12-2007

Odometry Correction of a Mobile Robot Using a Range-Finding Laser

Thomas Epton
Clemson University, tom.epton@gmail.com

ODOMETRY CORRECTION OF A MOBILE ROBOT USING A RANGE-FINDING LASER

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Computer Engineering

---

by
Thomas Epton
December 2007

Accepted by:
Dr. Adam W. Hoover, Committee Chair
Dr. Richard R. Brooks
Dr. Stanley T. Birchfield

ABSTRACT

Two methods for improving odometry using a pan-tilt range-finding laser is considered. The first method is a one-dimensional model that uses the laser with a sliding platform. The laser is used to determine how far the platform has moved along a rail. The second method is a two-dimensional model that mounts the laser to a mobile robot. In this model, the laser is used to improve the odometry of the robot. Our results show that one-dimensional model proves our basic geometry is correct, while the two-dimensional model improves the odometry, but does not completely correct it.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

Table Page

CHAPTER 1

INTRODUCTION

This thesis considers the problem of improving odometry using a range-finding laser. An odometer is a device used to estimate the distance traveled by a wheeled vehicle. The distance is measured mechanically by an encoder that counts the number of revolutions of the wheels (axles). Knowing the diameter of the wheel, the odometer is able to find the linear distance by multiplying the encoder's revolution count by the circumference of the wheel. Odometers are reliable over short distances, but they accumulate error over time. There are many reasons for this error, such as slippage between the wheel and the surface being traveled and encoder error. We seek to improve the odometry estimates using a range-finding laser by mounting the laser on a moving vehicle or platform. The laser is then used to continuously track a single point using a pan-tilt unit.

An odometer is most commonly used to track the total distance traveled by a motor vehicle. For this application, the distance estimate from the odometer is usually only used to gauge the general aging, the amount of usage, and the gas mileage of the vehicle. As long as the running estimate of the distance traveled is reasonably accurate, the odometer serves its intended purpose.

The motivation for our work is the problem of self-localization. Self-localization refers to the ability of a wheeled vehicle or moving platform to accurately track its position as it moves through the world. Maintaining an accurate estimate of position and orientation is vital for automated control and navigation. In this case, an odometer must be able to provide a much more accurate estimate of the distance traveled so that it can be used to compute a more accurate estimate of position. The lack of highly accurate odometers is a major limiting factor for the potential in autonomous vehicles and mobile robots. Until a wheeled vehicle can accurately maintain its position and orientation, it will be unable to safely operate in the real world.

1.1 Odometry

Odometry is unreliable because it is subject to many sources of error, which can be categorized as either systematic or non-systematic [11]. Systematic errors are those that depend on the robot independent of the environment. Non-systematic errors, on the other hand, are those that depend

on the environment. Borenstein [2] gives a review of all the different systematic and non-systematic errors for a mobile robot, as shown in Table 1.1.

Systematic Errors	**Non-systematic Errors**
a) Unequal wheel diameters	a) Travel over uneven floors
b) Average of both wheel diameters differs from nominal diameter	b) Travel over unexpected objects on floor
c) Misalignment of wheels	c) Wheel-slippage
d) Uncertainty about the effective wheelbase	d) External forces
e) Limited encoder resolution	e) Internal forces
f) Limited encoder sampling rate	f) Nonpoint wheel contact with the floor

Table 1.1 Summary of Systematic and Non-systematic Errors

1.1.1 Systematic Error

Understanding each of these errors will help explain why odometry error grows over time. First, we will examine the systematic errors. Unequal wheel diameters on a two-wheel robot do not cause much trouble, the robot just turns one way or the other. However, on a four-wheel robot, with each wheel being a drive wheel, the affect of different wheel diameters is far worse. In fact, the odometry would be quite unpredictable in most cases. Next, if the average of both wheel diameters differs from the nominal diameter, then this means the encoders are incorrectly converting the number of revolutions of the wheel to linear distance. A misalignment of the wheels will cause the robot to drift one way or the other without the robot knowing. An uncertainty about the effective wheelbase is caused by non- point wheel contact with the floor. This is the encoders assuming a point-contact with the floor for their distance computations. A limited encoder resolution refers to the number of samples the encoder takes during a revolution of the wheel. A better estimate of distance can be achieved by taking more samples per revolution of the wheel. The limited encoder sampling rate refers to how often the encoder can take a sample. If the sampling rate is slower than the resolution requires, the encoder will miss samples and will therefore translate to error.

Systematic errors can be limited by improving the technology of the robot. The first three causes of error can be fixed by improving the physical robot. Ensuring the wheels are the same diameter will take care of the first two. The third can be taken care of by properly aligning the wheels during the assembly of the robot. The last three causes are the focus of many groups who are trying to make better encoders for robots [13][16][10]. Letting the robot know what kind of wheel is being used

can help remove the amount of uncertainty there is in the effective wheelbase. Making the encoder resolution higher will increase the accuracy of the encoder. However, the resolution is limited by the encoder sampling rate. A higher resolution can be obtained by increasing the sampling rate.

1.1.2 Non-Systematic Error

Now, we will examine the environment dependent errors or non-systematic errors. Traveling over uneven floors causes error because the robot does not have any knowledge of the terrain. It only knows how much each wheel has turned. The same can be said for traveling over unexpected objects on the floor. The wheels have to turn more to go over an object than they do to go the same distance over an unobstructed floor. Wheel-slippage is a major cause of error. This is caused by a number of factors such as slippery floors, over-acceleration, skidding in fast turns, etc. External forces that could cause odometry error would be those such as the robot running into an object or an object hitting the robot, knocking it off the known course. Internal forces that can have the same effect are those such as a caster wheel. A two-wheel robot will have a caster wheel for balance, and when the robot changes direction such as from forward to back, the castor wheel turns and pushes the robot slightly off course. A complement to one of the systematic errors is non-point wheel contact with the floor. In this case, it is referring to the amount of friction there is between the wheel and the floor.

The effect from non-systematic errors can be limited by confining the environment. While this may be feasible in the lab, it is not always the case in the real world. The environment can be controlled in such applications such as in factories and warehouses where it has become cost effective to engineer the space around the robot so that it can do its job as efficiently as possible. This makes up just a small percentage of applications where robots are being designed to work. Other environments include office areas, outdoors, search and rescue areas, etc. For the most part, the robot must adapt to the environment instead of the environment adapting to the robot.

These causes for error are unavoidable in most cases; however, the odometer can still be a valuable tool for self-localization by fusing it with other sensors. By adding another sensor to the robot, the robot can compensate for the errors from the odometer. Many researchers have taken this approach to solve the self-localization problem. Some of these are discussed below and then we will present our approach to this same problem.

1.2 Previous solutions to improving odometry

1.2.1 Evaluation of odometry error to improve odometry

Several researchers have looked at the odometry error of mobile robots in the hope that if the errors can be classified and separated, then certain errors can be reduced or eliminated. Wang [18] looked at this exact problem. In his work, he analyzed the non-systematic errors theoretically and computed the odometry covariance matrix. Using the covariance matrix, he was able to estimate the robot's location from the information provided by the odometry encoders. He was able to do this by introducing a circular adjustment factor into the estimator.

Martinelli [11][12] developed methods for evaluating the odometry error. His intent for evaluating the error was to help reduce it and to know the accuracy of the odometry. In these works, he developed the odometry error model. To compute the global odometry error for a given robot motion, he divided the trajectory into N small segments. He first modeled the elementary error associated with a single segment. He then computed the cumulative error on the global path. Finally, he took the limit as $(N \rightarrow \infty)$.

Meng and Bischoff [15] developed an algorithm to detect non-systematic errors. Their idea was that the most common non-systematic error that caused the greatest decrease in accuracy for the odometry was the wheel-slippage. Because of this, they developed a wheel-slippage detector. Once they are able to detect a slippage, they would then be able to initiate countermeasures to correct the pose measurements. Their idea was to track the ratio of the two drive wheels' displacement increment over time. While their slippage detector helped with navigation, they came to the conclusion that odometry alone is not sufficient to track the robot's long term pose.

Abbas et al. [1] developed a new method for measuring systematic errors in a mobile robot called the Bi-directional Circular Path Test (BCPT). For their test, the robot is programmed to drive in two circles: one clockwise, one counter-clockwise. By measuring the radius of each path, the diameter of the wheels can be determined if they are equal. If the two radii are equal, then the wheel diameters are equal; however if they are not equal, then the wheel diameters are different. Based on the ratio of the wheel diameters, the wheelbase can be determined. They argue that by calculating these two parameters will allow for simple compensations to be made in software to help eliminate systematic errors.

1.2.2 New odometers

New odometers do not necessarily refer to a new type of encoder, it can refer to a robot that utilizes new algorithms and hardware to improve the odometry. This is the case with the OmniMate mobile robot. Borenstein [2] conducted tests using the OmniMate robot to implement a new method for detecting and correcting odometry errors without any extra sensors. The method is called internal position error correction (IPEC), and the OmniMate robot was specifically designed to implement this method. The key to IPEC is having two mobile robots linked so that they correct each other as they move, even during fast motion. The basic steps to the IPEC algorithm are as follows. First, at each sampling interval, the position and orientation of truck A from conventional odometry is computed. Then the direction of the link that connects the centers of trucks A and B is computed. Next, the angle between the direction of motion of truck A and the direction of the link is computed; this is known as the expected angle. Next, the difference between the expected angle and the actual angle is determined. The orientation of truck A is then corrected by this amount just determined. Finally, the position and orientation of truck B is computed from the kinematic chain, not from odometry.

1.2.3 Odometry fused with other sensors

Since knowing the location of a mobile robot is vital for navigation and odometry is unreliable, the next logical step is to add another sensor to the robot to help correct the odometer. This section looks at researchers who have tried to solve the self-localization problem by utilizing the odometer along with another sensor to aid in correcting the odometry.

Chenavier [5] developed a system for an intervention robot for a nuclear site. Since it was not possible to equip the environment with beacons, they developed a technique using the existing structure as beacons. The beacons were located using a camera attached to the robot via a steerable rotating platform. Image processing was used to find suitable structures such as doors, pipes, and corners. For typical triangulation, it is necessary to be able to observe three beacons simultaneously to obtain an estimation of pose. Instead of relying on this, they used an extended Kalman filter with a single observation and the current pose estimation given by the odometer to find the actual position of the robot.

Maeyama et al. [10] tried improving the odometry by integrating a gyroscope with the robot's odometer. The gyroscope was used to detect non-systematic errors. Their method is simple yet

seemingly effective. They take the difference between the angular rates from the gyroscope and the odometer. If the robot incurs no non-systematic errors, the difference will be small. In this case, they are able to measure the drift of the gyroscope using Kalman filtering. They also estimate the actual angular rate of the robot by fusion of the gyroscope and the odometry data using the maximum likelihood estimation. However, if the robot incurs an error such as wheel-slippage, the difference will be large. In this case, the gyroscope data is used to estimate the angular rate of the robot without any input from the odometer. Their results showed that their algorithm works better in outdoor environments than indoor due to the bumpy surfaces of travel.

Koshizen et al. [9] developed a mobile robot localization technique that was based on the fusion of sonar sensors with the odometry. Their purpose was to navigate a mobile robot while avoiding several static and dynamic obstacles of various forms. Their technique for doing so was an enhancement on another technique, which they developed previously [8]. This technique is called the Gaussian Mixture of Bayes with Regularised Expectation Maximization (GMB-REM). The GMB-REM has four basic steps. First, data is collected from the sonar sensors and is normalized to zero mean and unit variance. Second, the training data set is formed to predict the probabilistic conditional distribution using the EM algorithm. The EM algorithm is an iterative technique for maximum likelihood estimation. The third step is cross validation. In this step, a suitable value for the regularizing coefficient is found by finding the value that performs the best on a validation set. The final step is to calculate the probability of the robot's position. The enhanced version of this technique, which is described in their work, takes in the odometry estimates as well instead of relying solely on the sonar data. Their results showed that the new fusion technique out performed their previous technique which did not use the odometry.

Bouvet and Garcia [3] developed an algorithm for mobile robot localization by fusing an odometer and an automatic theodolite. Their algorithm is intended for mobile robots equipped with these two devices. A theodolite is a device for measuring both horizontal and vertical angles for triangulation purposes. Their method is based on the SIVIA (Set Inverter by Interval Analysis) algorithm. Using the assumption that the measurement errors are bounded, and an initial set of admissible solutions is computed. Next, incorrect solutions are eliminated to find the true solution to the localization problem. This method has the advantage over systems such as GPS in that it is usable under bridges and close to buildings. However, the calculations can not be performed in real time, and therefore

limit the capabilities of the system.

Rudolph [16] developed a different technique using the fusion of a gyroscope with the odometry. In this method, he used the gyroscope as a redundant system for the odometer. This method has both the gyroscope and the odometer outputting rotary increments. During straight movement, the rotary increments from the odometer are believed to be true due to gyroscope suffering from drift. In this case, the gyroscope output is corrected using the odometry information. During rotary movement, the gyroscope rotary increments are believed to be true due to the uncertainty of the deviation. The deviation is the difference between the odometry wheel base and the actual wheel base. In this case the odometry is compensated for by the gyroscope output.

Martinelli et. al. [13][14] estimated the odometry error using different filter algorithms as well as fusing the odometry with a range-finding laser to develop a simultaneous localization and odometry error estimation. An augmented Kalman filter is used to estimate the systematic components. This filter estimates a state containing the robot configuration and the systematic component of the odometry error. It uses the odometry readings as inputs and the range-finding laser readings as observations. The non-systematic error is first defined as an overestimated constant. Another Kalman filter is then used to carry out the estimation of the actual non-systematic error. The observations for this filter are represented by two previous robot configurations computed by the augmented Kalman filter. Using this second filter, the systematic parameters are updated regularly with the values computed by the first filter. In their experiment, they move the robot along a 10m square area, which a map has been developed for it. The laser is used to measure 36 distances every time step during the movement. They do not go into any detail of exactly how they aim the laser, so the meaning of the laser measurements is unclear. For the systematic parameters, their results show a rapid variation of the estimated value during the first 30m. However, the overall change never exceed 2 percent after the first 30m. For the non-systematic parameters, they show that the robot is required to move 100m. The overall change after this distance never exceeds 10 percent.

1.2.4 Non-odometry approaches for self-localization

Howard et al. [7] used a team of robots rather than relying on a single robot. The assumptions his team made was that each robot was able to measure the relative pose of nearby robots. Using this measure combined with its measure of its own pose from each robot, they were able to infer the relative pose of every robot in the team. The good thing about this approach is that the environment

doesn't matter. The robots only need to be able to see each other, not some external landmark. The downside to this approach is that multiple robots are needed, and that they have to be able to see each other. There are many cases where this is not feasible, such as search and rescue operations.

Georgiev and Allen [6] combined GPS with vision to compensate for the shortcomings of GPS. GPS works very accurately in open areas; however, in urban areas, tall buildings obstruct the view to the satellites. This is where the vision was used. They performed some image processing to match the view of the camera to an already existing map. While this does offer better performance than GPS alone, there are several drawbacks to this approach. This assumes there is already a map of the area available. Also, similar structures could be misinterpreted. And, like the previous example, this approach will not work well in cases such as search and rescue operations.

Vlassis et al. [17] reviewed a similar approach. They looked at how a robot tries to match its sensory information against an environment model at any instance of time. This is known as a probabilistic map. The robot uses its current position, its new sensory information, and the probabilistic map to best obtain its current position, and sometimes even update its previous position. The biggest drawback to this approach is that it requires a map to be already made, and in a lot of real world applications this is not available.

Calabrese and Indiveri [4] took a completely different route from the rest. In their work, instead of trying determine the robot's pose accurately, they sought how to measure the robot's pose as robust as possible. They argue that the accuracy will take care of itself as technology becomes better and therefore the sensors become better. Their approach uses an omni-vision camera mounted on a mobile robot. An omni-vision camera is a camera that uses a mirror to capture an image a full 360 degrees around the camera. The camera is used to measure the distance from the robot to landmarks in the real world. Their approach uses only two landmarks rather than the typical three needed for basic triangulation. The robustness of their work stems directly from this need for only two landmarks.

1.3 New method for improving odometry

Our method to solve the self-localization problem using a range-finding laser by mounting the laser on a moving vehicle or platform. The laser is used to continuously track a single point using a pan-tilt unit. The laser finds a suitable target on a wall when before the robot starts moving. As the robot moves, the laser is continuously moved using the pan-tilt so that it tracks that target. The

robot's odometry-based pose is constantly corrected based on the distance measurements obtained by the laser. Our technique differs from others' techniques using a range-finding laser in that ours does not require a map to be known a priori. In fact, no map is ever needed for our technique. Also, our technique is based solely on geometry, so computation time is rather low. The assumptions that we make are that there must be a wall flat enough for the laser to obtain accurate readings and that the robot is not moving parallel to that wall. This method will not completely correct the odometry as it is based on the odometry; however, our intent is to improve it. The actual technique is discussed in great detail in the Methods section.

CHAPTER 2

METHODS

2.1 Overview

The basic idea to our method is to mount a pan-tilt laser to a moving platform or vehicle to track a single point as the platform moves. This idea is shown in Figures 2.1 and 2.2. Taking a closer look at Figure 2.1, we see that the pan-tilt unit turns the laser toward the wall and fires the beam. The important information obtained from this is the angle at which the pan-tilt is turned and the distance from the laser to the wall. The platform then moves to some point represented in Figure 2.2. Using the information from part A and by knowing the distance the platform traveled to get from A to B, then we can determine geometrically how far to turn the laser and what distance we should expect to receive from the laser when it fires. The laser then turns to the calculated angle and then fires to measure the actual distance. If the actual distance does not equal the expected distance, then the platform did not travel the distance reported. The actual position of the platform is then calculated geometrically using the actual distance measured by the laser. It needs to be noted that the point being tracked is not a feature point, it is some arbitrary point on the wall.

In this chapter, we develop the geometry and methods to solve this problem. We first develop methods for a one-dimensional model, where the platform is only allowed to move in a straight line. For example, this would be the case for a platform moving along a conveyor belt. We develop this model in order to test the efficacy of our models. We then develop methods for a two-dimensional model, where the platform is allowed to move in an arbitrary flat space. This model could be used for problems involving autonomous vehicles and robot motion. We believe our methods could be extended to a three-dimensional model, but that is beyond the scope of the current work.

2.2 One-Dimension Model

In setting up the one-dimensional case, it is necessary to define certain parameters. Let α be the angle the platform is moving with respect to the wall. We then let γ be the initial angle the pan-tilt turns the laser, which is given by the user. This is allowed to be any arbitrary angle as long as it points the laser toward the wall. Finally, let δ_1 be the angle between laser measurements. In our model, we let this be equal to 20 pan-tilt positions or 0.0185 radians. Two distances d_1 and d_2

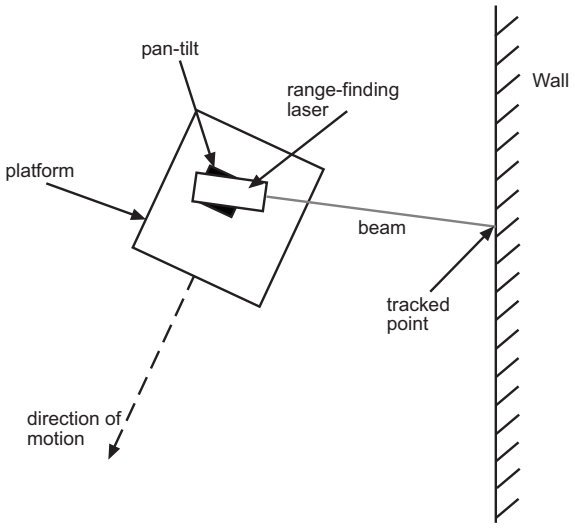

Figure 2.1 Tracking point on wall (part A)

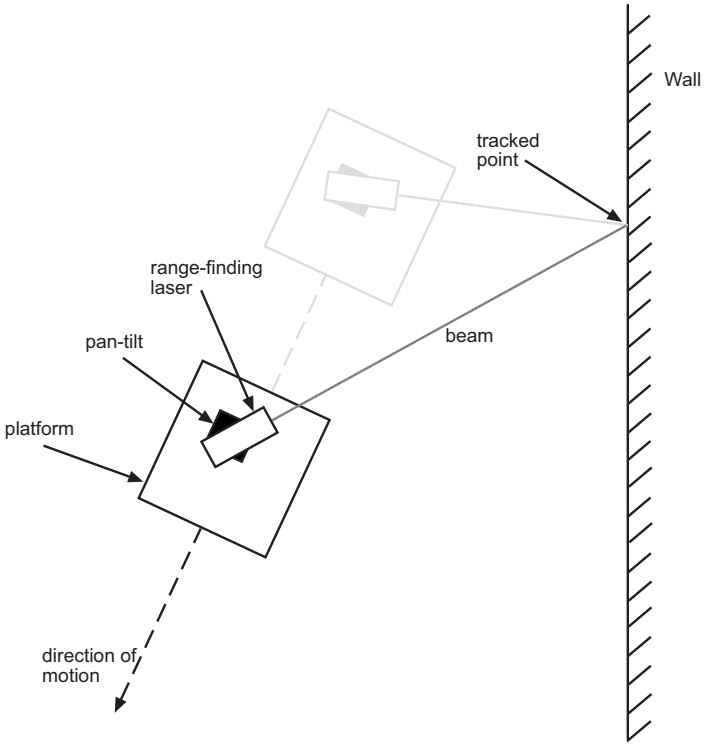

Figure 2.2 Tracking point on wall (part B)

are measured by the laser separated by δ_1. A summary of these parameters are given in Table 2.1. These parameters are used to find the angle of motion (α) as shown in Figure 2.3. This is found using the equations below.

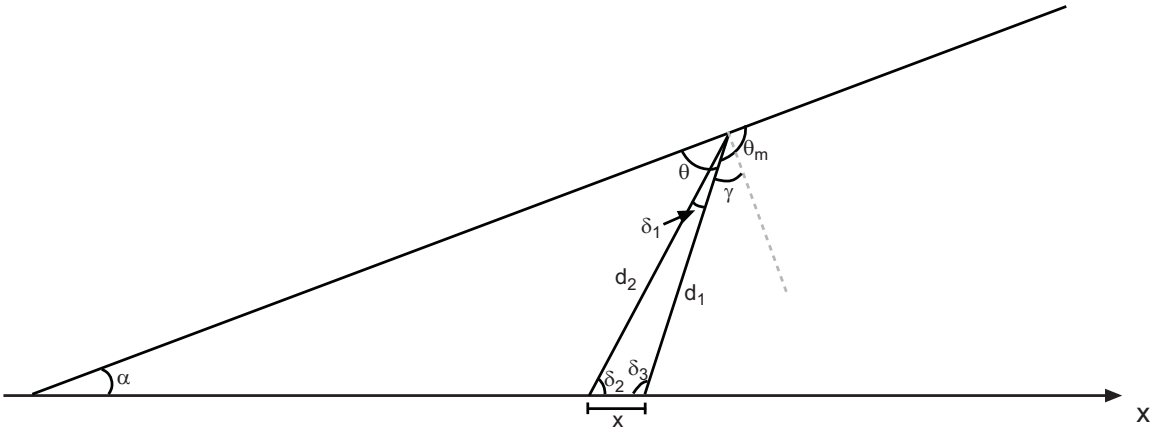

Figure 2.3 1D - Finding wall angle

γ	initial angle	Provided by user
d_1	first distance	Measured by laser
d_2	second distance	Measured by laser
δ_1	angle between laser measurements	0.0185 radians

Table 2.1 Wall angle calculation variables

First, we let θ be the complementary angle to γ, and θ_m be the supplementary angle to θ, as shown by Equations 2.1 and 2.2.

$$\theta = \frac{\pi}{2} - |\gamma| \tag{2.1}$$

$$\theta_m = \frac{\pi}{2} + \gamma \tag{2.2}$$

Given the two measured distances, d_1 and d_2, and the angle between them, δ_1, we can determine the distance x on the wall between the two laser point locations using Equation 2.3.

$$x = \sqrt{{d_1}^2 + {d_2}^2 - 2 \cdot d_1 \cdot d_2 \cdot \cos \delta_1} \tag{2.3}$$

The inner angles of this smaller triangle are found based on the direction the platform is moving. Angle δ_2 is only needed if the first condition for finding angle δ_3 from Equation 2.5 is met. If the laser and platform are in a position such as that shown in Figure 2.3, then the first condition for δ_3

is not met. Therefore, we do not use the calculation of δ_3.

$$\delta_2 = \arcsin\left(\frac{d_1 \sin \delta_1}{x}\right) \tag{2.4}$$

$$\delta_3 = \begin{cases} \pi - (\delta_1 + \delta_2) & ((\gamma < 0)\ AND\ (d_2 > d_1))\ OR\ ((\gamma > 0)\ AND\ (d_1 > d_2)) \\ \arcsin\left(\frac{d_1 \sin \delta_1}{x}\right) & otherwise \end{cases} \tag{2.5}$$

Once we find δ_3, we can then determine the angle of motion, α, using the summation of angles property of triangles. This is shown in Equation 2.6.

$$\alpha = \pi - (\theta + \delta_3) \tag{2.6}$$

After the angle of motion is determined, the platform then moves some distance m. Based on this distance, the angle θ and distance d_2 are calculated. The angle is the angle at which the pan-tilt unit should turn so that the laser points back to the original point on the wall. The distance is the expected distance between the laser and the wall. This is shown by Figure 2.4. A summary of the known parameters are given in Table 2.2. The equations to calculate θ and d_2 are also given below.

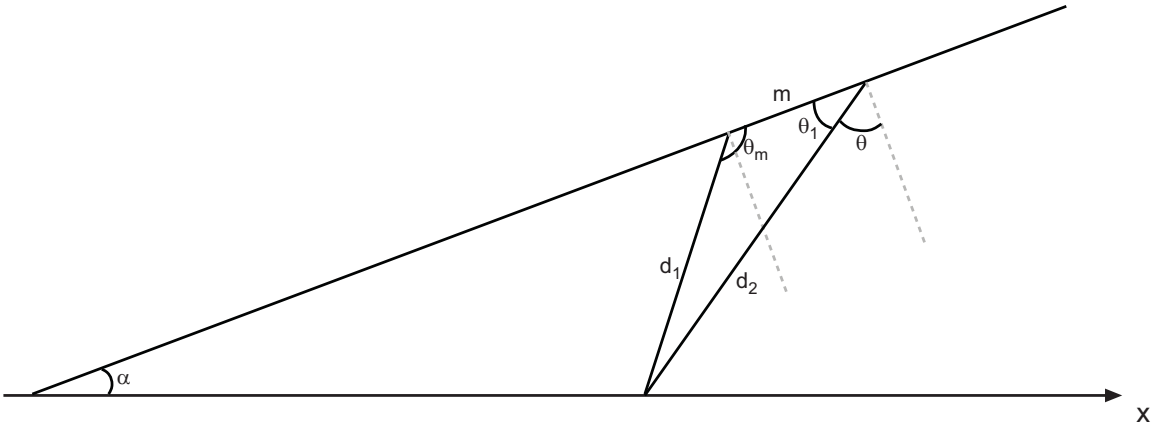

Figure 2.4 1D - Determining pan-tilt angle and expected distance

d_1	d_1	From wall angle calculations
θ_m	θ_m	From wall angle calculations
m	distance moved	Provided by user

Table 2.2 Step 1 calculation variables

Based on the known parameters (d_1, m, θ_m), we calculate the expected distance d_2 using the law of cosines as shown in Equation 2.7.

$$d_2 = \sqrt{{d_1}^2 + m^2 - 2 \cdot d_1 \cdot m \cdot \cos \theta_m} \tag{2.7}$$

Now, using the parameters (d_2, m, d_1), we calculate the inner angle θ_1 by using the law of cosines again shown by Equation 2.8.

$$\theta_1 = \arccos\left(\frac{{d_2}^2 + m^2 - {d_1}^2}{2 \cdot d_2 \cdot m}\right) \tag{2.8}$$

We find the angle θ by knowing that θ and θ_1 are complimentary angles, and is found using Equation 2.9.

$$\theta = \theta_1 - \frac{\pi}{2} \tag{2.9}$$

The final step is to determine the actual distance, m_{act}, traveled by the platform. In this step, the pan-tilt unit moves to angle θ and then the laser measures the actual distance to the wall, d_{meas}. Using the parameters in Table 2.3, we calculate the actual distance traveled by the platform. This step is represented by Figure 2.5. The equations used to calculate m_{act} are shown below.

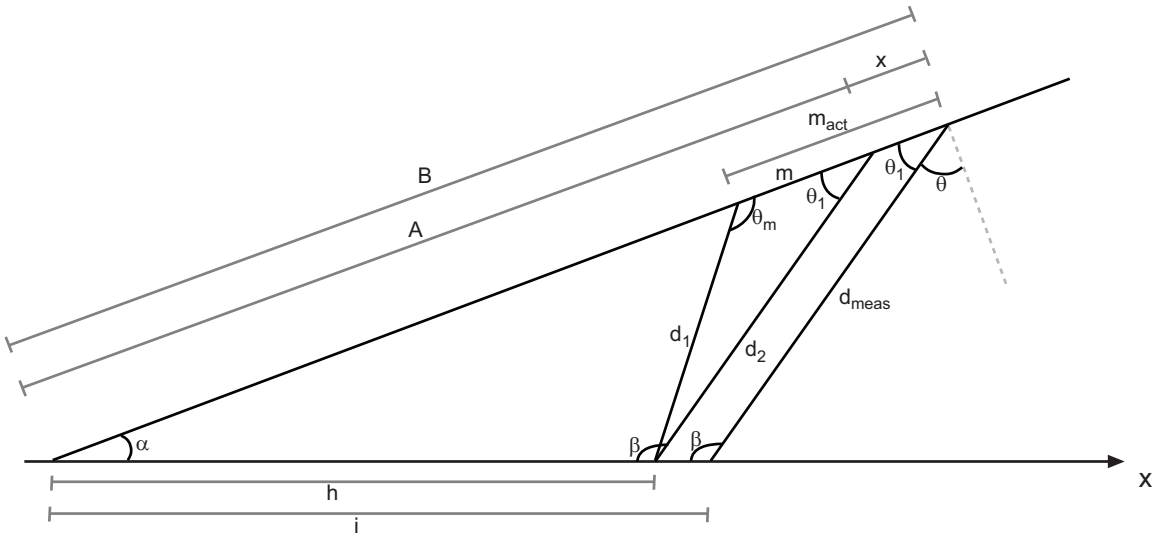

Figure 2.5 1D - Determining the actual distance the platform traveled

d_2	expected distance to wall	From previous step
m	distance moved	Provided by user
θ	angle to turn pan-tilt	From previous step
θ_1	complimentary angle to θ	From previous step
α	angle of motion	From previous step
d_{meas}	actual distance to wall	Measured by laser

Table 2.3 Step 2 calculation variables

Using the summation of angles property of triangles, the angle β is found based on angles α and

θ_1 as shown in Equation 2.10.

$$\beta = \pi - (\alpha + \theta_1) \tag{2.10}$$

Next, the distance along the wall, h, from the intersection of the wall and the direction of motion line to the original laser point is found. This is calculated by using the law of sines as shown in Equation 2.11.

$$h = \frac{d_2 \sin\theta_1}{\sin\alpha} \tag{2.11}$$

The corresponding distance along the direction of motion, A, is calculated based on the calculation of h by using the law of cosines as shown in Equation 2.12.

$$A = \sqrt{h^2 + {d_2}^2 - 2 \cdot h \cdot d_2 \cdot \cos\beta} \tag{2.12}$$

The corresponding distances i and B are found in the same manner as h and A. The difference is that the distances correspond to where the line d_{meas} intersects the two lines instead of the line d_2 as shown in Figure 2.4. The distances i and B are calculated using Equations 2.13 and 2.14.

$$i = \frac{d_{meas} \cdot \sin\theta_1}{\sin\alpha} \tag{2.13}$$

$$B = \sqrt{i^2 + {d_{meas}}^2 - 2 \cdot i \cdot d_{meas} \cdot \cos\beta} \tag{2.14}$$

The distance x is the difference between A and B along the direction of motion. This represents the difference in the distance the platform supposedly moved and the distance the platform actually moved. This is shown by Equation 2.15.

$$x = B - A \tag{2.15}$$

Finally, the actual distance the platform traveled, m_{act} is found by adding the difference of the two distances, x, to the distance the platform was supposed to move, m. This is shown by Equation 2.16.

$$m_{act} = x + m \tag{2.16}$$

2.3 Two-Dimension Model

2.3.1 Calibration

To prevent confusion, a distinction in spaces needs to be made. There are two two-dimensional spaces in which the robot operates. The real world space will be represented by the triplet (x, y, θ). The robot space, which is internal to the robot and is independent of the real world, will be represented by the triplet (p, q, α). The method by which robot space is converted to the real world and vice versa is shown in Figure 2.6.

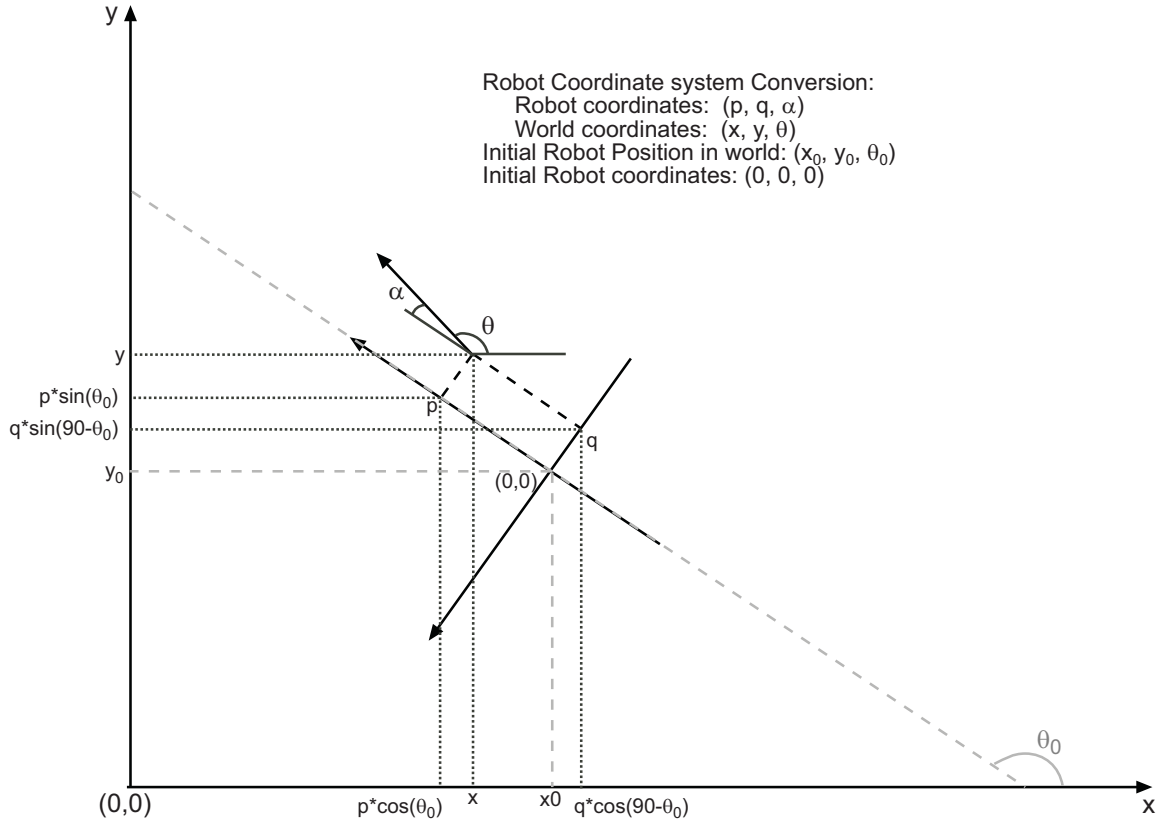

Figure 2.6 Converting between real world space and robot space

To calibrate the robot's space with the real world space, the laser is fired at 2 adjacent walls to find the robot's true position. The walls do not need to be perpendicular; however, it does simplify the computations. In this experiment, the walls are assumed (and measured) to be perpendicular. The way this is done is by moving the laser back and forth on the first wall until it finds the shortest distance between the laser and the wall; this is the perpendicular distance. Then the laser is turned 90 degrees and finds the distance to the other wall. If the walls are not perpendicular to one another, this is where extra computation is needed to extract xy-coordinates. The two perpendicular

distances found are the (x_0, y_0) coordinates for the real world. The pan-tilt unit gives the robot's initial heading. This is due to how everything is physically mounted to each other on the robot. The laser is mounted to the pan-tilt unit in such a way that it points forward at pan position 0 (straight forward). The pan-tilt laser combination is mounted to the robot such that the laser points straight forward on the robot. Based on this, the heading of the robot is the position of the pan-tilt when the laser finds the x-coordinate during calibration. Now that there is a connection between the robot space and the real world, converting from one to the other is possible. The conversion from one space to the other is shown by the equations below.

To convert from (p, q, α) to (x, y, θ), the following equations are used:

$$\theta = \theta_0 + \alpha \tag{2.17}$$

$$x = x_0 + p \cdot \cos\theta_0 - q \cdot \cos(90 - \theta_0) \tag{2.18}$$

$$y = y_0 + p \cdot \sin\theta_0 + q \cdot \sin(90 - \theta_0) \tag{2.19}$$

Similarly, to convert from (x, y, θ) to (p, q, α), the following equations are used:

$$\alpha = \theta - \theta_0 \tag{2.20}$$

$$q = \frac{y - y_0 - x \cdot \tan\theta_0 + \tan\theta_0}{\tan\theta_0 \cdot \cos(90 - \theta_0) + \sin(90 - \theta_0)} \tag{2.21}$$

$$p = \begin{cases} \frac{y - y_0 - q \cdot \sin(90 - \theta_0)}{\sin\theta_0} & (\theta_0 = 90)\, OR\, (\theta_0 = 270) \\ \frac{x - x_0 + q \cdot \cos(90 - \theta_0)}{\cos\theta_0} & otherwise \end{cases} \tag{2.22}$$

2.3.2 Laser Correction

2.3.2.1 Flatness Check and Angle Computation

When the laser takes a distance measurement from the wall, it does not just take a single measurement. Rather, a series of five measurements are taken, with the center being recorded as the distance to the wall for further calculations. Each measurement of the five is separated by 10 positions of the pan-tilt unit. This is equal to about 0.53 degrees (0.00925 radians). There are a couple of reasons for taking a five measurements rather than just one. The first reason is to make sure the area of the wall is flat enough; the other is to compute the angle at which the robot is traveling. Making sure the area is flat is necessary in order to insure the accuracy of the computations. In order to check for flatness, the five distance measurements along with their respective angles are used. We

consider each distance and its angle to be a pair. First the x and y coordinates of each pair are determined. Then the angle and distance (ϕ, r) from each point to the center point is found. Here, we will let (d_3, θ_3) be the center pair. For each of the other points (d_n, θ_n), where n = 1, 2, 4, 5, we determine their respective (ϕ, r) measurements. During the calculations, we check whether the distance from the center point to the n point equals the distance from the n point to the center point. If they are equal, then we continue. If, however, they are not equal, something has been calculated incorrectly. It is not a necessary step, but it acts as a simple check for continuity. The following equations describe how the flatness is determined.

First, we let the xy-coordinates of the center pair (d_3, θ_3) equal h and i as shown in Equations 2.23 and 2.24.

$$h = d_3 \cos \theta_3 \tag{2.23}$$

$$i = d_3 \sin \theta_3 \tag{2.24}$$

Next we determine the xy-coordinates of each pair as shown by Equations 2.25 and 2.26.

$$j = d_n \cos \theta_n \tag{2.25}$$

$$k = d_n \sin \theta_n \tag{2.26}$$

Then we calculate ϕ for each pair with respect to the center pair. This is shown in Equation 2.27.

$$\phi_n = \arctan\left(\frac{h-j}{k-i}\right) \tag{2.27}$$

We finally calculate the r distance from the n^{th} pair to the center pair as well as from the center pair to the n^{th} pair as shown in Equations 2.28 and 2.29. These two values should be equal if the calculations are performed correctly.

$$r = d_3 * \cos\left(\theta_3 - \phi_n\right) \tag{2.28}$$

$$r_n = d_n * \cos\left(\theta_n - \phi_n\right) \tag{2.29}$$

After we have all the rs and ϕs calculated, we then find the maximum and minimum of each set. We find the max and min of the set of ϕs using Equations 2.30 and 2.31.

$$\phi_{max} = max(\phi_1, \phi_2, \phi_4, \phi_5) \tag{2.30}$$

$$\phi_{min} = min(\phi_1, \phi_2, \phi_4, \phi_5) \tag{2.31}$$

Similarly, we find the max and min of the set of rs using Equations 2.32 and 2.33.

$$r_{max} = max(r_1, r_2, r_4, r_5) \tag{2.32}$$

$$r_{min} = min(r_1, r_2, r_4, r_5) \tag{2.33}$$

Once we have the maximums and minimums, then we check to see if the area is flat enough. The area is flat enough if the difference between the maximum and minimum ϕ is less than some ϕ-threshold and if the difference between the maximum and minimum r is less than some r-threshold. These threshold values are found offline using a trial and error method on known flat and non-flat surfaces. This flatness check is shown in Equation 2.34, where 0 refers to a non-flat surface and 1 refers to a flat surface.

$$f = \begin{cases} 0 & (\phi_{max} - \phi_{min}) > \phi_{threshold}\, OR\, (r_{max} - r_{min}) > r_{threshold} \\ 1 & (\phi_{max} - \phi_{min}) < \phi_{threshold}\, AND\, (r_{max} - r_{min}) < r_{threshold} \end{cases} \tag{2.34}$$

If the area is flat enough, the next step is to compute the angle of the robot with respect to the wall, Θ. This process not only finds the angle of the robot's heading, but it also determines how the robot is moving, such as toward or away from the wall. This is shown in Figure 2.7. The equations to find the wall angle and heading are shown below as well. Table 2.5 describes what each choice means in terms of the direction the robot is moving. A summary of known parameters are given in Table 2.4.

θ_1	angle to first point	From flatness check
θ_5	angle to fifth point	From flatness check
d_1	distance to first point	From flatness check
d_5	distance to fifth point	From flatness check
θ_3	angle to third (center) point	From flatness check

Table 2.4 Robot heading calculation variables

Choice (c)	**Robot Heading**
1	Moving toward wall with wall to the left
2	Moving away from wall with wall to the right
3	Moving away from wall with wall to the left
4	Moving toward wall with wall to the right

Table 2.5 The four headings a robot can take with respect to the wall

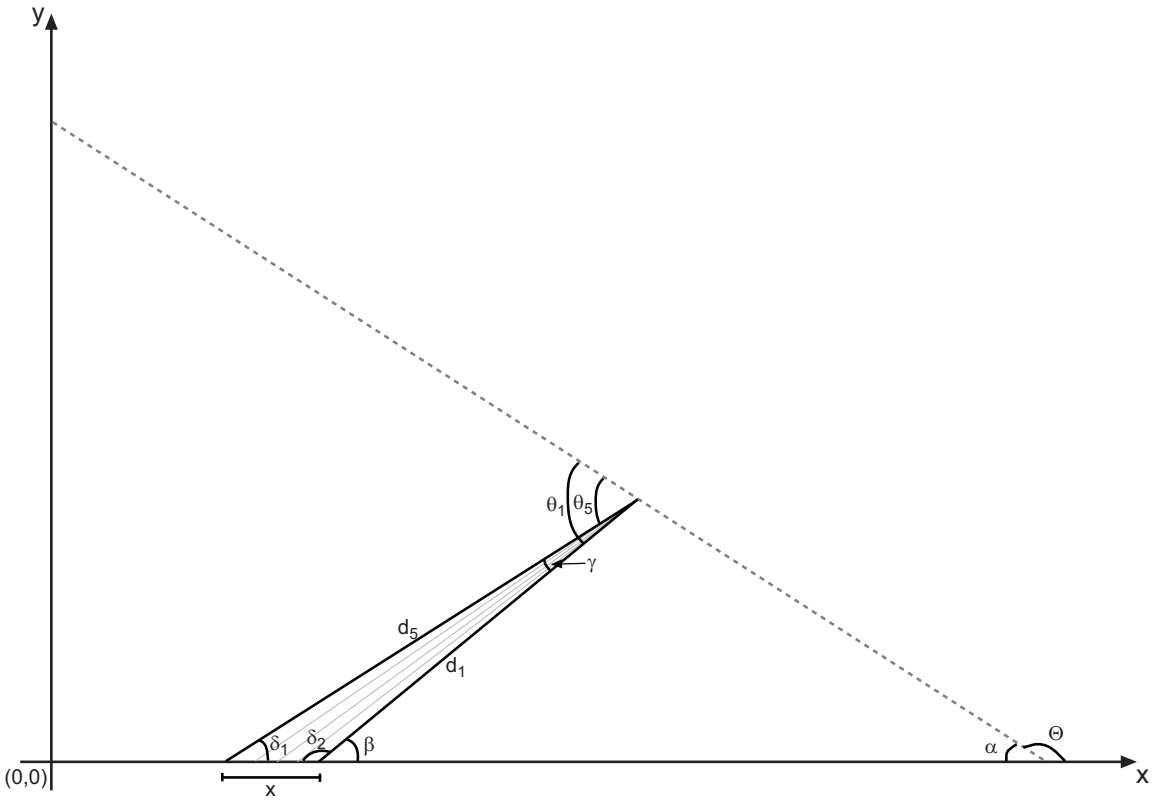

Figure 2.7 Computing the wall angle and heading of the robot

The angle γ is the angle between the first angle and the fifth angle and is found using Equation 2.35.

$$\gamma = |\theta_1 - \theta_5| \tag{2.35}$$

The distance x is the distance along the wall between the first laser point and the fifth laser point. This distance is calculated using the law of cosines as shown in Equation 2.36.

$$x = \sqrt{{d_1}^2 + {d_5}^2 - 2 \cdot d_1 \cdot d_5 \cdot \cos\gamma} \tag{2.36}$$

Next, the inner angle δ_2 is calculated using Equation 2.38. The intermediate step of Equation 2.37 is needed only if the first condition of Equation 2.38 is met. This condition depends on how the robot and laser are oriented with the wall.

$$\delta_1 = \arcsin\Big(\frac{d_1 \sin\gamma}{x}\Big) \tag{2.37}$$

$$\delta_2 = \begin{cases} \pi - (\gamma + \delta_1) & d_5 > d_1 \\ \arcsin\left(\frac{d_5 \sin\gamma}{x}\right) & d_5 < d_1 \end{cases} \tag{2.38}$$

The angle β is the supplementary angle to δ_2 and is found using Equation 2.39.

$$\beta = \pi - \delta_2 \tag{2.39}$$

The calculation of angle α depends on if the laser was turned left or right to point to the wall. If it was turned to the right, then θ_3 is positive, and α is calculated using the first condition of Equation 2.40. If, however, it was turned to the left, then θ_3 is negative, and α is calculated using the second condition of Equation 2.40.

$$\alpha = \begin{cases} \pi - (\beta + \theta_1) & \theta_3 > 0 \\ \pi - (\beta + (\pi - |\theta_1|)) & \theta_3 < 0 \end{cases} \tag{2.40}$$

Also determined by how the laser is turned, the choice c is 1 if θ_3 is positive, and 2 if θ_3 is negative. This is shown in Equation 2.41.

$$c = \begin{cases} 1 & \theta_3 > 0 \\ 2 & \theta_3 < 0 \end{cases} \tag{2.41}$$

If, however, α is less than zero, then the robot is moving in one of the other two directions, and α needs to be recalculated based on the other two directions the robot can move. Recalculating α and determining the actual motion of the robot is shown in Equations 2.42 - 2.46.

First, we recalculate the inner angle δ_2 using Equation 2.43 based on different assumptions on how the robot and laser are oriented. Once again, we only use the intermediate step of Equation 2.42 if the first condition of Equation 2.43 is met.

$$\delta_1 = \arcsin\left(\frac{d_5 \sin\gamma}{x}\right) \tag{2.42}$$

$$\delta_2 = \begin{cases} \pi - (\gamma + \delta_1) & d_1 > d_5 \\ \arcsin\left(\frac{d_1 \sin\gamma}{x}\right) & d_1 < d_5 \end{cases} \tag{2.43}$$

The angle β is still the supplementary angle to δ_2 and is found using Equation 2.44.

$$\beta = \pi - \delta_2 \tag{2.44}$$

Finally, the angle α is recalculated and the choice c is determined based on if laser is turned left or right to point toward the wall. This is shown by Equations 2.45 and 2.46.

$$\alpha = \begin{cases} \pi - (\beta + (\pi - \theta_5)) & \theta_3 > 0 \\ \pi - (\beta + |\theta_5|) & \theta_3 < 0 \end{cases} \tag{2.45}$$

$$c = \begin{cases} 3 & \theta_3 > 0 \\ 4 & \theta_3 < 0 \end{cases} \tag{2.46}$$

After the angle α is calculated, whether it be from Equation 2.40 or from Equation 2.45, the angle Θ is calculated by knowing that it is the supplementary angle to α. This is shown in Equation 2.47.

$$\Theta = \pi - \alpha \tag{2.47}$$

2.3.2.2 Step 1

After calibration, the laser is turned at an arbitrary angle A such that the laser is pointing somewhere on one of the walls. After the pan-tilt unit stops, the distance to the wall is measured by the laser. Since this is the first point of the robot's movement, the xy-coordinates have already been obtained by the calibration process. The only new information is the angle the pan-tilt unit turned (A) and the distance the laser measured (d_0). This is shown in Figure 2.8.

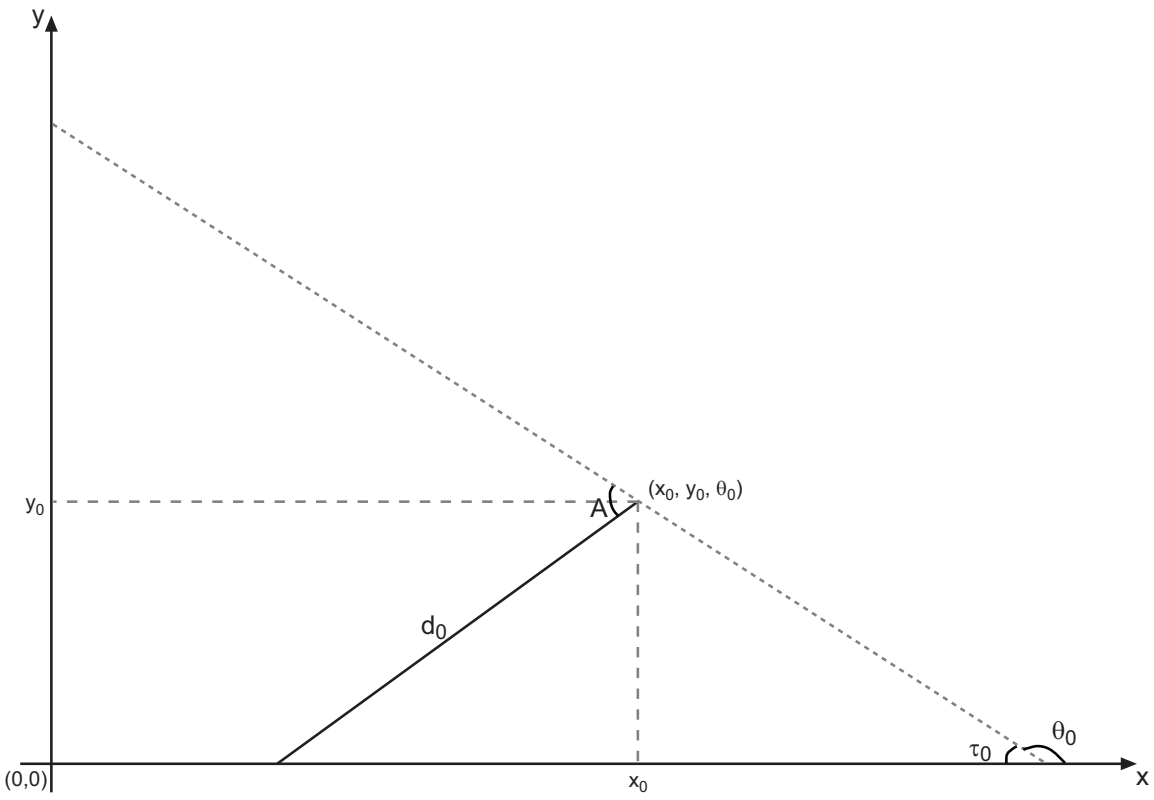

Figure 2.8 Finding first distance measurement

2.3.2.3 Step 2

Now that everything has been initialized, the robot is finally ready to move. The robot moves either forward or back some distance. When the robot stops, the odometry is recorded and converted from robot coordinates to world coordinates using the conversions from above. Using this information, we compute the angle, B, at which the pan-tilt should be turned such that the laser points back to the original point on the wall. We also compute the expected distance, d_1, from the laser to that point on the wall. This process is shown in Figure 2.9. The list of known parameters is given in Table 2.6, and the equations used to compute B and d_1 are given below.

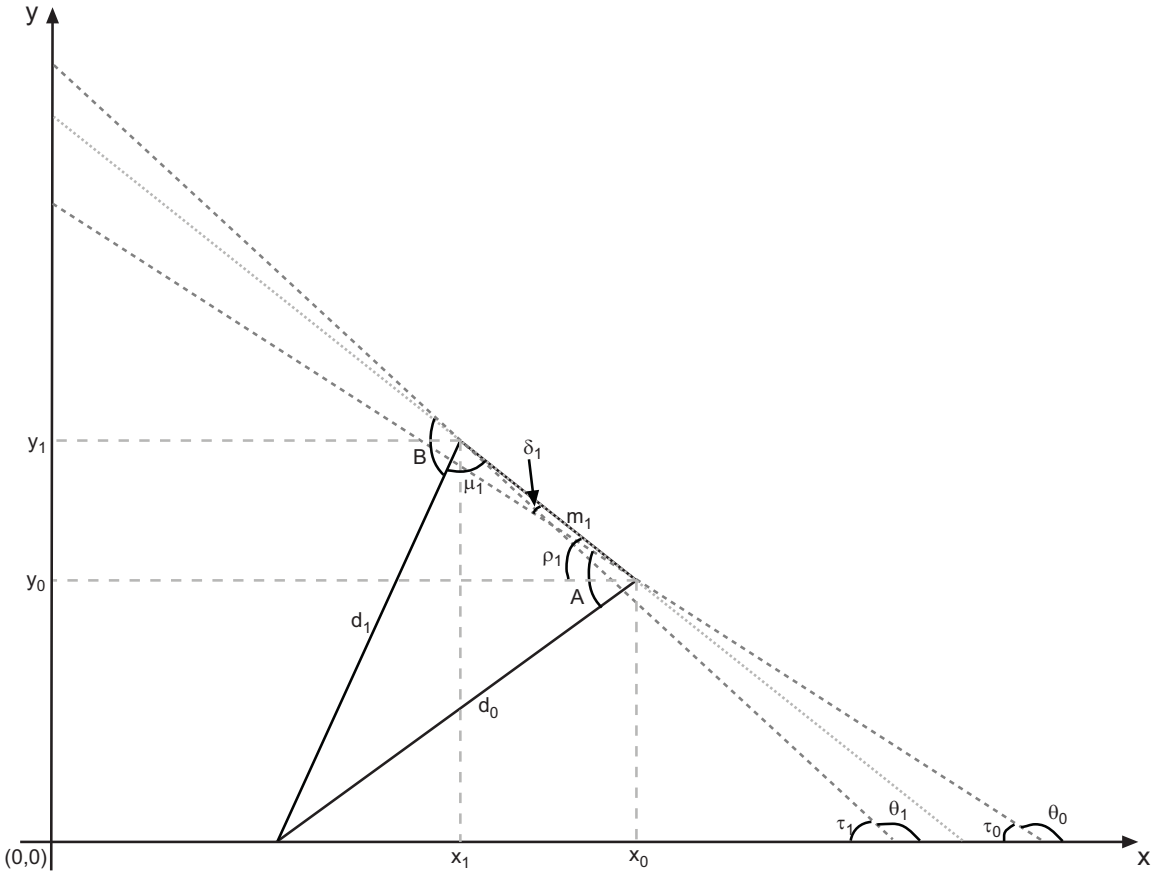

Figure 2.9 Finding angle and distance based on odometry for forward motion

The angle τ_1 is the supplementary angle to θ_1 and is found using Equation 2.48.

$$\tau_1 = \pi - \theta_1 \tag{2.48}$$

We then find the distance, m_1, between the previous point and the current odometry point using

(x_0, y_0)	previous location coordinates	From previous step
(x_1, y_1)	odometry location coordinates	Measured by odometer
A	pan-tilt angle for previous distance	From previous step
d_0	previous distance to wall	From previous step
θ_0	previous angle of motion	From previous step
θ_1	odometry angle of motion	Measured by odometer

Table 2.6 Step 2 calculation variables

Equation 2.49.

$$m_1 = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2} \tag{2.49}$$

If the robot moves in the reverse direction, which is not represented by Figure 2.9, we use Equations 2.50 - 2.54 to solve for B and d_1. First, we find the angle ρ_1, which is the angle between the previous point and the current odometry point. This is shown in Equation 2.50.

$$\rho_1 = \arcsin\left(\frac{y_0 - y_1}{m_1}\right) \tag{2.50}$$

We then calculate the angle, δ_1, which is the difference between τ_1 and ρ_1. We find this by using Equation 2.51. This represents the angle difference between the previous point and the current point.

$$\delta_1 = \tau_1 - \rho_1 \tag{2.51}$$

The inner angle, μ_1, between m_1 and the expected distance d_1, is computed using Equation 2.52.

$$\mu_1 = \pi - A + ((\pi - \theta_0) - \rho_1) \tag{2.52}$$

We then compute the expected distance d_1 using the law of cosines as shown in Equation 2.53.

$$d_1 = \sqrt{{d_0}^2 + {m_1}^2 - 2 \cdot d_0 \cdot m_1 \cdot \cos\mu_1} \tag{2.53}$$

Finally, we calculate the angle B using the distance we just found by once again using the law of cosines as shown in Equation 2.54.

$$B = \arccos\left(\frac{{d_1}^2 + {m_1}^2 - {d_0}^2}{2 \cdot d_1 \cdot m_1} + \delta_1\right) \tag{2.54}$$

If, however, the robot moves forward, which is represented in Figure 2.9, we use Equations 2.55 - 2.59 to solve for the pan-tilt angle B and the expected distance d_1.

First, we compute angle, ρ_1, between the previous point and the current point using Equation 2.55.

$$\rho_1 = \arcsin\left(\frac{y_1 - y_0}{m_1}\right) \tag{2.55}$$

We then compute the difference between the previous angle of motion and ρ_1. This is represented by δ_1 and is found using Equation 2.56.

$$\delta_1 = (\pi - \theta_0) - \rho_1 \tag{2.56}$$

We then compute the expected distance d_1 using the law of cosines as shown in Equation 2.57.

$$d_1 = \sqrt{{d_0}^2 + {m_1}^2 - 2 \cdot d_0 \cdot m_1 \cdot \cos(A - \delta_1)} \tag{2.57}$$

By first finding the inner angle μ_1 using Equation 2.58, we can then compute angle B as shown in Equation 2.59.

$$\mu_1 = \arccos\left(\frac{{d_1}^2 + {m_1}^2 - {d_0}^2}{2 \cdot d_1 \cdot m_1}\right) \tag{2.58}$$

$$B = (\pi - \mu_1) + (\tau_1 - \rho 1) \tag{2.59}$$

After the pan-tilt angle and expected distance (B, d_1) are calculated, the pan-tilt unit is then moved to angle B. At this position, the laser is then used to find the actual distance between the laser and the wall. As it does this, it also finds the angle at which it says the robot is moving with respect to the wall. Using this information, the position, (x_2, y_2), where the laser says the robot is located is found. This method is shown by Figure 2.10 and is outlined by the equations below. A summary of known parameters is given in Table 2.7. For this step, we introduce the parameter R. We let R equal 0 if the robot moves forward, and 1 if the robot moves in the reverse direction.

d_2	actual distance	Measured by laser
θ_2	actual wall angle	Measured by laser
R	[0,1]	Reverse parameter
A	previous angle of pan-tilt	From previous steps
B	current angle of pan-tilt	From previous step
(x_0, y_0)	previous point coordinates	From previous steps
(x_1, y_1)	odometry point coordinates	From previous step
θ_0	previous angle of motion	From previous steps
θ_1	odometry angle of motion	From previous step

Table 2.7 Step 2 calculation variables

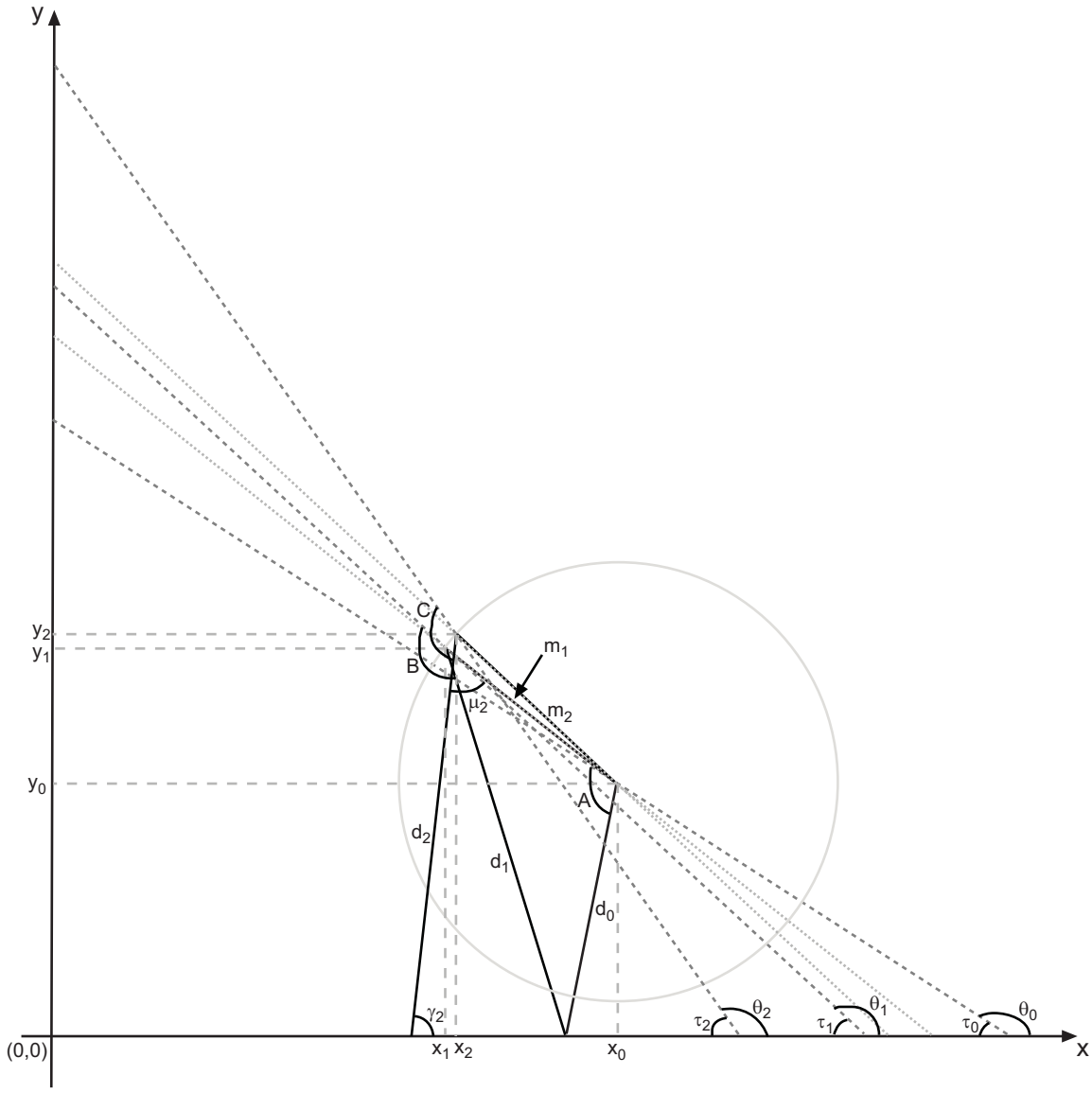

Figure 2.10 Finding xy-coordinates of actual location of robot using laser based on odometry

We first let angle C be the angle we actually turn the pan-tilt unit, which is equal to the previously calculated angle B, as shown in Equation 2.60.

$$C = B \tag{2.60}$$

We then find two supplementary angles to angles we now know. First, the angle τ_2 is supplementary to the angle of motion θ_2 and is found using Equation 2.61. Second, the angle μ_2 is supplementary to angle C and is found using Equation 2.62.

$$\tau_2 = \pi - \theta_2 \tag{2.61}$$

$$\mu_2 = \pi - C \tag{2.62}$$

Using the summation of angles property for triangles, we can calculate the angle γ_2, as shown by Equation 2.63.

$$\gamma_2 = \pi - (\mu_2 + \tau_2) \tag{2.63}$$

We can now find the first coordinate (y_2) of our corrected position by using the sine law as shown by Equation 2.64.

$$y_2 = d_2 \sin \gamma_2 \tag{2.64}$$

The only way to associate the laser corrected information to existing information is to make an assumption based on the odometry information. We either assume that the angle the odometer returns is correct, or assume that the overall distance the robot traveled based on odometry coordinates is correct. If the first assumption is correct, then the odometry angle and the converted wall angle found by the laser will be equal. If this is the case, then the following equations are used to find the x-coordinate. It needs to be noted that this method is not depicted in Figure 2.10.

Using the tangent, we find the distances on the x-axis based on y_2 from Equation 2.65 and on y_1 from Equation 2.66. We also know that by making this assumption, τ_1 equals τ_2.

$$x_{12} = \frac{y_2}{\tan \tau_2} \tag{2.65}$$

$$x_{11} = \frac{y_1}{\tan \tau_1} \tag{2.66}$$

We now calculate the difference in x-distance based on whether the robot traveled forward or backward, as shown in Equation 2.67.

$$x_{22} = \begin{cases} x_{12} - x_{11} & R = 0 \\ x_{11} - x_{12} & R = 1 \end{cases} \tag{2.67}$$

Once the difference in x-distance between the odometry position and the laser-corrected position is known, we can determine the x-coordinate (x_2) of the laser-corrected position by using Equation 2.68.

$$x_2 = x_1 - x_{22} \tag{2.68}$$

If the first assumption is not correct, then we must assume that the overall distance the robot traveled was correctly computed based on the odometry information. If we assume this, then the actual position of the robot lies somewhere on a circle of radius m_1, centered at the previous position of the robot (x_0, y_0). Using the information we've already gathered, the actual position on the circle is where the calculated y-coordinate intersects the circle. It intersects in two locations, one in front of the robot in the x-direction and one behind the robot in the x-direction. If the robot is moving forward, the point in front of the robot is used. If, however, the robot is moving in reverse, then the point behind the robot is used. The equations that calculate the x-coordinate are given here.

If we make this assumption, then we assume that the distance from the previous point to the laser-corrected point is equal to that from the previous point to the odometry point. This is shown by Equation 2.69.

$$m_2 = m_1 \tag{2.69}$$

By using the Pythagorean theorem, we can find the x-coordinate (x_2) by using Equation 2.70. Which condition we use is determined by the motion of the robot, whether it moved forward or reverse.

$$x_2 = \begin{cases} x_0 - \sqrt{{m_2}^2 - (y_0 - y_2)^2} & R = 0 \\ x_0 + \sqrt{{m_2}^2 - (y_0 - y_2)^2} & R = 1 \end{cases} \tag{2.70}$$

The method we present here continually moves the robot then repeats step 2. For our experiment, we moved the robot 49 times so that we have a total of 50 data points counting the starting point. The results from our experiment are given later.

CHAPTER 3

EXPERIMENTAL RESULTS

In this chapter, we describe two experiments we performed to test our methods. The first experiment tests the one-dimensional model. This experiment was performed to test the efficacy of our methods, meaning it was done to test whether the basic geometry was correct. For this test, we placed the laser on a pan-tilt unit, and mounted it on a sliding platform. The platform was moved in a straight line along a rail, and the laser was used to determine how far the platform moved. The second experiment tests the two-dimensional model. For this experiment, we mounted the pan-tilt laser to a mobile robot. We allowed the robot to move freely in a flat two-dimensional environment, and we used the laser to correct the odometry of the robot. We also used the laser to acquire ground truth position measurements, which we explain in more detail later. These measurements are used to objectively evaluate the performance of our laser-based odometry corrections.

3.1 Testing apparatus

3.1.1 1D Model

The equipment we use for this part of the experiment is a flat sliding platform, a rail, a pan-tilt unit, and a range-finding laser. The completely assembled platform is shown in Figure 3.1. The platform is a square of aluminum plate measuring 30.5 cm x 30.5 cm with two holes drilled in it to mount the pan-tilt unit. Small felt pads have been adhered to the bottom side of the plate to allow it to slide smoothly along the rail. The rail was a 1m piece of aluminum angle with cm markings on one side to show how far the platform slid. The markings ranged from 0cm up to 40cm. The pan-tilt unit is the Computer Controlled Pan-Tilt Unit (Model PTU-D46-17.5) of DirectedPerception. It operates via RS-232 using ASCII commands. It has a load capacity of 2.72 kg, speeds up to 300 degrees/second, and a resolution of 3.086 arc minute (0.0514 degrees). The laser is the MiniLASER from HiTECH Technologies, Inc. Like the pan-tilt unit, it operates via RS-232 using ASCII commands. It can measure distances accurately from 0.1m to 100m with natural surfaces. Its accuracy is ±2mm under defined conditions. This refers to using a planar non-glossy target operating at 20 °C. The accuracy under undefined conditions is from ±3mm to ±5mm. Its resolution is 0.1mm depending on the target and atmospheric conditions. Its measuring time is between 0.16s to 6s.

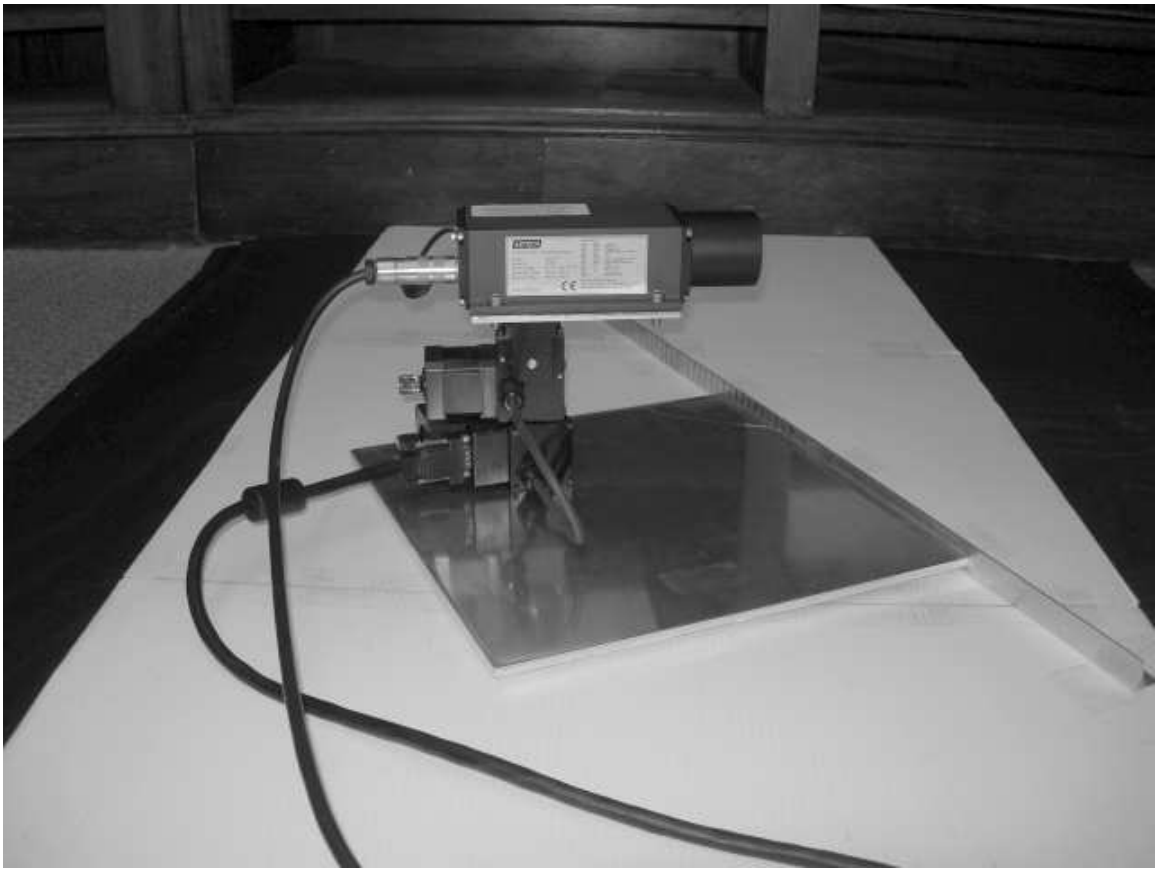

Figure 3.1 One-Dimensional model of pan-tilt unit and range-finding laser

3.1.2 2D Model

The two-dimensional model replaces the sliding platform with a mobile robot. The mobile robot is an ActivMedia Inc Pioneer 2 Mobile Robot. It operates via RS-232 communication using the Saphira client-development software. The robot along with the pan-tilt unit and range-finding laser are shown mounted together as they were being used in Figure 3.2.

Figure 3.2 Mobile robot with pan-tilt unit and range-finding laser

3.2 Results

3.2.1 1D Model

Our one-dimensional experiment was strictly an efficacy test. We only wanted to know if our theory held true. Therefore, our testing was not elaborate. For our test, we slid the platform along the rail a certain distance. We then input a distance to the program, which may or may not be the match the distance we moved the platform. The program then would compute the actual distance

the platform moved. Table 3.1 shows our results from our test. The results show that our theory is correct. We could not extract any meaningful statistics from this data because there is no odometry information. We were physically moving the platform by hand along a rail that had hand drawn markings on it. This experiment was just to test to see if our theory worked before we moved on to the two-dimensional experiment. However, if we wanted to test this 1D model further, we could attach the platform to a motor that has an encoder to move the platform forward and back to get actual odometry measurements from it.

Input move amt (cm)	**Physical move amt (cm)**	**Actual move amt (cm)**
10.0	10	10.166
10.0	5	4.653
-10.0	-5	-4.652
20.0	20	21.050
-20.0	-15	-15.395
25.0	5	5.811
-10.0	-1	-0.588
1.0	10	10.178
10.0	-5	-5.012
20.0	3	3.626

Table 3.1 Listing of 1D Test Results

3.2.2 2D Model

Our two-dimensional experiment took place in the corner of our lab. The floor was carpeted and each of the two walls was a different type of surface. One wall had a plastic corrugated board taped to the wall, the other wall was just the wall surface. The two walls were only needed to get an actual position of the robot for comparison purposes. All the calculations were based on the measurements obtained from the plastic board covered wall. For the test, we programmed the robot to move forward and back 49 times so that we would have a total of 50 positions counting the initial position. This process was repeated 10 times. For 5 of these, the first movement of the robot was forward. For the other 5, the first movement was backward.

The data returned as three sets of (x,y,θ) points for each step of the robot. The first set was the odometry position of the robot. The second set was the laser-corrected position, and the third set was the actual position. Four statistics were drawn from each step. The first was the linear distance from odometry (x,y) point to the actual (x,y) point. The second was the linear distance from the laser-corrected (x,y) point to the actual (x,y) point. The third was the difference between the actual

heading and the odometry heading. Finally, the fourth was the difference between the odometry heading and the laser-corrected heading. These four statistics were found for each of the 50 steps of the tests. The average of the 10 tests for each of these four statistics were taken along with the standard deviation. The two average distance stats were plotted against each other as shown in Figure 3.3. Likewise, the two average angle stats were plotted against each other in Figure 3.4.

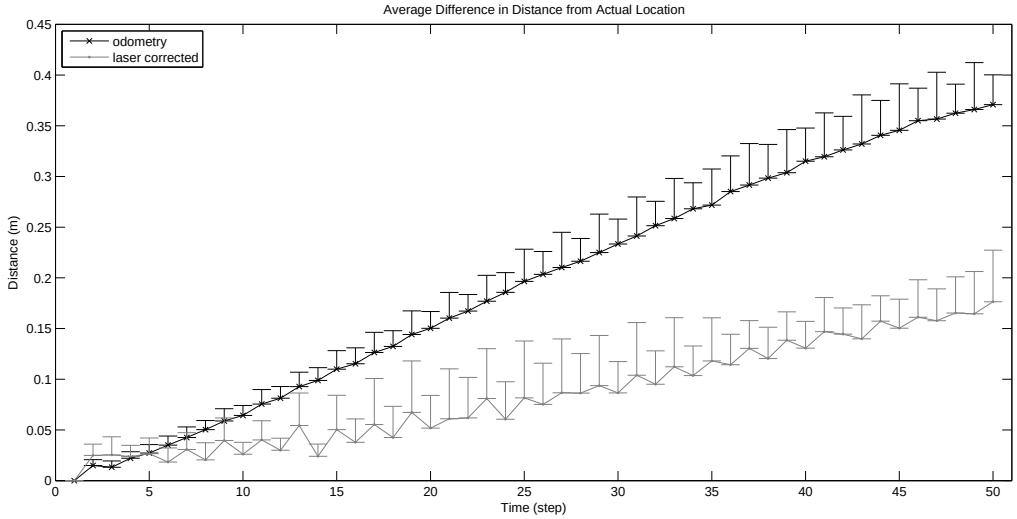

Figure 3.3 Average Difference in Distance from Actual Position

Figure 3.3 shows the accumulation of error by the odometer represented by the top line of the plot. The lower line represents the error of the laser-correction algorithm. It shows that while the laser does not eliminate the error, it does improve the odometry. In actuality, it cuts the error in half.

Figure 3.4 shows the same accumulation of error by the odometer. The laser-correction, however, is not as predictable as in the Figure 3.3. The benefit of the laser-corrected angle that can be drawn from this plot is that it does stay around zero.

3.2.2.1 2D Model - Actual position

In order for us to be able to analyze our two-dimensional model, we had to develop a technique to measure the actual position of the robot in the real world. We were able to do this by using two perpendicular walls. Then, the perpendicular distance from each wall to the robot was found

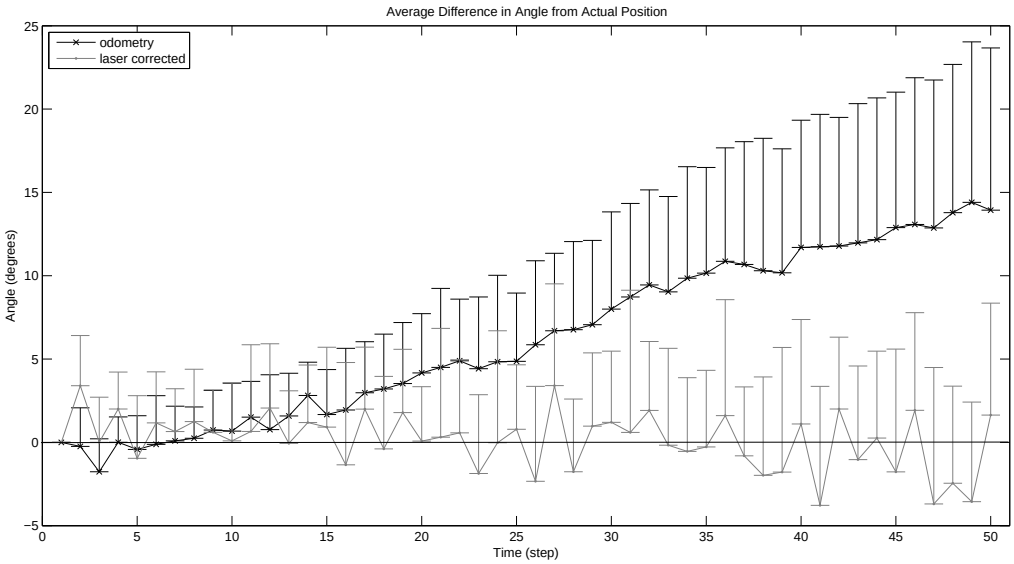

Figure 3.4 Average Difference in Angle from Actual Position

using the laser. We used the same technique as in the calibration step to find these distances, which became the xy-coordinates in the real world. Since we needed two walls in order to find the ground truth position of the robot, we were limited in our selection of operable spaces. It was for this reason that we only tested the model in one location.

There were ideas of running the robot around the room or up and down the hallway, but neither of these options allowed for accurate ground-truth measurements to be obtained. Without these measurements, there would not be a way to compare the odometry location with the laser-corrected location, which would render those tests inconclusive.

3.2.3 Surface testing

During testing, we found that the laser had different accuracy depending on the surface the laser was targeting. To determine what kind of effect each surface had, we performed straight line tests on each of the surfaces that we were using. We found that the laser performed better for the flatter surfaces. The flatness tests for our surfaces are represented by Figures 3.5, 3.6, 3.7, 3.8, and 3.9. The two surfaces used to obtain the results above were the plastic board shown by Figure 3.8 and the wall surface shown by Figure 3.9. Each of these figures shows the readings from the laser with a best fit line, depicted by the top plot of each figure. The bottom plot of each of these figures is

the residual plot. The residuals are the distances between the data plot and the line. Also depicted on this bottom plot is the norm of the residuals. The lower the norm is, the flatter the surface is.

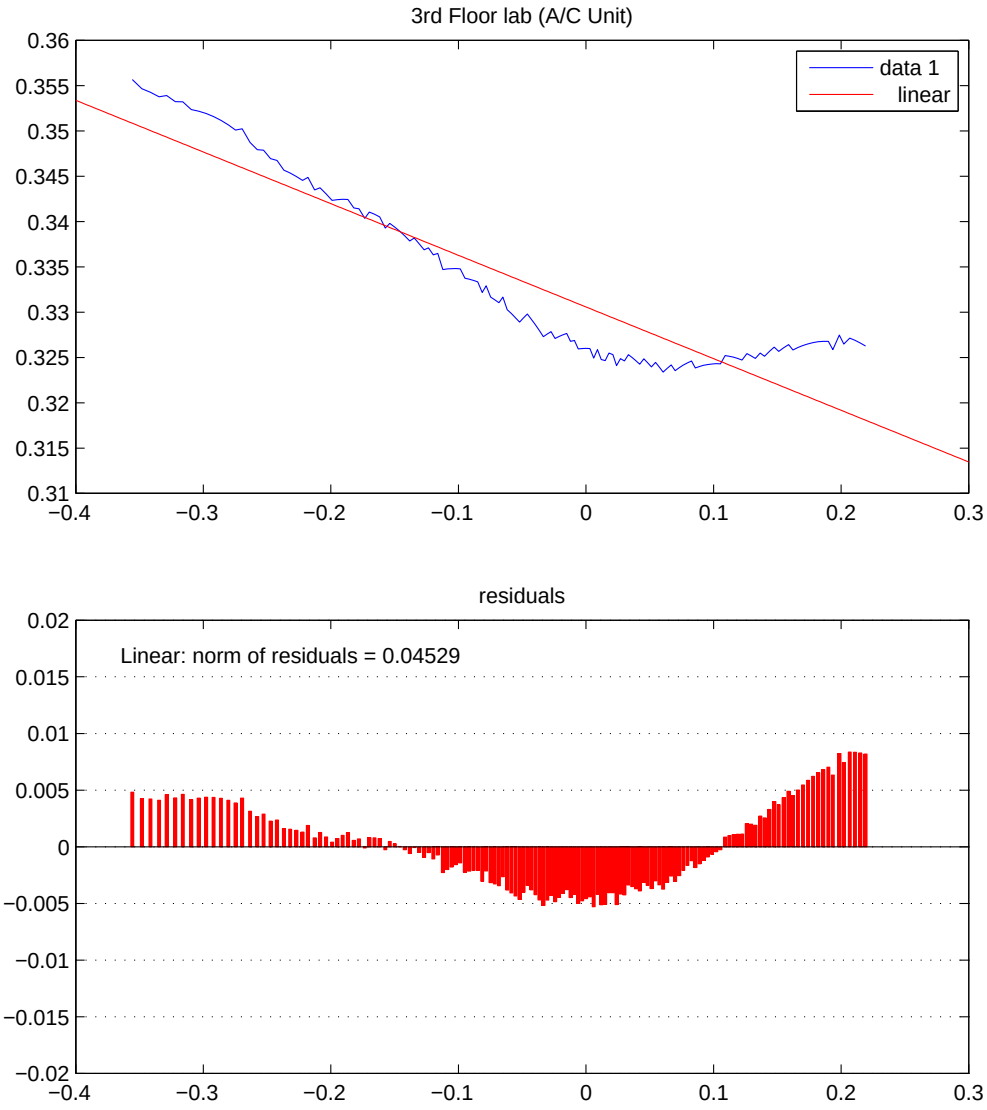

Figure 3.5 3rd Floor lab (A/C Unit)

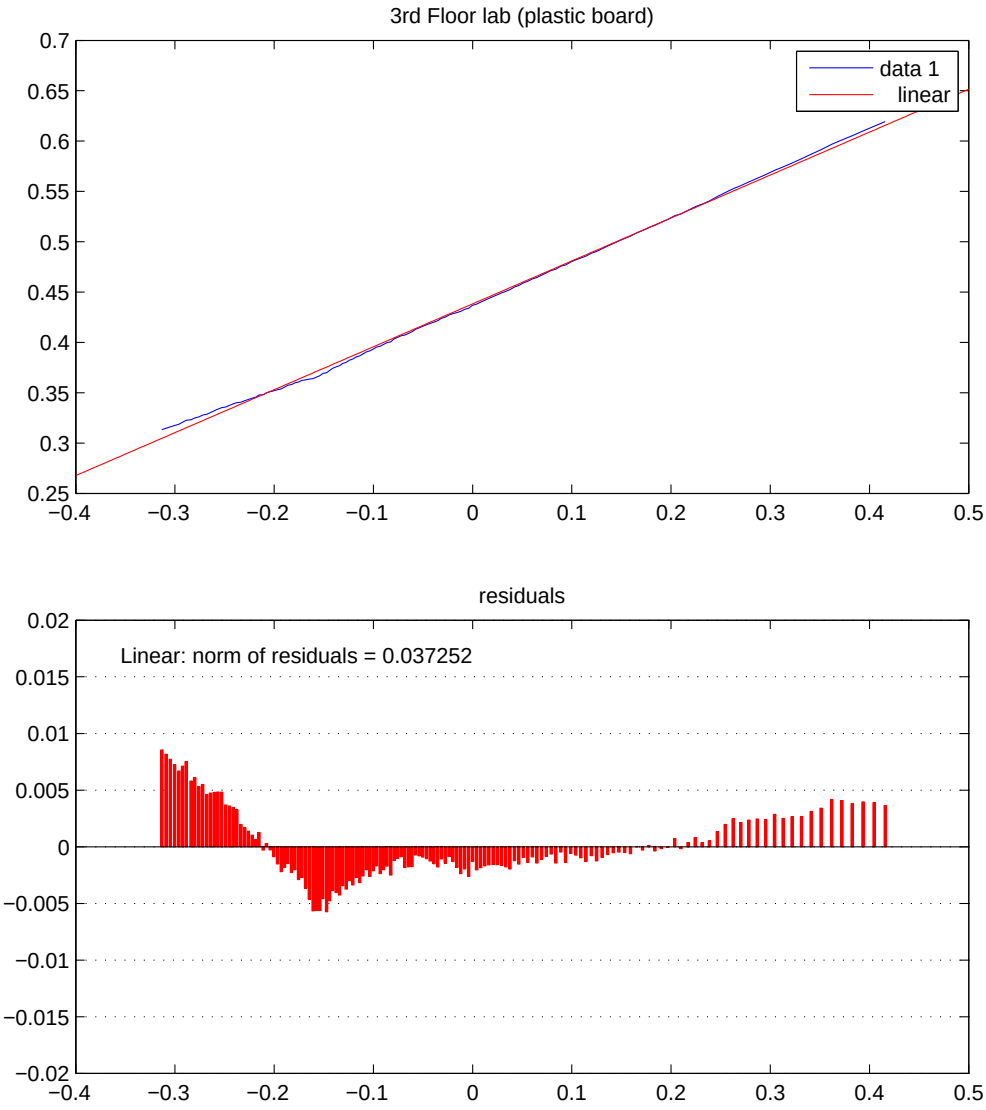

Figure 3.6 3rd Floor lab (plastic board)

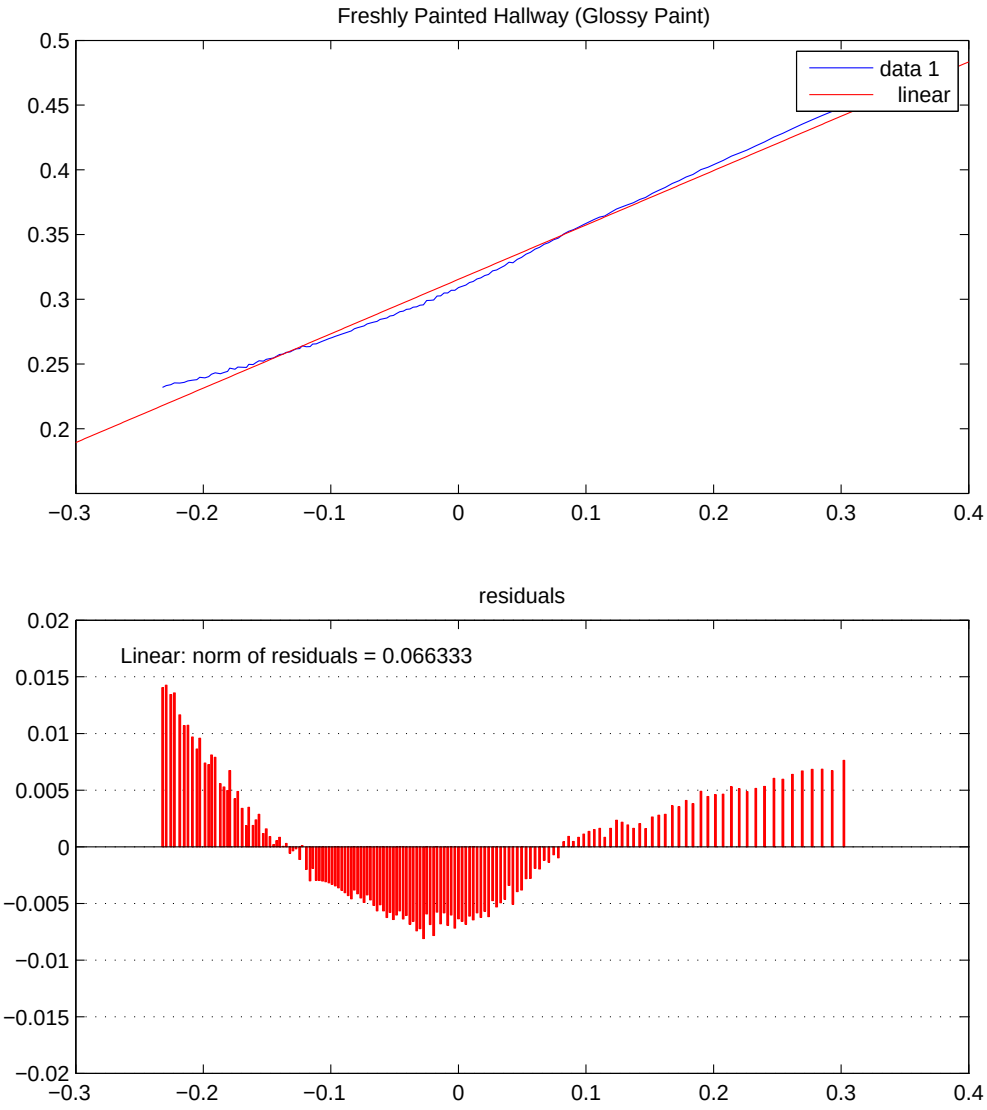

Figure 3.7 Freshly Painted Hallway (Glossy Paint)

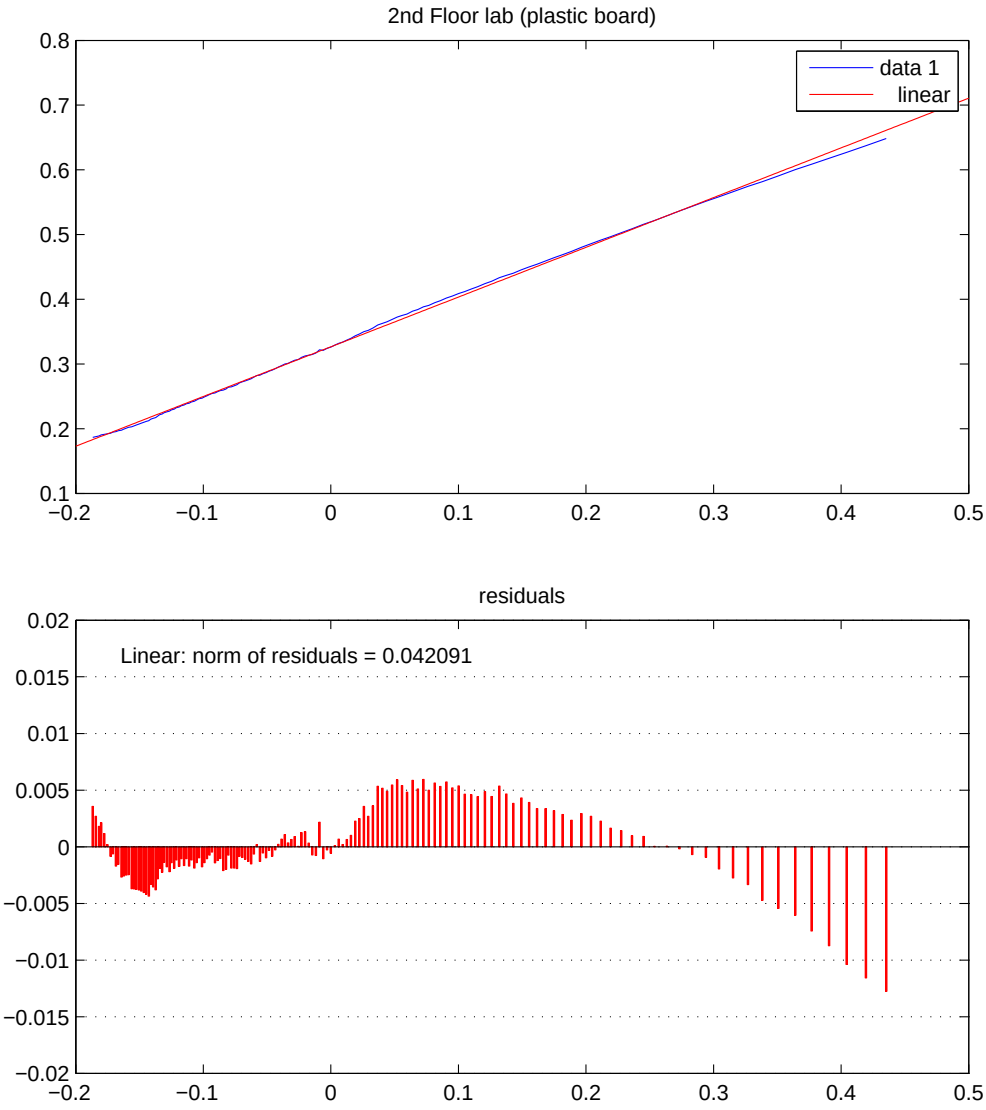

Figure 3.8 2nd Floor lab (plastic board)

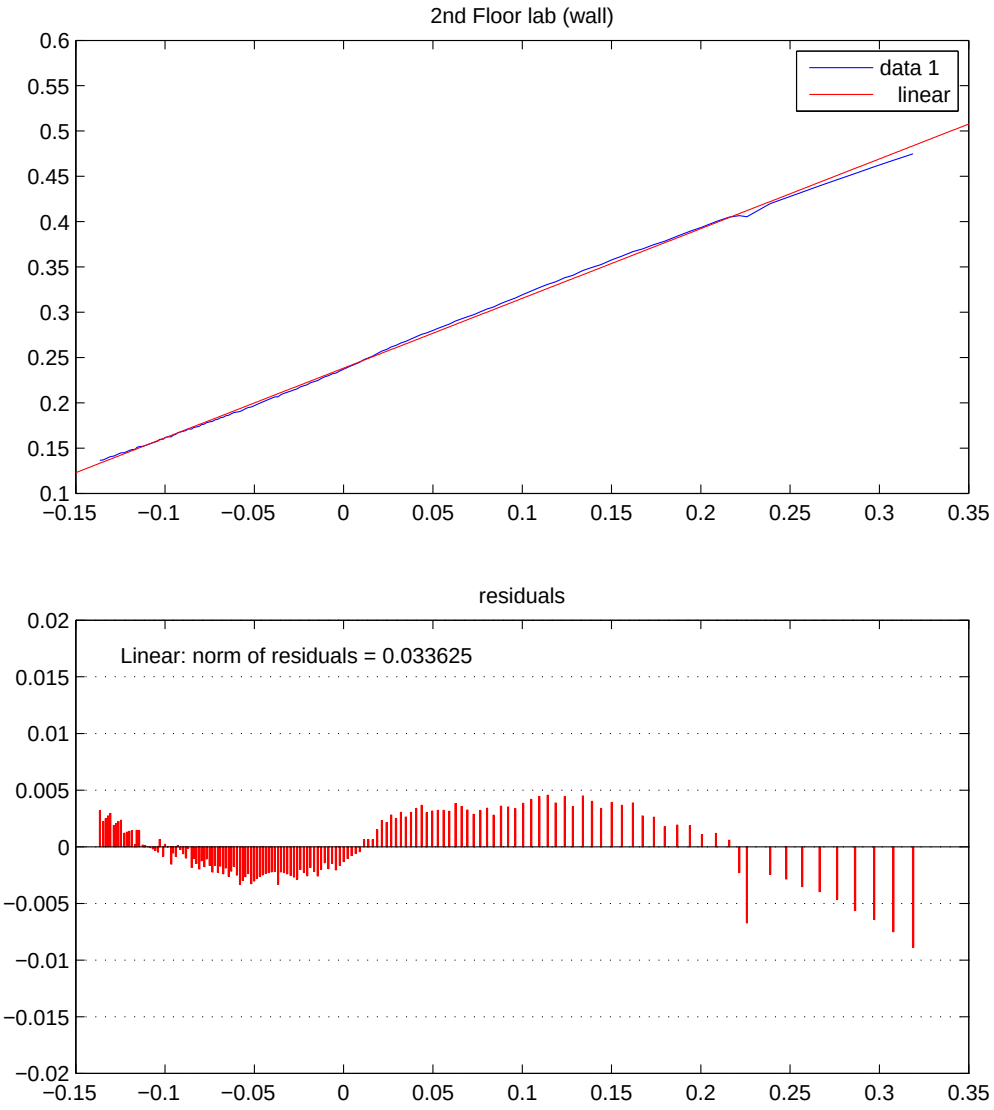

Figure 3.9 2nd Floor lab (wall)

CHAPTER 4

CONCLUSIONS

4.1 Results Discussion

We have shown that a range-finding laser can be used to improve odometry both in the one-dimensional model as well as in the two-dimensional model. While the one-dimensional model was only used as an efficacy test, further experimentation can be performed to show that a laser can be used to correct odometry for one-dimensional movement. In the two-dimensional case, we were not able to completely correct the odometry, but we were able to improve it. Our model showed half the error accumulation that shown by the pure odometry model.

One thing we noticed in our results from each fifty-step test was that there was a definite oscillation in the error. The odometry as well as our laser corrected model showed this behavior. What we saw in the data was that there was more error when the robot moved forward than when it moved in reverse. By taking the average of ten runs, we were able to remove most of the oscillation, but it can still be seen in the laser-correction error line. What we have not been able to determine is if this behavior is limited to this particular robot or the environment.

4.2 Future

The basis for this work came from a science fiction idea of vehicles navigating themselves. A vehicle would be equipped with 10 to 20 of these pan-tilt range-finding lasers, all pointing at different objects. Some would be tracking the ground, while others track the surrounding vehicles and environment. As the vehicle moved, the lasers would constantly be finding points, tracking them, and then as they would be lost, the lasers would lock on to new points. There would then be filters designed to combine all the data from set of lasers to determine the vehicle's location with respect to the world or map, as well as to its surrounding vehicles. The basic priciple for this idea is determining if these pan-tilt lasers can be used to precisely determine location.

REFERENCES

[1] T. Abbas, M. Arif, and W. Ahmed. Measurement and correction of systematic odometry errors caused by kinematics imperfections in mobile robots. In *SICE-ICASE International Joint Conference*, pages 2073–2078, Busan, South Korea, October 2006.

[2] J. Borenstein. Experimental results from internal odometry error correction with the omnimate mobile robot. *Robotics and Automation, IEEE Transactions on*, 14(6):963–969, December 1998.

[3] D. Bouvet and G. Garcia. Guaranteed 3-d mobile robot localization using an odometer, an automatic theodolite and indistinguishable landmarks. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 3612–3617, Seoul, Korea, May 2001.

[4] F. Calabrese and G. Indiveri. An omni-vision triangulation-like approach to mobile robot localization. In *Proceedings of IEEE International Symposium on Intelligent Control*, pages 604–609, Limassol, Cyprus, June 2005.

[5] F. Chenavier and J. Crowley. Position estimation for a mobile robot using vision and odometry. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 3, pages 2588–2593, Nice, France, May 1992.

[6] A. Georgiev and P. Allen. Vision for mobile robot localization in urban environments. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 472–477, Lausanne, Switzerland, October 2002.

[7] A. Howard, M. Mataric, and G. Sukhatme. Localization for mobile robot teams using maximum likelihood estimation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 434–439, Lausanne, Switzerland, October 2002.

[8] T. Koshizen. The gaussian mixture bayes with regularised em algorithm (gmb-rem) for real mobile robot position estimation technique. In *Proceedings of IEEE International Conference of Automation, Robotics, Control and Vision*, volume 1, pages 256–260, Sinigapore, 1998.

[9] T. Koshizen, P. Bartlett, and A. Zelinsky. Sensor fusion of odometry and sonar sensors by the gaussian mixture bayes' technique in mobile robot position estimation. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, volume 4, pages IV–742–IV–747, Tokyo, Japan, October 1999.

[10] S. Maeyama, N. Ishikawa, and S. Yuta. Rule based filtering and fusion of odometry and gyroscope for a fail safe dead reckoning system of a mobile robot. In *Proceedings of IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 541–548, Washington, DC, December 1996.

[11] A. Martinelli. A possible strategy to evaluate the odometry error of a mobile robot. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 1946–1951, Maui, HI, Oct 29–Nov 3 2001.

[12] A. Martinelli. Evaluating the odometry error of a mobile robot. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 853–858, Lausanne, Switzerland, October 2002.

[13] A. Martinelli and R. Siegwart. Estimating the odometry error of a mobile robot during navigation. In *European Conference on Mobile Robots (ECMR 2003)*, Warsaw, Poland, September 2003.

[14] A. Martinelli, N. Tomatis, and R. Siegwart. Simultaneous localization and odometry self calibration for mobile robot. *Autonomous Robots*, 22(1):75–85, January 2007.

[15] Q. Meng and R. Bischoff. Odometry based pose determination and errors measurement for a mobile robot with two steerable drive wheels. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 41(4):263–282, January 2005.

[16] A. Rudolph. Quantification and estimation of differential odometry errors in mobile robotics with redundant sensor information. *International Journal of Robotics Research*, 22(2):117–128, February 2003.

[17] N. Vlassis, G. Papakonstantinou, and P. Tsanakas. Dynamic sensory probabilistic maps for mobile robot localization. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 718–723, Victoria, B.C., Canada, October 1998.

[18] C. Wang. Location estimation and uncertainty analysis for mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1231–1235, April 1988.