angle calculations. For example, the robot is close to an open door and is veering towards the left wall at a 45-degree angle. If the front right sensor catches the door but the back right one does not, then our angle would be much larger than 45. We can bypass this problem if we know exactly which locations will yield correct angle measurements, and allow the robot to get the angle from the sensors only in these defined locations. Luckily, the operation domain of this robot's preliminary version is restricted to the second and third floors of the CS Building, which have clear floor plans. We marked out coordinates for appropriate areas (see Figure 1), and my algorithm is a loop that watches the robot's location. Whenever the robot enters these areas, the algorithm would immediately read the sensors and update the angle by sending a notification so that the other modules, especially one for motion, can apply a correction angle, if necessary. I chose these specific areas in order to avoid doors.

To ensure angle accuracy, the sensors on the side closer to the wall should be used. The hallway is 1700mm across, and the robot is 510mm wide. Assuming the robot is in one of the areas defined above, the sensors on the closer side should both read less than half of 1700mm. So we would only read from these sensors for reliable data. My algorithm would read from any side

on which both sensors are at most 850mm (half of the hallway's width) from the wall. If the sensors don't meet these conditions, we will not update the angle. My algorithm can easily bypass open hallways, which would record more than 1000mm away, and thus we need not avoid areas with open hallways. As a safety measure, I require that *both* sensors read less than 850 to prevent calculating angle from noise. If none of the side sensors are less than 850, then do not update angle. However, closed doors could still give less than 850mm in distance, but would yield inaccurate angles as they are farther into the wall. As an additional precaution against inaccurate calculations, the algorithm would not run if there is any unexpected obstacle on the path.

C. *Fine-grained Localization:*

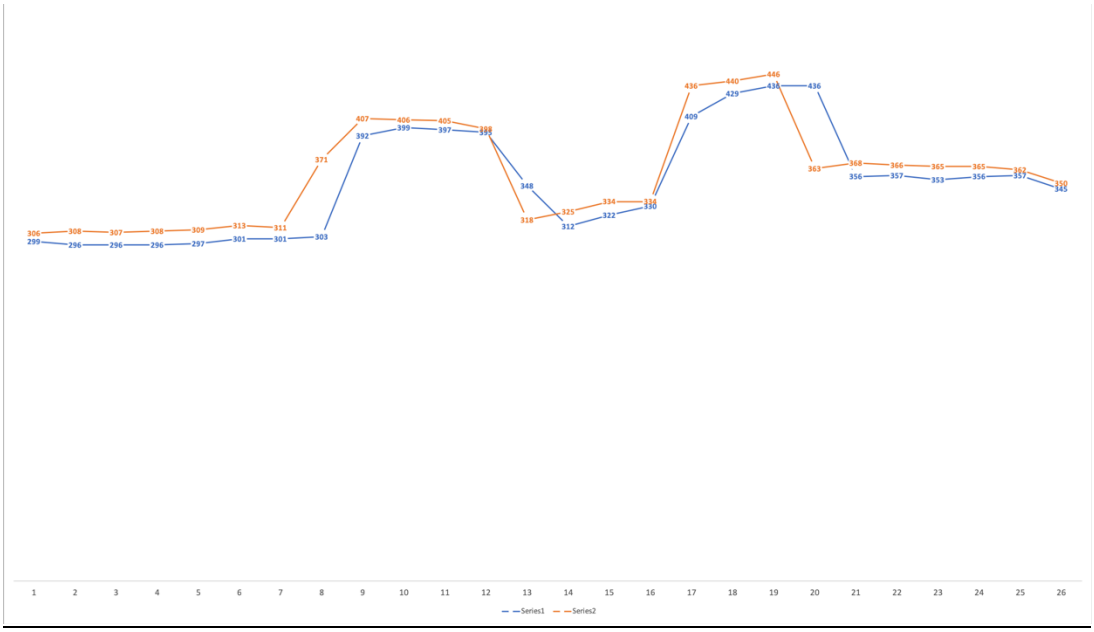

Figure 3. Plot graph of sensor data points when the robot was passing two doors in a row. Y-axis represents distance from sensor, in mm, X-axis represents time, with the time interval being 0.25 seconds. Series 1 represents sensor 0, the farthest back right. Series 2 is sensor 1, the front right.

As the sensor moves past the door, it should read a slightly larger number, as the door is deeper in and farther from the sensor than the wall. The CS Building office doors don't have

large door frames that are noticeable in the readings, so the data resembles going straight from wall to door (see Figure 2). In order to account for noise, which may cause one to two data points to vary but should not signify a change between wall and door, I have decided to keep a sliding window average. The window size should be twice the number of data points the sensors should read of the door, which, in the example above, is 4 per door. It needs to be twice because we must compare the average of the four points at the door and the four points before the door, and if there is an increase of more than 80mm, then we are at the door or almost passing it and will send a signal to halt the robot. The door is about 100mm in from the wall, so allowing for a few lower data points that were recorded at the door frame, I figured out that 80mm was the optimal threshold, through trial and error. The window size is customizable, depending on the speed of the robot is travelling at when it passes the door.

V. Evaluation:

A. Collision Avoidance:

With the radii defined above, I ran a series of experiments, the most notable of which is obstacle detection with various different objects. Within the CS Building, the main objects that are most likely to appear unexpected in the hallway include empty cardboard boxes and trolley carts left against the wall after mail delivery, chairs that professors leave outside their doors during office hours. There are three types of chairs used in the building: thick-framed sofas, one-legged spinning chairs, and four-legged wooden chairs. I collected sensory data by moving the robot towards the obstacle so that the obstacle completely or at least mostly obstructs the path. Figure 4 pictures the front four sensors' readings while moving towards a mail cart with two shelves, thick plastic bars in the four corners connecting the lower and upper shelves. The lower

shelf was about 20mm below the sensors. The center sensor (Series 3 in Figure 4) started high then decreased as the robot approached the cart, as expected. However, there was the sudden rise at data point 21, which can be explained by the sensor getting too close to the cart so that it couldn't detect the lower shelf anymore. The right sensor (Series 1) completely missed the cart, and is shown with a lot of noise, as the other end of the hallway is much farther than the sensors' range. The left sensor (Series 4) detected the cart's left bar at data point 11, while before that, it was producing noise as it was trying to read the other end of the hallway.

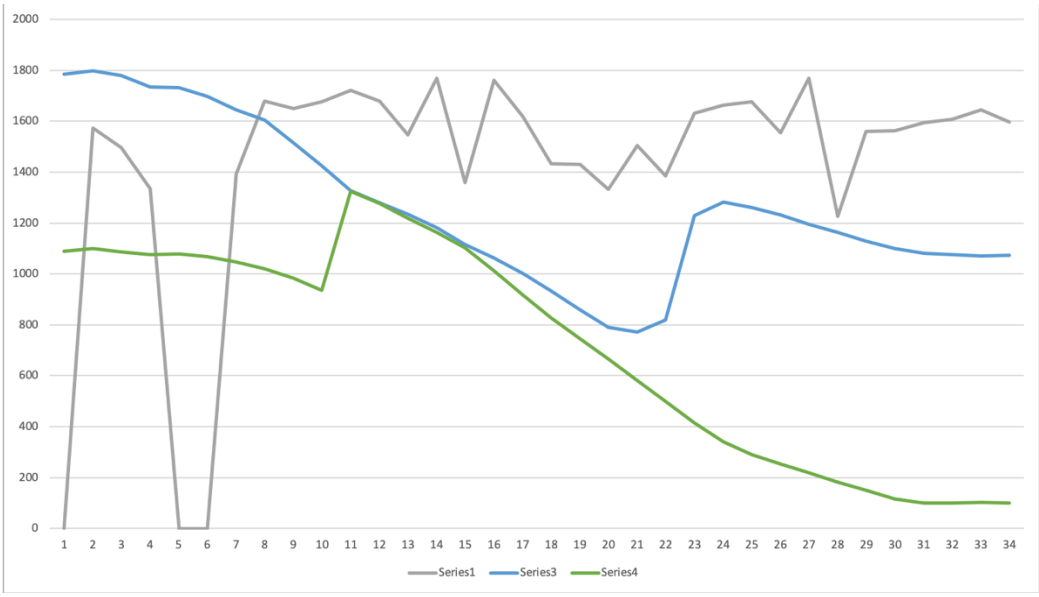

Figure 4. Plot graph of sensor data points when the robot was approaching the mail cart with a low base. Y-axis represents distance from sensor, in mm, X-axis represents time, with the time interval being 0.1 seconds. Series 1, 3, and 4 are the front sensors from right to left, respectively. The cart was against the right wall, and the robot was traversing down the hallway aligning itself close to the right wall.

Figure 5 depicts four sensors with non-noisy data, as the robot was moving towards a chair with one pole at the sensor's eye-level. Series 1 shows a stable distance from the right wall. Series 3 was noisy at first, until it detected a piece of fabric on the underside of the seat (part of the design) at which point it steadily decreased. Series 5 was close to the center of the robot, and thus was able to detect the pole, but unsteadily because it was on the edge and alternated between detecting the pole and past it.

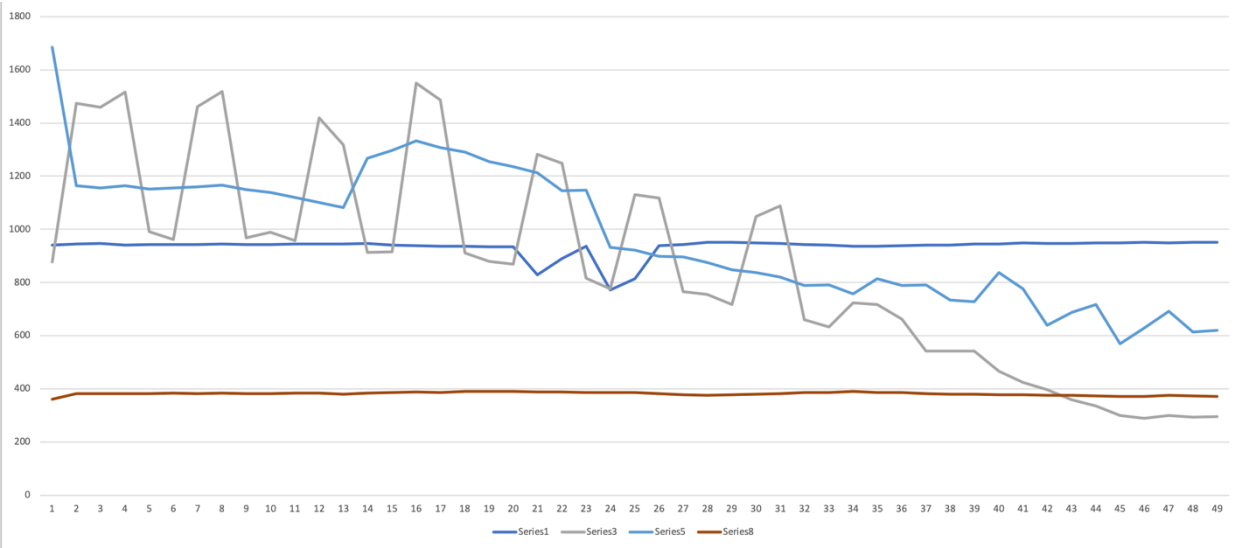

Figure 5. Plot graph of sensor data points when the robot was approaching a spinning chair whose five-wheel rollable base was connected to the seat by a single large pole. Y-axis represents distance from sensor, in mm, X-axis represents time, with the time interval being 0.1 seconds. Series 1 is a sensor on the right side of the robot. Series 8 is a sensor on the left side. Series 3 and 5 are sensors at the front.

Figure 6 depicts the data points for an obstacle of a large sofa. Series 8 showed that the robot was sticking close (only 200mm) to the left wall. The front sensors were all a bit noisy at first, when the obstacle was more than 1000mm away, but steadily decreased from data point 27. The sensors had no issues with detecting such a large object. However, not depicted are the sensor readings for a wooden chair with four thin-framed legs. The sensors are only able to detect them in certain circumstances when the robot approaches the chair at the right angle.

From these results, we can see that the sensors' data are accurate and understandable. However, these sensors' range is quite restricted, so they fail to detect obstacles appropriately when the obstacles are not directly in front. With the current configuration of sensors on the robot, even with four at the front, twice as concentrated as than on the sides, it is still possible for certain thin objects or certain angles of approach to make the sensors fail at detection.

Additionally, noise tends to occur above 1000mm in distance. This is explainable because although the maximum range of these sensors is 4m, I currently use the mid-range mode (out of 3 modes available within the sensors), so the maximum range is lower. However, ignoring noisy data points above 1000mm should not affect the effectiveness of my algorithms. The first algorithm is only concerned with close-ranged readings to avoid collision. The second algorithm deals with the closest of the two walls in the hallway to best align the robot, so its data points should not be more than 850mm. The third algorithm to detect doors is not affected either. For

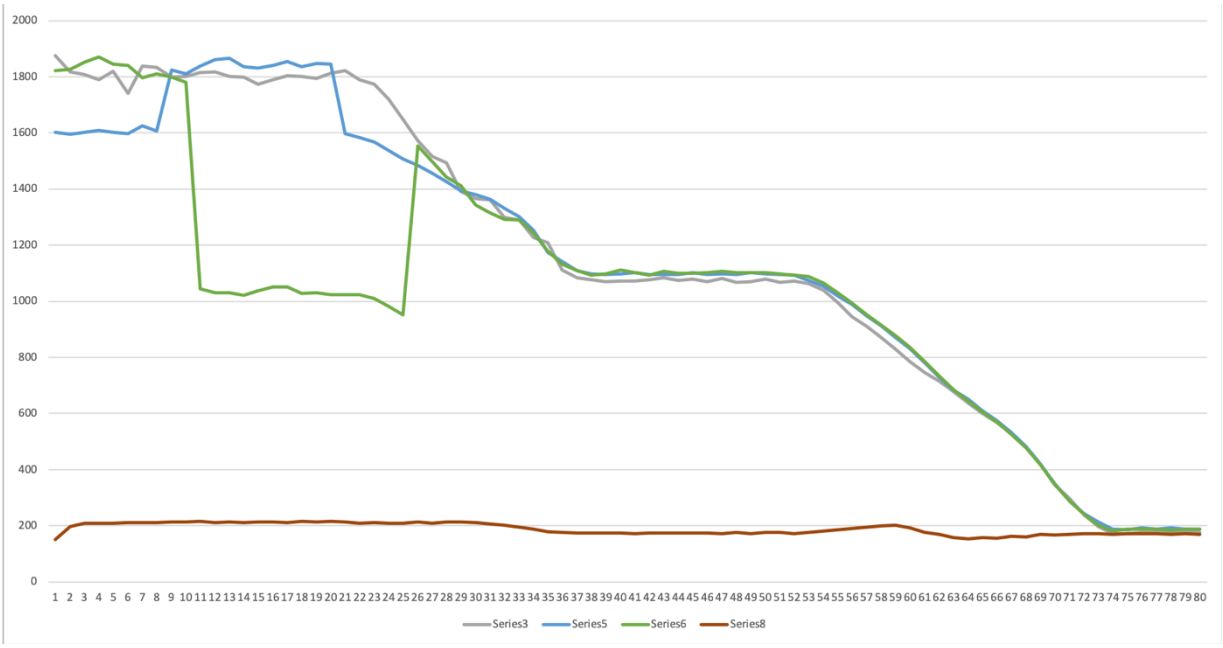

Figure 6. Plot graph of sensor data points when the robot was moving towards a large sofa with solid armrests, resembling a rectangular box from the side. Y-axis represents distance from sensor, in mm, X-axis represents time, with the time interval being 0.1 seconds. Series 3, 5, and 6 are front sensors, while Series 8 is on the left side. The sofa was leaning against the left wall of the hallway, and the robot was aligning itself close to the left wall as well.

closed doors, the small increase should not make the distance data (also close-ranged) noisy. For open doors, we can easily distinguish it from closed doors if we see noise.

B. Angle Correction:

Angles were calculated for various orientations and positions the robot could assume in the hallway (see Figure 7). In the following examples, the robot was situated near or within the middle red area in Figure 1, outside of room 209 and 210. In Situation 1, the robot was between two segments of uninterrupted walls. Since it was closer to the right wall, the algorithm calculated the angle based on the right side sensors. The actual angle, measured manually, was 35 degrees, and the calculated angle was 35.4 degrees. In Situation 2, the robot was manually positioned parallelly to the walls, in the same hallway segment as Situation 1, and the calculated angle was 0.03 degrees.

In Situation 3, the robot was close to an open hallway, with the right front sensor pointing into the hallway but the right back sensor pointing at the wall. The angle measured manually was 12 degrees. In this instance, since the robot was closer to the left wall, the angle calculated based on the left sensors was 13.2 degrees. If the angle had been based on the right sensors, it would have been 64 degrees, as the right front sensor's distance reading was too large.

In Situation 4, a closed door was on the robot's right side. The right front sensor reflected off of this door, but the right back sensor reflected off of the wall. The robot was closer to the right wall, so the angle was calculated based on the right sensors. The manually measured angle was 19 degrees, while the calculated angle was 36 degrees.

The areas marked red in Figure 1 gave angle calculations of +/-1 degree accuracy, even with open hallways because the algorithm ignores data from open hallways. Closed doors, which

are only a few inches in from the wall, give drastically different angles, so were rightfully avoided. Additionally, from Situation 2, we can see that possible inconsistencies in walls or sensor readings could result in a small margin of deviation that we should take into account. Therefore, all angles below 5 degrees are considered trivial and need not be corrected.

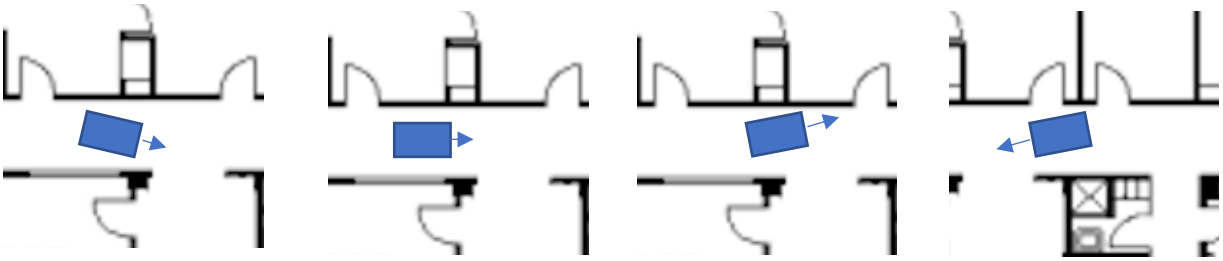

Figure 7. Depictions of the robot in relation to the hallway under four different circumstances. The arrow signifies the direction the robot is facing. Situation 1 to 3 (left 3 pictures): the robot entering area appropriate for angle measurement outside room 210 (refer to Figure 1). Situation 4 (rightmost picture): the robot having just exited that area and is now in front of a closed door.

C. Fine-grained Localization:

Our algorithm for door detection, as described above, maintains a moving window average between the first half and second half of data points within the window. The window size must change according to the robot's current speed. As the robot moves fast, it passes the door quickly and thus have sharp increases in distance of one to two data points (see Figure 8). As the robot moves slowly, the increases are more gradual, with no clearly defined peak points. Therefore, the greater the speed, the smaller the window size.

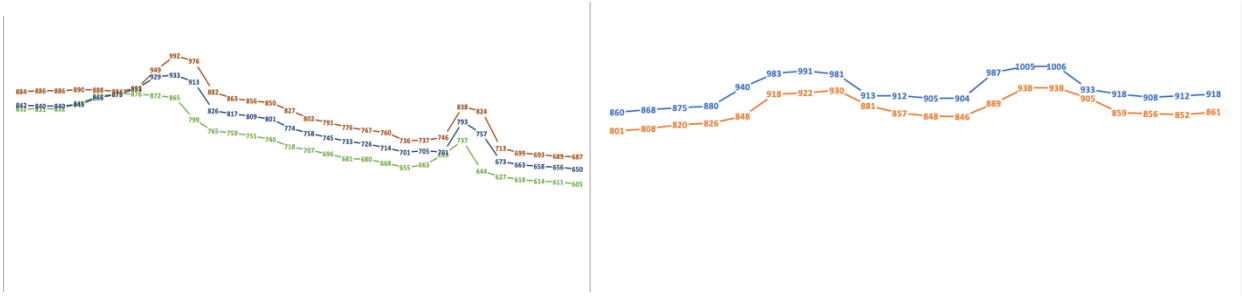

Figure 8. Plot graph of sensor data points when the robot was moving past closed doors. Y-axis represents distance from sensor, in mm, X-axis represents time, with the time interval being 0.1 seconds. The leftmost graph depicts sensory data points at high speed, and the rightmost graph depicts data points at low speed.

VI. Conclusion:

In this paper, I present a cheap and simple solution, using easily accessible distance sensors, sold commercially, to solve an indoor service robot's navigation three main problems. The first problem is anti-collision. The solution is to maintain a sphere of adjustable radii around the robot. If the sensors give readings lower than this radii threshold, then an obstacle is becoming close and the robot should stop. The second problem is about fine-tuning the robot's motion, ensuring it moves in a straight line by aligning it with the closest wall. If an angle greater than 0 is detected, the robot should adjust accordingly to get back on track. The final problem is fine-grained localization, relying on distance sensors to pinpoint the exact location instead of the wireless tags' approximation. The sensors' data demonstrates difference between wall and door clearly.

VII. References:

[1] C. Stachniss and W. Burgard, "An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, 2002, pp. 508-513 vol.1.[4]

[2] D. Fox, W. Burgard, and S. Thrun, "A Hybrid Collision Avoidance Method for Mobile Robots", *IEEE Robotics & Automation Magazine*, vol. 4, no. l, 1997.[5]

[3] D. Fox, W. Burgard, S. Thrun, and A. B. Cremers, "A hybrid collision avoidance method for mobile robots," *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, Leuven, Belgium, 1998, pp. 1238-1243 vol.2.[6]

[4] J. S. Zelek and M. D. Levine, "Local-global concurrent path planning and execution," in *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 30, no. 6, pp. 865-870, Nov. 2000.[7]

[5] N. Gageik, P. Benz, S. Montenegro, "Obstacle Detection and Collision Avoidance for a UAV with Complementary Low-Cost Sensors", *Access IEEE*, vol. 3, pp. 599-609, 2015.[8]

[6] S. Morikawa, T. Senoo, A. Namiki and M. Ishikawa, "Realtime collision avoidance using a robot manipulator with light-weight small high-speed vision systems," *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Roma, 2007, pp. 794-799.

[7] S. Khan and M. K. Ahmmed, "Where am I? Autonomous navigation system of a mobile robot in an unknown environment," *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*, Dhaka, 2016, pp. 56-61.[9]

[8] S. Adarsh, S. Mohammed Kaleemuddin, Dinesh Bose, K. I. Ramachandran, "Performance comparison of Infrared and Ultrasonic sensors for obstacles of different materials in vehicle/ robot navigation applications", *International Conference on Advances in Materials and Manufacturing Applications, 2016,* Bangalore, India, 2016, vol. 149.[10]

[9] T. Mohammad, "Using Ultrasonic and Infrared Sensors for Distance Measurement", *World Academy of Science, Engineering and Technology*, vol.3, 2009.[11]

[10] T. Suzuki and S. Shin, "Goal-directed navigation strategy for a mobile robot under uncertain world knowledge," *IEEE International Conference Mechatronics and Automation, 2005*, Niagara Falls, Ont., 2005, pp. 741-746 Vol. 2.[12]

---

[4] https://ieeexplore.ieee.org/document/1041441
[5] https://www.ri.cmu.edu/pub_files/pub1/fox_dieter_1997_1/fox_dieter_1997_1.pdf
[6] https://ieeexplore.ieee.org/document/677270
[7] https://ieeexplore.ieee.org/document/895924
[8] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7105819&tag=1
[9] https://ieeexplore.ieee.org/document/7760188
[10] https://iopscience.iop.org/article/10.1088/1757-899X/149/1/012141/meta
[11] https://www.researchgate.net/profile/Tarek_Mohammad2/publication/258883212_Using_Ultrasonic_and_Infrared_Sensors_for_Distance_Measurement/links/00b4953bd945ba5d21000000/Using-Ultrasonic-and-Infrared-Sensors-for-Distance-Measurement.pdf
[12] https://ieeexplore.ieee.org/document/1626642

[11] A. Ibisch *et al*., "Towards autonomous driving in a parking garage: Vehicle localization and tracking using environment-embedded LIDAR sensors," *2013 IEEE Intelligent Vehicles Symposium (IV)*, Gold Coast, QLD, 2013, pp. 829-834.

[12] Sooyong Lee and Jae-Bok Song, "Robust mobile robot localization using optical flow sensors and encoders," *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, New Orleans, LA, USA, 2004, pp. 1039-1044 Vol.1.

[13] P. Biswas, T. -. Liang, K. -. Toh, Y. Ye and T. -. Wang, "Semidefinite Programming Approaches for Sensor Network Localization With Noisy Distance Measurements," in *IEEE Transactions on Automation Science and Engineering*, vol. 3, no. 4, pp. 360-371, Oct. 2006.

[14] E. Bicho, P. Mallet, and G. Schoner, “Target Representation on an Autonomous Vehicle with Low-Level Sensors”, *International Journal of Robotics Research*, Vol. 19, No. 5, May 2000, pp. 424-447.