EECE 5554: Robotics Sensing & Navigation
Professor Kristen Dorsey

Ultrasonic Mapping
Final Project Report
4/28/23

Gopisainath Mamindlapalli
Vasu Kumar Reddy Sirigiri
William Gao
Jacob Film

Project Application

The inspiration for our project came from the technique of echolocation, a sound-based sensing method used by animals such as bats and whales as a primary means of navigating and locating prey while hunting. Humans with varying levels of vision-impairment have also been able to employ this auditory-based technique to augment their reduced sensory capabilities. Our group wanted to replicate the sensory feedback that echolocation provides, so we set out to create a system that produces spatial information about one's surroundings through the use of an ultrasonic device. While there are obviously several other types of sensors used today (laser-based, depth cameras, etc..) for obtaining spatial information, we wanted to dig into the challenge of relying principally on sound.

We envisioned a system that could give users live haptic feedback corresponding to the spatial proximity of objects that the sensor has detected. This would provide them with a contact-free method (i.e. without the use of a cane) of not only sensing the world around them but confidently navigating through it. While a full implementation of this application lies outside the scope of this relatively brief project, our aim was to develop a spatial mapping foundation that the navigational feedback could then be built upon.

Approach and Sensing Principle

Ultrasonic Transceivers (UT's) emit high-frequency pulses of sound and measure the time between when the pulse is sent out and when a reflection is detected. This 'time of flight' is converted to a distance by a temperature-compensated factor for the speed of sound in air. As discussed in our lectures, Inertial Measurement Units can sense linear acceleration, angular velocity, magnetic field, and orientation in space through the use of MEMS-based spring-mass systems, gyroscopes, and magnetometers.

The sensors we selected were the MaxBotix HRUSB-MaxSonar MB1433 and the VectorNav VN-100. By pairing the distance-sensing capability of our UT with an increasingly complex model of IMU position and orientation tracking, we sought to map out the sensor's surrounding objects (features of a room) relative to a global coordinate frame. Our three 'levels' of goals mark increases in the complexity of our IMU usage. Following a brief explanation of our sensor fusion approach, we discuss each level's mapping model, method of data collection, primarily limitations, and visualization approach, accompanied by the presentation of sample results.

Sensor Fusion

Our UT outputs data at 4Hz, whereas the IMU outputs data at 40Hz. To fuse these readings together for use in both live and post-processing, we developed our own driver which pairs each IMU reading with the most recent UT reading by running two separate threads and sharing the UT range reading as a variable. This then outputs a single message containing all relevant info at a synchronized rate. Due to the discrepancy between data publishing rates, this does result in a potential growth of range approximation error between UT data updates, but our assumption of range continuity is valid for slower movements.

Level One

Our first mapping goal was to simply pair the yaw output values of the IMU with the range output values of the UT. This required the constraints (and consequent mathematical assumptions) of keeping the sensor origin fixed in space and keeping the sensor's x-y plane parallel to the world x-y plane, or put simply, rotating about the z-axis. The positions (x,y,z) of objects with respect to the world reference frame were calculated as shown:

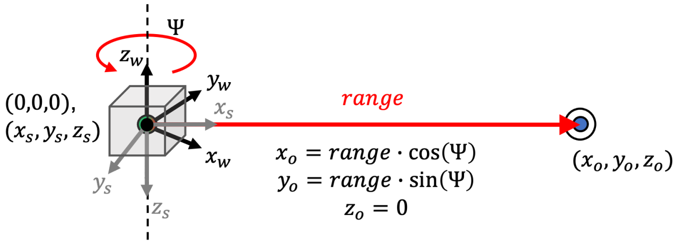

Figure 1 - Level 1 Mapping Model

To collect data with these constraints, we placed and leveled our combined sensor unit on a stable surface in the center of a room. Slowly rotating it through a complete revolution, we obtained sets of data like the plot shown for Hayden Hall Room 419 in Figure 2 below.

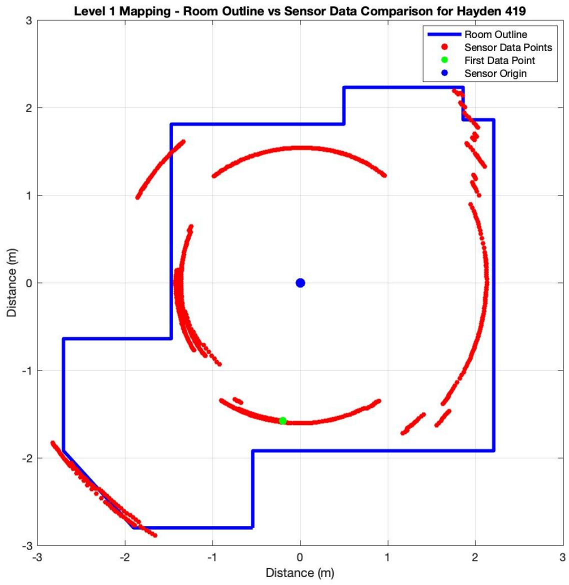

Figure 2 - Example of Level 1 Results

The errors and limitations of our Level One implementation are not hard to see. The primary source of error that our data was subject to for this goal was the non-zero beam angle of our ultrasonic sensor (see Fig. 3 at right).

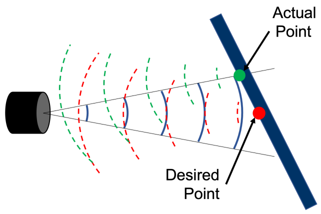

Figure 3 - Beam Angle Error

This causes lower range values to occur when the sensor is facing non perpendicular surfaces as the reflections of sound from adjacent surfaces return to the sensor more quickly. The 3D 'cone' shape of the beam also can pick up nearby obstacles that the sensor is not directly pointing at, (e.g. chairs, cabinets, etc…) leading to some undesired data points.

In addition to plotting this data after collection, we wanted to implement a visualization tool that displayed live results. Through the use of ROS's RViz package, we were able to obtain a real-time virtual display of our sensor data. By utilizing a node subscribed to the combined driver, we are able to collect synchronized IMU and UT data and convert to (x, y, z) global coordinates as outlined above, maintaining our 2D assumptions. This node outputs PointCloud data which is then read into the RViz render panel. While collecting data via our Level 1 methodology, the display progressively populates with points located relative to the sensor origin at (0, 0, 0), culminating in a display such as that shown in Figure 4.

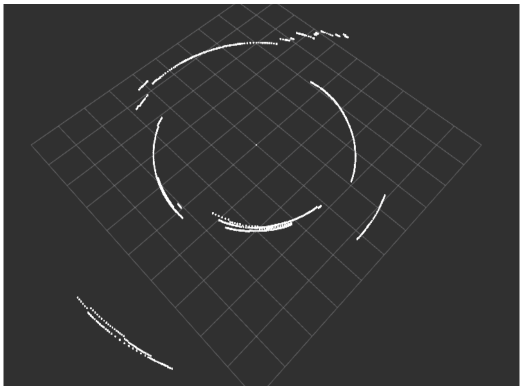

Figure 4 - Level 1 RViz Display

Again, the effect of beam angle error is quite evident in the curvature shown, but the general goal of providing live spatial information and obstacle proximity to the user is nevertheless achieved as there are clear differences in radius that correlate to the walls and other objects present in the space.

Level Two

Our second level goal again constrained the sensor's roll and pitch to 0, keeping the sensor and world x-y planes parallel to one another, but we now allowed for rotation and translation in the plane. For mapping in this manner, we needed to pair absolute yaw with range data while also performing an acceleration-based dead-reckoning tracking of the sensor position.

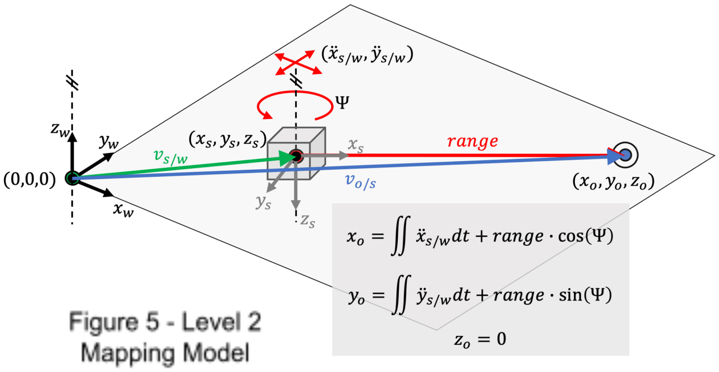
$(\ddot{x}_{s/w}, \ddot{y}_{s/w})$
Ψ
z_w
y_w
(x_s, y_s, z_s)
x_s
$range$
$v_{s/w}$
$(0,0,0)$
$v_{o/s}$
(x_o, y_o, z_o)
y_s
x_w
z_s
$x_o = \iint \ddot{x}_{s/w} dt + range \cdot \cos(\Psi)$
$y_o = \iint \ddot{y}_{s/w} dt + range \cdot \sin(\Psi)$
$z_o = 0$

Figure 5 - Level 2 Mapping Model

To physically collect this data, we placed and leveled our sensor unit on a rolling podium and collected data while moving along a path offset to the walls of a room. In order to reliably track sensor location, the IMU was reconfigured to output gravity-compensated NED-frame acceleration. An example of our results is shown in Figure 6 for Hayden 425:

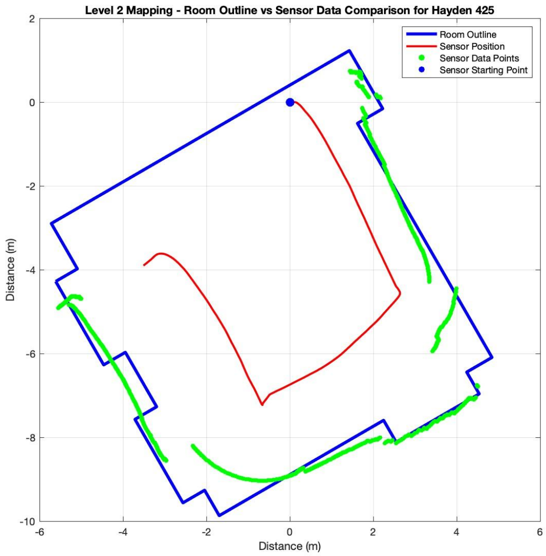

Figure 6 - Example of Level 2 Results

With this model, we continued to experience some beam angle error, particularly at sharp corners, but since we were able to now maintain a more perpendicular orientation, the effect of this issue was reduced.

Our primary difficulty arose due to drift in the IMU's acceleration. Even with the device's on-board corrections, the sensor's position tracking accumulated error over time as the integral of trending acceleration values added up. Detrending and filtering were applied post-collection to improve the accuracy of our results, but the performance of the onboard compensation fluctuated notably from run to run and we did not consistently get results like those shown in Figure 6.

We also addressed two issues that occurred with our UT. The sensor outputs sound waves at 42kHz, and while this is a rather high and specific frequency, the sensor would occasionally saturate at its minimum detection value of 30cm, presumably due to inaudible environmental noise. It would also 'time out' and report its max range of 5m if no reflection was detected. We approximated range values in these instances by assigning the last error-free range value to errant readings.

Our Level 2 visualization was also more complex, as there was now object mapping points and sensor pose information to display. With RViz still configured to maintain a world coordinate system display, we processed IMU acceleration data in real time to output the changing position of the sensor and displayed the relative motion through the use of our 'tf_visualizer' node built via ROS' Quaternion-based coordinate frame handling package, 'tf'. Unfortunately, we experienced ongoing difficulty in reconciling the differences between our post-processing data analysis and our real-time dead-reckoning methods. The following figure shows an RViz display of the first half of the data recorded and shown in Figure 6, before it began to diverge drastically due to an unsolved error. These results were admittedly unsatisfactory, but the process of pursuing this complex implementation provided excellent learning and potentially a launching point for further development.

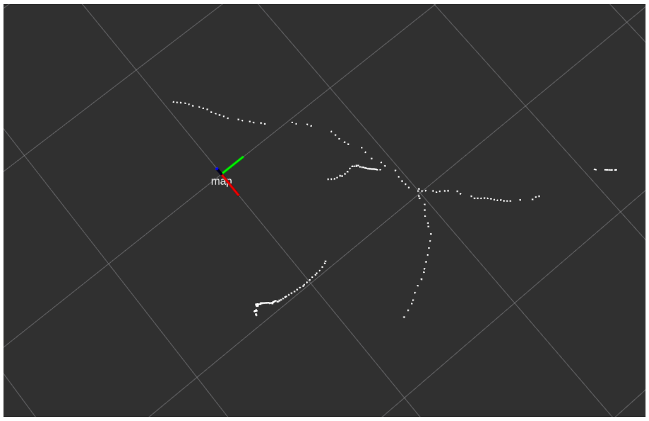

Figure 7 - Level 2
RViz Display

Level Three

Our Level 3 goal was to allow for unconstrained 3D movement and mapping. While we did not reach a point where we could implement this model for data analysis, we were able to pair the orientation tracking of the IMU with the UT's range data in RVIZ to begin investigating 3D mapping. Through the quaternion-based reference frame tracking provided by ROS' tf package, we were able to use the IMU's Yaw, Pitch, and Roll and project our UT's range data along the continuously tracked front-facing sensor X-axis. This now mapped our object points with (x, y, z) components. The room from our Level 1 mapping example was again used. See Figure 8 for a photo of the corner of the room and the 3D representation that was obtained.

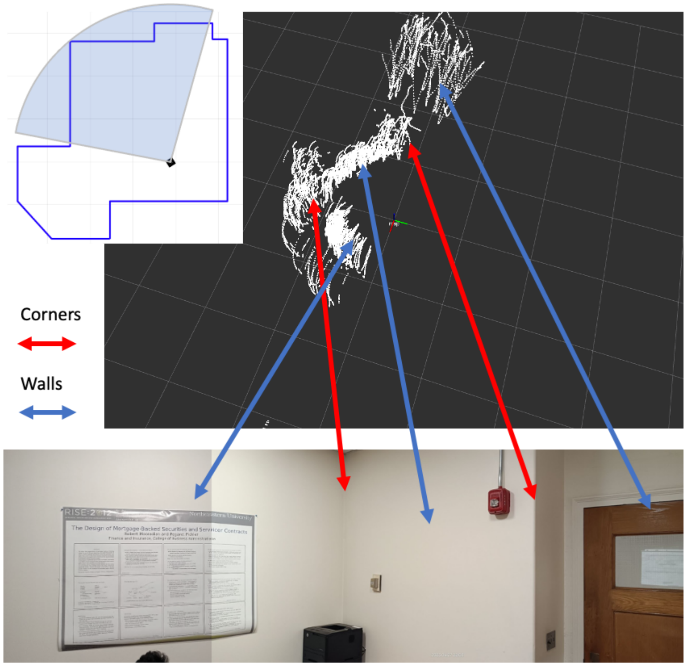

Figure 8 - Level 3 RViz Display

Figure 8 at the left shows the visualization results for when the sensor was moved throughout a wide range of yaw, pitch, and roll values, all while keeping the sensor origin as fixed in space as possible. The RViz output demonstrates the presence of the L-shaped feature of the corner of the room, as well as the walls / surfaces and the edges present in the corner.

In Figure 9 at the right, the same RViz results are shown from a different perspective, demonstrating the previously discussed 'time out' errors that cause incorrect PointCloud data points at 5m away from the sensor origin.

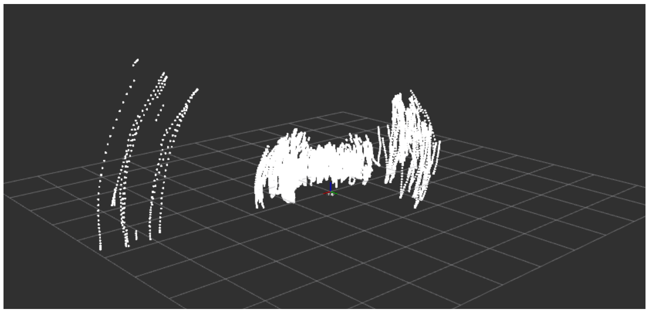

Figure 9 - Alternate Angle of Level 3 RViz Display

References:

- Writing a tf2 package: http://wiki.ros.org/tf2/Tutorials/Writing%20a%20tf2%20broadcaster%20%28Python%29
- Writing a Point Cloud message: http://docs.ros.org/en/melodic/api/sensor_msgs/html/msg/PointCloud.html
- HRUSB-MaxSonar Ultrasonic sensor datasheet: https://maxbotix.com/pages/hrusb-maxsonar-ez-datasheet
- Ultrasonic sensor : https://www.digikey.com/en/products/detail/maxbotix-inc/MB1433-000/7896842
- Vectornav(VN-100): https://www.vectornav.com/products/detail/vn-100