Team 8: High Speed SLAM Implementation on an Inverted Pendulum Robot

Joseph Abrash jabrash@umich.edu
Mohieddine Amine mohie@umich.edu
Wei Jian wwjian@umich.edu
Chi-Kuan Lin chikuanl@umich.edu
Emily Sheetz esheetz@umich.edu

I. Introduction

Modern robotics applications are pushing robots further into environments that are more demanding and challenging, yet are more likely to be experienced as robots integrate into everyday operation. For household robots to become widely used in everyday life, cost of hardware and the trade-off between computational complexity and accuracy of algorithms must be considered. We present **SodaBot**, the inexpensive, highly customizable, snack delivering household robot. SodaBot is an inverted pendulum robot designed using inexpensive hardware that can be customized by users to fit their needs. Its inverted pendulum design introduces challenges with balance and SLAM implementation so the system can safely and accurately map and navigate users' homes. Our approach transforms sensor measurements from 3D space into the 2D map plane. This allows the robot to take advantage of 3D information caused by non-planar and high-speed motion to make a more accurate but computationally efficient 2D map.

The rest of the paper is outlined as follows: Section II formulates the problem; Section III motivates the problem and explains why it is worth investigating; Section IV describes the physical robot; Section V explores related work; Section VI details our approach to the problem; Section VII describes experiments and the results of our approach; Section VIII specifies how our work could be applied as a practical household robot; Section IX discusses limitations for our approach and opportunities for future research; Section X summarizes the presented work.

II. Problem Formulation

In this paper, we implement SLAM on an inverted pendulum robot with a LIDAR sensor. We target the challenge of creating and navigating accurate maps of the environment under disturbances caused by the moving body frame of the robot due to balancing. Balancing results in LIDAR measurements outside of the desired 2D map plane. These 3D sensor readings must be transformed in order to perform SLAM accurately and create a 2D map of the environment.

III. Motivation

SodaBot, the inverted pendulum robot, is specifically designed using inexpensive and customizable hardware. This is significant as cost and customizability will be important factors as household robots become more widely used in everyday life. Furthermore, household robots will be tasked with operating quickly and safely for practical use, requiring a trade-off between algorithmic efficiency and accuracy. Performing SLAM on an inverted pendulum robot requires that we develop a computationally efficient transformation so SodaBot can transform 3D sensor measurements into the 2D map plane to quickly map the environment. Limiting the problem to 2D SLAM makes our algorithm more computationally efficient while taking advantage of 3D information, resulting in a map that is safe for household operation. This project is interesting because it explores the trade-offs between inexpensive hardware and resulting algorithmic challenges as well as between computational efficiency and safety of operation.

IV. System Description

Our platform for this project consists of a Plexiglass and aluminum frame onto which we mount two 12 Volt Polulu Motors (42 radians per second no-load speed), IMU, LIDAR, and micro-controllers. The IMU provides information about the changing body frame angles, which is used to balance the robot. An RP LIDAR A2 running on a Raspberry Pi 3 Micro-Controller is responsible for providing the necessary sensor updates for building the map of the environment of operation. The LIDAR is mounted on a specially designed 3D printed plate. A Beaglebone Green Micro-Controller with a custom cape is programmed to control the motors. The two micro-controllers run Linux OS (Debian). The Lightweight Communication and Marshalling (LCM) protocol is used for transferring data between the operating programs. Figure 1 shows the physical system.

V. Related Work

Our high-level objective is implementing a system that uses the state-of-the-art SLAM algorithms while serving as a candidate contribution for the SLAM field of research. Implementing SLAM on an inverted pendulum robot introduces three challenges addressed by previous works: practical balancing and operation, utilizing 3D information to create 2D maps, and efficiently performing SLAM. Each of these challenges and key works are discussed below.

A. Applications Requiring Balance

An inverted pendulum robot serves as a scaled down version of other robot systems with additional motion caused by movement of the robot. Adachi et al. introduce Walk'n

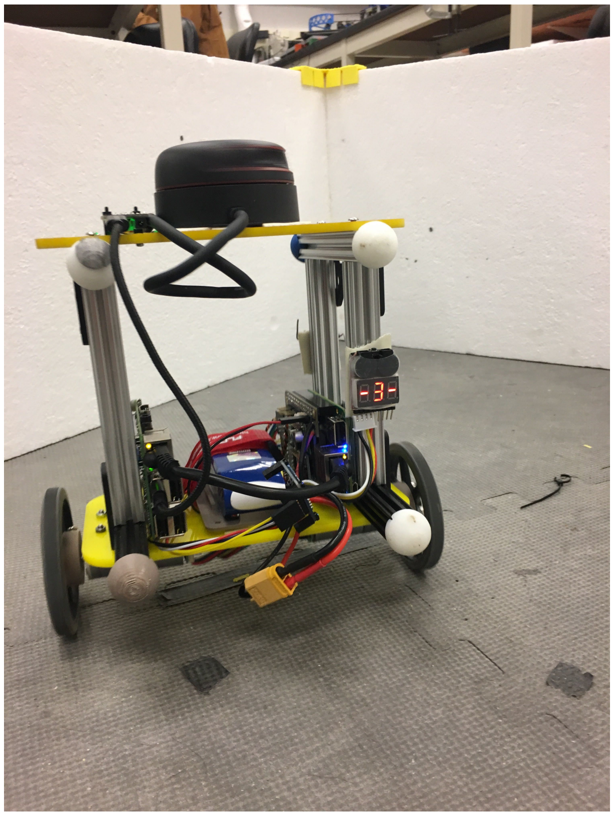

Fig. 1: The physical system for the inverted pendulum robot.

Roll, a leg-wheel hybrid mobile robot, that combines wheels' speed and efficiency and legs' ability to traverse uneven and rough terrain [1]. Linear movement of rotor-based Unmanned Aerial Vehicles (UAVs) results in additional pitch motion [7], similar to that caused by movement of an inverted pendulum robot. The University of Michigan's bipedal Cassie robot must balance while it moves [6]. Similar humanoid robots must account for high linear and angular accelerations in 3D when mapping and navigating environments [8]. Research using inverted pendulum robots, leg-wheel hybrid mobile robots, UAVs, bipedal robots, and humanoid robots demonstrates the desire for robots that move in different ways and through different types of environments. Accounting for motion caused by the robot's hardware is an important challenge as mobile robots move in more diverse and interesting ways.

B. 2D Mapping from 3D Perception

The greatest challenge of this project is creating a 2D map based on 3D sensor data. Most SLAM implementations make the *2D assumption*, which states that observing a cross section of the environment is sufficient to represent the entire environment, allowing for precise navigation. Work by Wulf et al. relaxes the 2D assumption by using 3D sensor data to create a 2D map of a cluttered environment. They present a heuristic for creating *virtual 2D scans* from 3D sensor data using orthogonal projections [11]. Incorporating 3D data built more accurate maps while maintaining the relative computational efficiency of 2D SLAM algorithms. The method for creating virtual 2D scans inspired the method used in this project for transforming the LIDAR sensor readings in 3D space into the 2D map plane. The details of our approach will be described in more detail in Section VI-B.

C. SLAM

With the considerable advancement in SLAM over the past three decades, SLAM problems have branched out into different combinations of operating conditions. Cadena et al. state that the 2D SLAM problem is largely solved [4]. Current research in SLAM investigates dynamic environments, high-dimensional state spaces, and complex motion and sensor models. For example, Vidal et al. explore performing visual SLAM to account for high velocities using high-speed cameras [10]. The inverted pendulum robot used in this work similarly must perform SLAM in the presence of complex motion and sensor models and high velocities caused by balancing the robot.

We approach the SLAM problem for the inverted pendulum robot as particle filter robot localization and occupancy grid mapping, as described in the *Probabilistic Robotics* textbook [9]. Existing challenges in SLAM are concerned with computational complexity, specifically the additional complexity that comes with 3D SLAM [2]. For this reason, we reduce computational complexity by creating a 2D map while accommodating for the challenge of non-planar and high-speed motion.

VI. Methods

In order to implement SLAM on an inverted pendulum robot and account for the changes in pitch that occur due to balancing, our method must address the three challenges described in Section V, specifically: balancing, mapping in 2D from 3D sensor data, and efficiently implementing SLAM. Our approach to addressing these particular challenges are described in the following sections.

A. Balancing the Inverted Pendulum Robot

The inverted pendulum robot is balanced using a Proportional Integral Derivative (PID) controller. PID controllers output control commands to change the state of the robot based on the current, cumulative past, and expected future error between the current state and desired state of the robot.

Let $\boldsymbol{x}_t$ be the current state of the robot, $\hat{\boldsymbol{x}}$ be the desired state of the robot. The current error at timestep t is $e(t) = \hat{\boldsymbol{x}} - \boldsymbol{x}_t$. The control $\boldsymbol{u}_t$ that will correct the current state to match the desired state is:

$$\boldsymbol{u}_t = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{1}$$

The gains K_p, K_i, and K_d affect how much the proportional, integral, and derivative terms, respectively, affect the control command $\boldsymbol{u}_t$. These coefficients need to be tuned for the particular system being controlled. By tuning these gains, the inverted pendulum robot is able to balance.

Changing the nature of the system means that the gains must be tuned again. The first step of the project was to balance the inverted pendulum robot. However, mounting the LIDAR sensor on the robot in order to perform SLAM changed the system, meaning the PID controller used for balancing the robot needed to be tuned again.

The intended application for the inverted pendulum robot is SodaBot, a snack delivering household robot. While carrying snacks, SodaBot's weight could change significantly, meaning the PID controller it uses to balance on its own may not be effective. To address this, our inverted pendulum robot system could be extended with a hybrid controller. This hybrid controller could switch between PID controllers tuned specifically for snacks of certain weights. This would make our inverted pendulum robot an effective platform for delivering snacks in household settings.

B. Transforming 3D Data into 2D

Most 2D SLAM research operates on the 2D assumption, which states that the cross section of the environment observed by the sensor provides enough information for the robot to navigate the environment. This project relaxes the 2D assumption by incorporating sensor measurements taken in 3D outside of the 2D map plane.

Balancing the robot will result in changes in the pitch angle of the robot body frame. The changing pitch will cause the LIDAR to take measurements in 3D outside of the 2D map plane. In order to effectively perform 2D SLAM on this 3D sensor data, we must transform the 3D measurements and project them into the 2D map plane. The transformation we use is similar to that presented in [11], as discussed in Section V-B.

We introduce the following notation:

- θ, the pitch angle induced by robot balancing.
- r, the range measured by the LIDAR sensor, which may lie outside of the 2D map plane.
- ϕ, the bearing measured by the LIDAR sensor.
- (x, y, z), the 3D point being observed by LIDAR measurement (r, ϕ) on some obstacle.
- r^*, the virtual range measurement in the 2D map plane, which is a transformation of measured range r and pitch angle θ.
- (x^*, y^*, z^*), the virtual 3D point corresponding to virtual measurement (r^*, ϕ) on the obstacle.

Suppose at some time t, the robot has a pitch angle of θ and the LIDAR observes range and bearing (r, ϕ) to some obstacle. We can call the observed point on the obstacle (x, y, z). We want to find the virtual range r^* that lies in the 2D map plane and corresponds to virtual point (x^*, y^*, z^*) on the obstacle. In order to perform this transformation from measured range r to virtual range r^*, we perform an orthogonal projection. This means that to transform measured point (x, y, z) to virtual point (x^*, y^*, z^*), we can set $z^* = 0$.

Since we are working with a range-bearing sensor model, we know that:

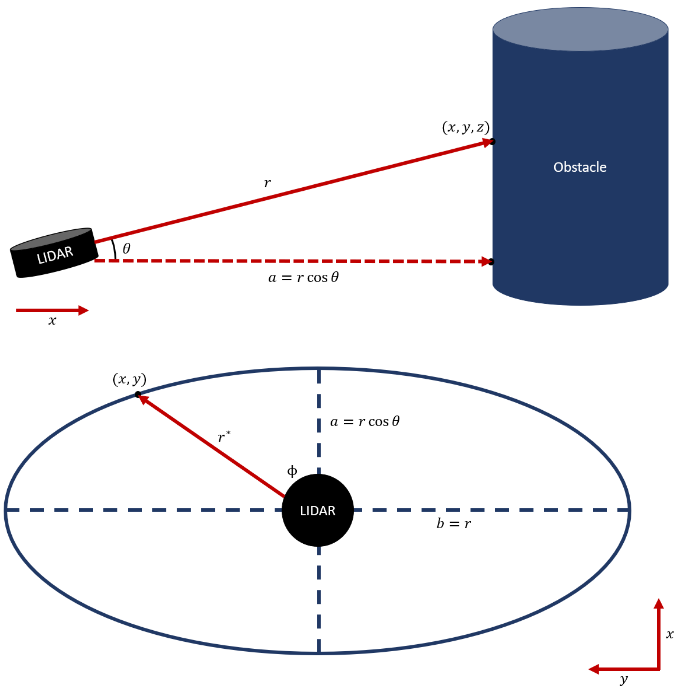

Fig. 2: Depiction of elliptical projection approach. The top image shows the derivation of one of the axes of the ellipse. The bottom image shows the ellipse caused by projecting the circular LIDAR scan into a 2D plane.

$$\begin{aligned} x &= r\cos\phi \quad & x^* &= r^*\cos\phi \\ y &= r\sin\phi \quad & y^* &= r^*\sin\phi \end{aligned} \tag{2}$$

From this foundation, we explored two types of projections, which we will call the elliptical projection and the plane projection. Each of these projection approaches will be described in the following sections.

1) Elliptical Projection: To derive the virtual range r^* in the 2D map plane, we can consider the ellipse created by projecting the 3D circular LIDAR scan into the 2D map plane, as depicted in Figure 2. Recall the equation of an ellipse:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{3}$$

The range scans taken at the side of the LIDAR are not affected by the changing pitch angle of the robot, meaning the measured range values represent the true range to the surrounding obstacles. However, the range scans taken at the front of the robot are affected by the changing pitch angle. This differentiates the major and minor axes of the ellipse. Based on the top image in Figure 2, we can see that the axes of the ellipse are:

$$\begin{aligned} a &= r\cos\theta \\ b &= r \end{aligned} \tag{4}$$

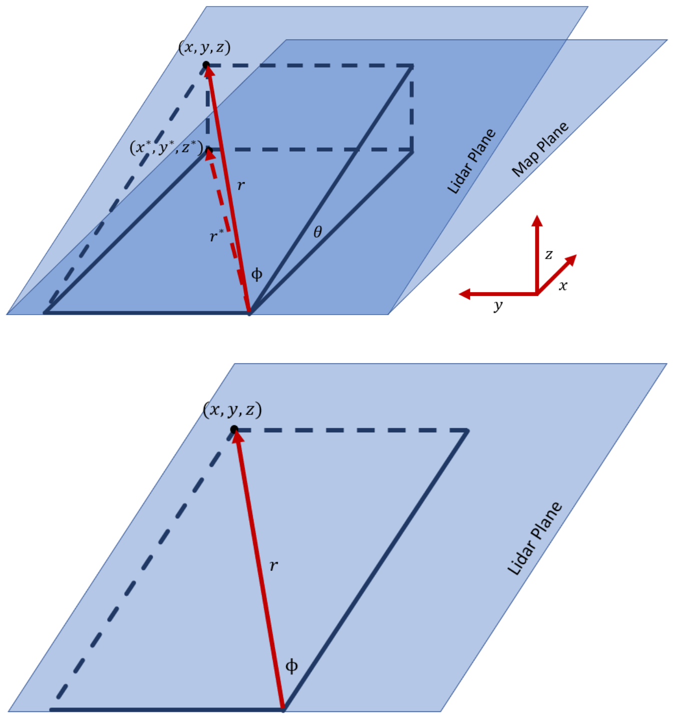

Fig. 3: Depiction of plane projection approach. The top image shows how the LIDAR plane intersects the map plane. The bottom image shows the LIDAR plane in isolation.

We can substitute x^* and y^* into the equation of an ellipse based on the computations in Equation 2 and substitute in the major and minor axes based on Equation 4.

$$\frac{(r^* \cos\phi)^2}{(r\cos\theta)^2} + \frac{(r^* \sin\phi)^2}{r^2} = 1 \tag{5}$$

Solving for virtual range r^*, we obtain our elliptical projection transformation:

$$r^* = \frac{r\cos\theta}{\sqrt{\cos^2\phi + \sin^2\phi\cos^2\phi}} \tag{6}$$

2) Plane Projection: An alternative transformation of the measured range r to virtual range r^* is derived from a series of orthogonal projections based on the intersections of the LIDAR plane and map plane, as depicted in Figure 3. We have already computed x^* and y^* in Equation 2. However, in this formulation, we see that our computation of $x*$ is inaccurate as it still lies in the LIDAR plane rather than the map plane. We need to perform an additional transformation on x^* to put it into the map plane. Our computed value of y^* is the same between the LIDAR and map planes. This gives us our desired virtual point:

$$\begin{aligned} x^* &= r\cos\phi\cos\theta \\ y^* &= r\sin\phi \end{aligned} \tag{7}$$

We can then solve for virtual range r^*, giving us our plane projection transformation:

$$\begin{aligned} r^* &= \sqrt{(x^*)^2 + (y^*)^2} \\ r^* &= r\sqrt{\cos^2\phi\cos^2\theta + \sin^2\phi} \end{aligned} \tag{8}$$

Both the elliptical projection and the plane projection transform measured sensor readings (r, ϕ) in 3D space into virtual readings (r^*, ϕ) in the 2D map plane. Though they formulate the problem differently, both approaches rely on an orthogonal projection from the 3D LIDAR scan to the 2D map plane.

C. Performing SLAM

Once the LIDAR measurement data is transformed to the 2D map plane, the map of the environment can be built using a state-of-the-art SLAM algorithm. This project utilizes particle filter localization and occupancy grid mapping. Particle filter localization represents the probability distribution of the robot location as a set of weighted samples or particles. Occupancy grid mapping discretizes the map into grid cells and determines whether each cell is occupied or free.

SLAM algorithms estimate the joint posterior probability distribution of the current robot pose $\boldsymbol{x}_t$ and the map $\boldsymbol{m}$ given controls $\boldsymbol{u}_t$, observations $\boldsymbol{z}_t$, and initial robot pose $\boldsymbol{x}_0$:

$$p(\boldsymbol{x}_t, \boldsymbol{m} | \boldsymbol{u}_{0:t}, \boldsymbol{z}_{0:t}, \boldsymbol{x}_0) \tag{9}$$

Rather than performing inference on the entire joint probability distribution, our SLAM algorithm factors the joint probability distribution in Equation 9 into separate robot and map components:

$$\begin{aligned} &p(\boldsymbol{x}_{0:t}, \boldsymbol{m} | \boldsymbol{u}_{0:t}, \boldsymbol{z}_{0:t}, \boldsymbol{x}_\mathbf{0}) \\ &\quad = p(\boldsymbol{x}_{0:t} | \boldsymbol{u}_{0:t}, \boldsymbol{z}_{0:t}, \boldsymbol{x}_\mathbf{0}) p(\boldsymbol{m} | \boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t}) \end{aligned} \tag{10}$$

Note that Equation 10 differs from Equation 9 in that the probability distribution considers the entire robot trajectory $\boldsymbol{x}_{0:t}$ rather than the current robot location $\boldsymbol{x}_t$. This is because map locations are independent when conditioned on the trajectory [5]. We will address the robot and map components in Equation 10 separately in the following sections.

1) Particle Filter Localization: To localize the position of the robot, we use a particle filter [9]. The particle filter represents the robot component of the factored joint probability distribution in Equation 10 as sets of N particles and their weights:

$$\left\{ \boldsymbol{w}_t^{(i)}, \boldsymbol{X}_t^{(i)} \right\}_i^N \tag{11}$$

To estimate the robot position, particle filters sample particles, weigh particles, and resample if necessary. Samples are chosen based on a proposal distribution, which is generally the motion model. In the case of the inverted pendulum robot, the motion model depends on odometry commands—based on the odometry motion model, $\boldsymbol{u}_t = [\delta_{rot1}, \delta_{trans}, \delta_{rot2}]^T$—and the forward and backward motion caused by balancing. Weighing the particles is generally based on the observation model. For this project, this is a range-bearing sensor model. Resampling occurs to combat degeneration of samples, but sampling when it is not necessary leads to sample impoverishment. Using these steps, particle filters propagate particles to maintain multiple hypotheses over the location and uncertainty of the robot.

2) Occupancy Grid Mapping: In this project, the map is computed using occupancy grid mapping [9]. Occupancy grid maps factor the map into individual grid cells:

$$\boldsymbol{m} = \{m_i\} \tag{12}$$

The grid cells are assumed to be independent. This means that we can compute the map component of the factored joint probability distribution in Equation 10:

$$p(\boldsymbol{m}|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t}) = \prod_i p(m_i|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t}) \tag{13}$$

Each individual grid cell m_i represents the probability that the particular map location is occupied. For example, if the occupancy of a map location is known with absolute certainty, $m_i = 0$ indicates the grid cell is free and $m_i = 1$ indicates the grid cell is occupied. Multiplying these probabilities to obtain the total map probability as in Equation 13 leads to numerical instability, so occupancy grid mapping algorithms generally utilize log-odds notation. Let $m_i = p(m_i|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t})$ be represented by the quantity $l_{t,i}$ where:

$$l_{t,i} = \log \frac{p(m_i|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t})}{1 - p(m_i|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t})} \tag{14}$$

To recover the probability $p(m_i|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t})$, we can compute:

$$p(m_i|\boldsymbol{x}_{0:t}, \boldsymbol{z}_{0:t}) = 1 - \frac{1}{1 + \exp\{l_{t,i}\}} \tag{15}$$

Utilizing log-odds notation to represent the probability of occupancy of a particular grid cell m_i leads to numerical stability in occupancy grid mapping algorithms. Mapping the occupancy of map locations shows where the robot can safely navigate its environment.

3) 3D Occupancy Grid Mapping: Though the focus of the project is to create a 2D map based on 3D data, we can also create a 3D map. The occupancy grid mapping algorithm is similar to that described in Section VI-C2, but extended using 3D Bresenham's algorithm [3] to update the map grid cells in 3D space. Bresenham's algorithm finds the grid cells in between the known start and end points of the LIDAR range. These grid cells are then updated according to the occupancy grid mapping algorithm. Through this extension, we can perform 2D particle filter localization and 3D occupancy grid mapping simultaneously.

VII. Results

A description of the project and results can be found on the SodaBot website.[1] All code is contained in our public GitHub repository.[2]

[1]https://thesodabotgroup.github.io/Sodabot_Team8/

[2]https://github.com/TheSodabotGroup/Sodabot_Team8

Using the methods described in Section VI, the inverted pendulum robot is able to balance, transform 3D LIDAR scans into the 2D map plane, and perform SLAM on the transformed sensor data to localize the robot and create an occupancy grid map of the environment. We tested our methods to compare the accuracy of the elliptical and plane projections. We also explored creating 2D and 3D maps. All of our experiments were performed by manually controlling the robot through remote controls. Autonomous exploration was not used as the focus of our work is to create an accurate maps from 3D sensor data. The results of the 2D map experiments are discussed in Section VII-A. The results of the 3D map experiments are discussed in Section VII-B.

A. 2D SLAM

Figure 4 shows the 2D occupancy grid maps created during our experiments. We can see that without transforming the raw sensor data, the resulting map is very noisy. The elliptical projection improves the resolution of the map significantly, especially along the walls. The plane projection further improves the resolution of the map. We hypothesize that the plane projection may be more accurate because it transforms each range measurement individually whereas the elliptical projection considers the LIDAR scan in its entirety when transforming the measured range. The results of our experiments demonstrate that we can use an inverted pendulum robot to incorporate 3D sensor data into an accurate 2D map of the environment.

B. 3D SLAM

Figure 5 compares the 2D occupancy grid map created from transformed sensor data and the corresponding 3D occupancy grid map created from raw sensor data. We can see that when viewed from the same angle, the 2D and 3D maps are very similar. This further demonstrates the accuracy of the described transformations. However, we can see that taking advantage of 3D data provides a richer map of the environment, as seen in the bottom image of Figure 5.

VIII. Applications

The proposed application of implementing SLAM accurately on an inverted pendulum robot is **SodaBot**, the inexpensive, highly customizable, snack delivering household robot. Because SodaBot transforms its 3D sensor data to make accurate 2D maps of the environment, SodaBot can safely navigate through homes without damaging furniture or injuring occupants. The hardware makes the system inexpensive. SodaBot is specifically designed with components that can be easily taken apart and reconfigured to meet users' needs. SodaBot's size makes it perfect for delivering cans of soda or other lightweight snacks to users while avoiding the clutter in typical household environments.

The methods described in this project—balancing a robot designed with inexpensive and customizable parts, transforming 3D sensor data into the 2D map plane, and using state-of-the-art SLAM algorithms to create occupancy grid maps of

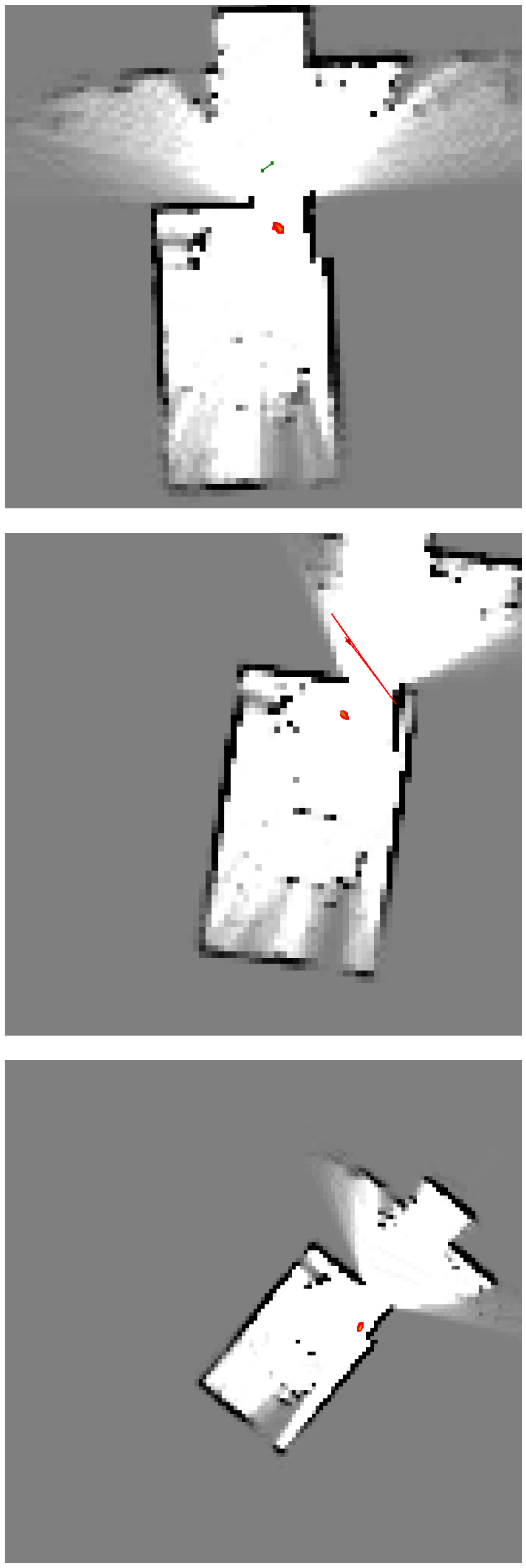

Fig. 4: Comparison of 2D occupancy grid maps with different projections. The top image shows the map created with no projection, the middle image shows the map with the elliptical projection, and the bottom image shows the map with the plane projection.

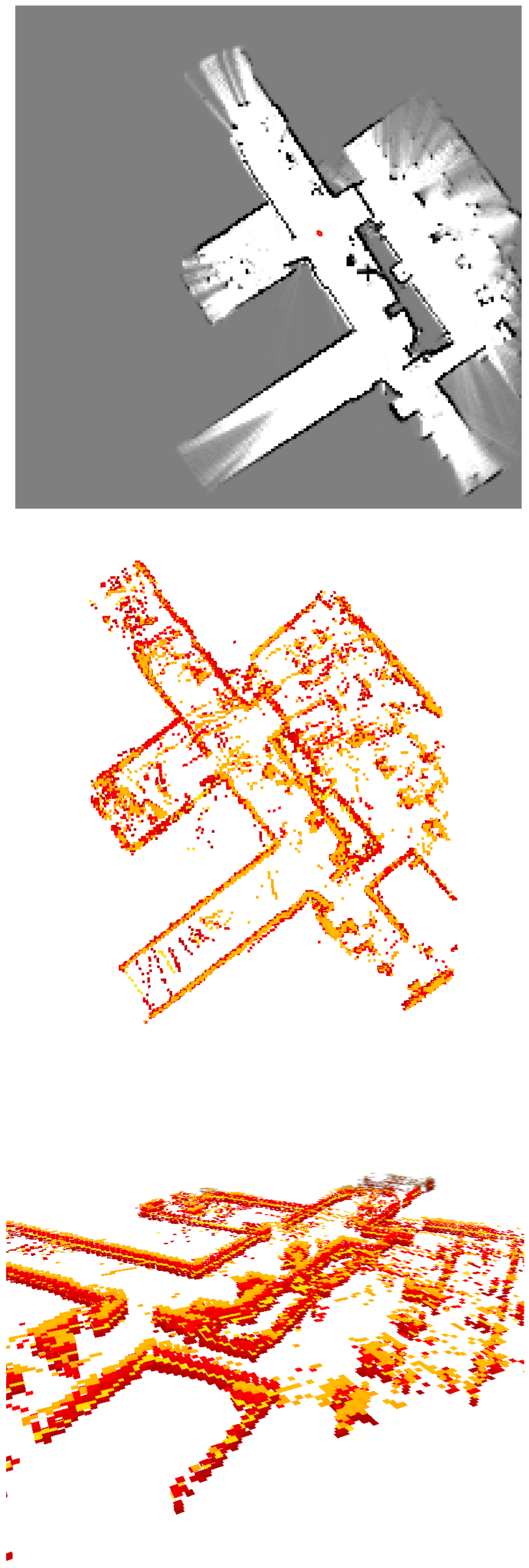

Fig. 5: Comparison of 2D map and 3D map using transformed and raw sensor data, respectively. The top image is the 2D map, the middle image is the 3D map, and the bottom image is the same 3D map skewed to show the third dimension. These experiments were run using the elliptical projection.

the environment—can be deployed on a household robot such as SodaBot for efficient, accurate, and safe navigation.

IX. Future Work

Future work would address some of the limitations of this project and extend the system to more advanced capabilities. This project evaluates the SLAM implementation on an inverted pendulum robot in a controlled laboratory environment. Since our proposed application is a household environment, more insight could be obtained by testing the system in a household environment. Household environments present additional challenges of clutter, more variety of obstacle size and shape, and dynamic obstacles such as humans or animals. Though the results presented here demonstrate the promise of our proposed methods, more experiments run in household settings would provide additional insight for evaluation.

Our methods could be improved with a more complicated sensor data transformation that allows us take more advantage of the 3D sensor data when creating our maps. Orthogonal projections enforce assumptions on the environment and may not accurately capture the variety of objects in a household environment. Future research could explore more complex data transformations or explore the computational complexity of creating 3D maps on inexpensive hardware.

As mentioned in Section VI-A, changing the robot changes the dynamics of the physical system and requires the PID controller responsible for balance to be retuned. In the proposed application, using SodaBot to transport snacks could change the weight of the robot, making its PID controller ineffective. An extension of this work could be to implement a hybrid controller that switches between PID controllers depending on the weight of the snack. The individual PID controllers could be tuned for snacks of particular weights based on snacks the system is likely to transport in household settings. Addressing the changing nature of the system during operation and snack delivery would extend the system used for this project and make it a more viable option for household environments.

X. Conclusion

We present a SLAM implementation on an inverted pendulum robot that transformed 3D sensor data to create a 2D map of the environment. The robot is balanced using a PID controller. The 3D sensor data is transformed into the 2D map plane using orthogonal projections. The transformed data is processed using particle filter localization to determine the robot position and occupancy grid mapping to create a safe and navigable map of the environment. Our methods work well and our experiments demonstrate that the plane projection provides the most accurate transformation of the sensor data. We also demonstrate that our methods can be extended to create richer 3D maps of the environment. Our proposed application for this system is SodaBot, the inexpensive, highly customizable, snack delivering household robot. Since the resulting occupancy grid maps are accurate and efficient to compute, our system would work well in cluttered household environments where safety is a concern. Further research could improve the sensor data transformation and practicality of the system in order to make it a viable option for efficient, accurate, and safe navigation through household environments.

References

[1] Hironori Adachi, Noriho Koyachi, Tatsuo Arai, A Shimiza, and Yashiroh Nogami. Mechanism and control of a leg-wheel hybrid mobile robot. In *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No. 99CH36289)*, volume 3, pages 1792–1797. IEEE, 1999.

[2] Tim Bailey and Hugh Durrant-Whyte. Simultaneous Localization and Mapping (SLAM): Part II. *IEEE Robotics and Automation Magazine*, 13(3):108–117, 2006.

[3] Jack Bresenham. An incremental algorithm for digital plotting. 1963.

[4] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J.J. Leonard. Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age. *IEEE Transactions on Robotics*, 32(6):13091332, 2016.

[5] Hugh Durrant-Whyte and Tim Bailey. Simultaneous Localization and Mapping: Part I. *IEEE Robotics and Automation Magazine*, 13(2):99–108, 2006.

[6] Ayonga Hereid, Omar Harib, Ross Hartley, Yukai Gong, and Jessy W. Grizzle. Rapid Bipedal Gait Design Using C-FROST with Illustration on a Cassie-Series Dynamic Walking Biped. 2019. URL http://arxiv.org/abs/1807.06614.

[7] Taeyoung Lee, Melvin Leok, and N. Harris McClamroch. Geometric Tracking Control of a Quadrotor UAV on SE(3). *Proceedings of the IEEE Conference on Decision and Control*, pages 5420–5425, 2010.

[8] Olivier Stasse, Andrew J. Davison, Ramzi Sellaouti, and Kazuhito Yokoi. Real-time 3D SLAM for humanoid robot considering pattern generator information. *IEEE International Conference on Intelligent Robots and Systems*, pages 348–355, 2006.

[9] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT press, 2005.

[10] Antoni Rosinol Vidal, Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios. *IEEE Robotics and Automation Letters*, 3(2):994–1001, 2018.

[11] O. Wulf, K.O. Arras, H.I. Christensen, and B. Wagner. 2D Mapping of Cluttered Indoor Environments by Means of 3D Perception. pages 4204–4209 Vol.4, 2004.