Trajectory / Temporal Planning of a Wheeled Mobile Robot

A Thesis Submitted to the

College of Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the Degree of Master of Science

in the Department of Mechanical Engineering

University of Saskatchewan

Canada

By

Imran Waheed

December 2006

PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Mechanical Engineering
University of Saskatchewan
57 Campus Drive
Saskatoon, Saskatchewan, Canada
S7N 5A9

ABSTRACT

In order for a mobile robot to complete its task it must be able to plan and follow a trajectory. Depending on the environment, it may also be necessary to follow a given velocity profile. This is known as temporal planning. Temporal planning can be used to minimize time of motion and to avoid moving obstacles. For example, assuming the mobile robot is an intelligent wheelchair, it must follow a prescribed path (sidewalk, hospital corridor) while following a strict speed limit (slowing down for pedestrians, cars). Computing a realistic velocity profile for a mobile robot is a challenging task due to a large number of kinematic and dynamic constraints that are involved. Unlike prior works which performed temporal planning in a 2-dimensional environment, this thesis presents a new temporal planning algorithm in a 3-dimensional environment. This algorithm is implemented on a wheeled mobile robot that is to be used in a healthcare setting. The path planning stage is accomplished by using cubic spline functions. A rudimentary trajectory is created by assigning an arbitrary time to each segment of the path. This trajectory is made feasible by applying a number of constraints and using a linear scaling technique. When a velocity profile is provided, a non-linear time scaling technique is used to fit the robot’s center linear velocity to the specified velocity. A method for avoiding moving obstacles is also implemented. Both simulation and experimental results for the wheeled mobile robot are presented. These results show good agreement with each other. For both simulation and experimentation, six different examples of paths in the Engineering Building of the University of Saskatchewan, were used. Experiments were performed using the PowerBot mobile robot in the robotics lab at the University of Saskatchewan.

ACKNOWLEDGEMENTS

I wish to express my heartfelt gratitude and indebtedness to Dr. Reza Fotouhi for his help and direction throughout the Master’s program. Without his valuable suggestions and encouragement this work would not have been possible. I would also like to thank the members of the advisory committee, Dr. Richard Burton and Dr. Daniel Chen of the Mechanical Engineering Department. Mr. Mohammad Vakil and Mr. Reza Moazed also were of great help in their technical support and guidance.

Financial support was provided by the University of Saskatchewan Graduate Scholarship and an NSERC Discovery Grant, and is gratefully acknowledged.

DEDICATION

This thesis is dedicated to my beautiful fiancé, Hana Rekieh, and my dear parents, Mr. Abdul and Mrs. Rubina Waheed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

a = linear acceleration.

a_g = acceleration of the center of gravity.

a_i = coefficient for a cubic spline for segment i.

a_n = normal acceleration of the robot.

a_{robot} = maximum acceleration of the robot's motors.

a_{slip} = maximum acceleration to avoid wheel slippage.

$A_{\lim}$ = combined acceleration limits.

b = distance of the middle of a wheel to the midpoint of the two driving wheels.

b_i = coefficient for a cubic spline for segment i.

c = center of the driving wheels.

c_i = coefficient for a cubic spline for segment i.

d = distance.

d_i = coefficient for a cubic spline for segment i.

d_{robot} = maximum deceleration of the robot's motors.

D_{ik} = distance traveled on the i^{th} segment and k^{th} interval.

$D_{\lim}$ = combined deceleration limits.

$\|E\|$ = mean squared error norm.

F_A = frictional force at point A (driving wheels).

F_B = frictional force at point B (caster wheels).

F_c = resultant of the frictional forces of all wheels.

F_n = sum of all the normal forces.

F_t = sum of all the tangential forces.

F_z = sum of all the vertical forces.

g = acceleration of gravity.

G = location of the center of gravity.

m = mass of the robot.

M_A = sum of the moments at point A (driving wheels)

M_G = sum of the moments at the center of gravity.

M_{int} = number of intervals in each segment.

$\left(M_k\right)_A$ = moment of inertia force with respect to point A (driving wheels).

N = number of knots.

N_A = normal force at point A (driving wheels).

N_B = normal force at point B (caster wheels).

N_c = resultant of the normal forces of all wheels.

Q = total number of data points

s = set of knots.

s_i = knot i.

s_{x_i} = x direction in knot i.

t = un-scaled time.

t_i = un-scaled time for segment i.

$\overline{t}$ = scaled time.

$\overline{t_i}$ = scaled time for segment i.

T_{ik} = discrete time for segment i and interval k.

T_s = sampling time.

U = intermediate variable.

v = linear velocity.

v_{d_i} = simulated (desired) velocity of data point i.

v_{e_i} = experimental velocity of data point i.

v_{final} = final velocity.

v_i = derivative of the path.

$v_{initial}$ = initial velocity.

v_l = linear velocity of the left wheel.

v_r = linear velocity of the right wheel.

v_{robot} = maximum linear velocity of the robot’s motors.

v_s = specified velocity.

v_{safety} = safety speed limit.

v_{slide} = slide out velocity limit.

V_{ik} = discrete velocity for segment i and interval k.

$V_{\lim}$ = combined velocity limits.

W = weight of the robot.

x_j = x direction of knot j.

(x, y)=position of the center of the robot's driving wheels.

(x_{cg}, y_{cg})=position of the center of gravity of the robot.

(x_d, y_d, z_d)= simulated (desired) goal point.

(x_e, y_e, z_e)=experimental goal point.

(x_l, y_l)= position of left wheel.

(x_r, y_r)= position of right wheel.

(x_w, y_w)= fixed world coordinates.

X_i = cubic spline for the x direction for segment i.

y_j = y direction of knot j.

Y_i =cubic spline for the y direction for segment i.

z_j = z direction of knot j.

ϕ = angle of the ground with respect to a horizontal surface.

λ = scaling factor.

λ_1 = velocity scaling factor.

λ_2 = acceleration scaling factor.

λ_3 = deceleration scaling factor.

λ_i = scaling factor for segment i.

μ_s = static coefficient of friction between the robot's wheels and the ground.

θ = orientation of the robot with respect to the world coordinates

θ_j = orientation of knot j.

ρ = radius of curvature of a path.

$\rho_{\min}$ = minimum radius of curvature of the path.

CHAPTER 1: INTRODUCTION

1.1: General

Robots are essential elements in society today. They are capable of performing many tasks repetitively and precisely without the comforts required by humans. The word robot is used to refer to a wide variety of mechanical machines that are capable of movement. It comes from the Czech word “robota” meaning forced labor. The word robot first appeared in Karel Čapek's science fiction play Rossum's Universal Robots in 1921 [1].

Robots are defined as mechanical devices that are controlled by a computer [1]. In order for a machine to be classified as a robot, it must possess artificial intelligence meaning that it must be at least partially controlled by a computer. For example, a remote controlled car would not be considered to be a robot since it is completely operated by a human; whereas an industrial manipulator in a manufacturing factory would be considered a robot since it is controlled by computers.

Robots can be grouped into two main categories: manipulator robots and mobile robots. Manipulator robots are fixed to a certain location and have a fixed workspace. Workspace is defined as all the points the robot’s links and joints can reach [1]. Mobile robots are free to move around and therefore have no fixed workspace. Mobile robots may also incorporate manipulator robots onto them.

As robots grow in complexity, their use in life becomes much more apparent. Robots are widely used in the automation of manufacturing industries such as vehicle production and grain harvesting. They are ideal for such industries because the tasks that must be performed are generally repeated in exactly the same fashion. They are also useful in environments dangerous to humans such as mining, toxic waste sites, space, etc. More recently, robots are being used for domestic and personal use such as in the healthcare industry where “intelligent” wheelchairs help disabled people, or at home where domestic robots are used for vacuuming or cutting the grass.

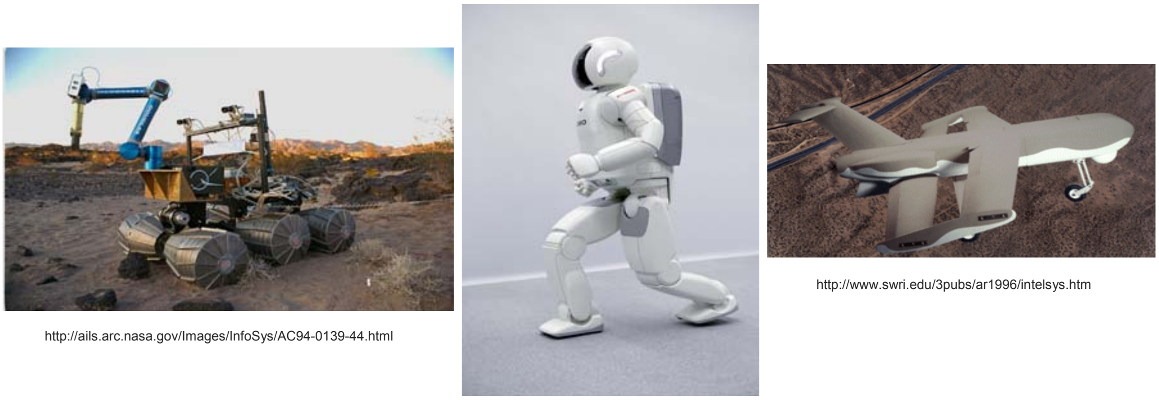
1.2: Motivation of Study: Reducing Health Care Spending in Canada

As the population ages, an essential task for Canadian society is the improvement of the quality of life for the elderly and disabled. As a result, a large expense can be related to providing support for these individuals. The future trend in the health system has its focus on home care rather than the conventional facility care. Health care costs are increasing at a faster rate than the revenue of any provincial government in Canada and health spending is increasing at almost 7% a year, a rate of increase that exceeds the Canadian Government’s rate of revenue growth [2],[3].

Robotic technology has the potential to assist the physically challenged individuals in their everyday lives, particularly in the mobility aspect. For example, in using a traditional wheelchair, an individual must navigate through close quarters in their home. Individuals controlling their movement through a joystick interface may find it frustrating and time-consuming, and it may be virtually impossible for those who have a limited physical functionality. Simple maneuvering tasks, that are a typical part of an individual's daily routine, such as a trip from the bedroom to the kitchen, can be very difficult to navigate. One solution to this problem is to create intelligent wheelchairs to help disabled people [4]. Not only do these devices assist in reducing caretaking costs, but they increase the quality of life by allowing the users to regain partial independence [5]. Some of these devices allow users to input the final destination by voice or keypad, directing the wheelchair to automatically track the path [6]. A key challenge in tracking the path is to be able to maneuver with a realistic velocity, thereby creating a practical trajectory. In this thesis, a wheeled mobile robot was used as a test bed for creating and testing an algorithm for maneuvering in an indoor environment. This algorithm can be used for generating and following a velocity profile in 3D.

1.3: Mobile Robots

A mobile robot is defined as a moving, intelligent and autonomous vehicle. A mobile robot should be able to react to its environment through the use of sensors, while having limited human interaction. Mobile robots come in many types including wheeled, tracked, legged, winged, etc. Figure 1 shows a few of these robots.



http://ails.arc.nasa.gov/Images/InfoSys/AC94-0139-44.html

http://world.honda.com/news/2005/c051213.html

http://www.swri.edu/3pubs/ar1996/intelsys.htm

Figure 1: Different types of mobile robots including the 1994 Russian built Mars rover, the Honda ASIMO robot and a military unmanned aircraft.

In this project, mobile robot refers to a wheeled and skid steered autonomous vehicle that is able to navigate through indoor environments as shown in Figure 2. The environment surrounding the robot could be classified as an indoor or outdoor setting. In the indoor setting, which is usually known and flat, a full map describing the environment is available. In an outdoor setting, which is unknown and usually uneven, a robot has little to no information about its surroundings [7].

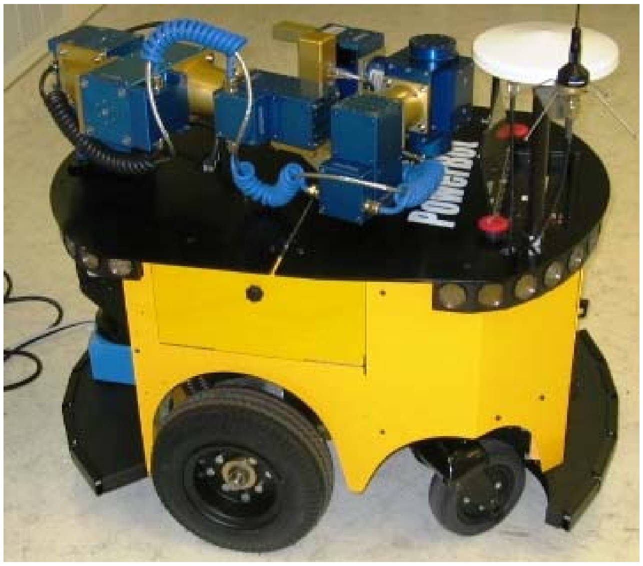


Figure 2: PowerBot mobile robot used for experimentation.

1.4: Path Planning

Mobile robot research has made considerable advances in the past three decades. Earlier works in mobile robots concentrated on path planning, where the objective was to plan a collision free path and avoid stationary obstacles, while working in an indoor environment. Stationary obstacles are defined as fixed or non moving obstructions such as walls or furniture.

The most popular method in early path planning was the *configuration space method*, where the mobile robot is shrunk to a point, while correspondingly growing the obstacles in order to compensate for taking the robot as a point [8] as illustrated in Figure

3, Figure 4 and Figure 5.

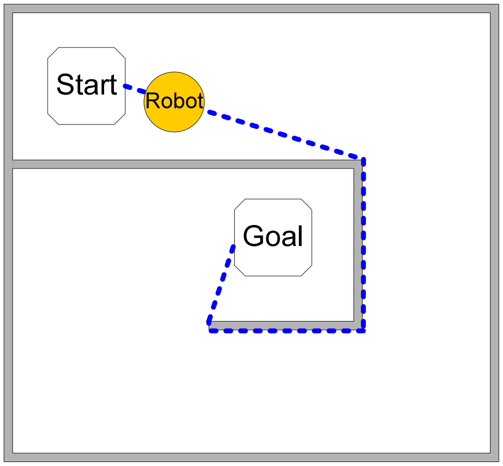


Figure 3: Configuration space method. The dotted line represents the shortest possible path to the goal. Notice how the robot would hit the wall if it were allowed to traverse the path.

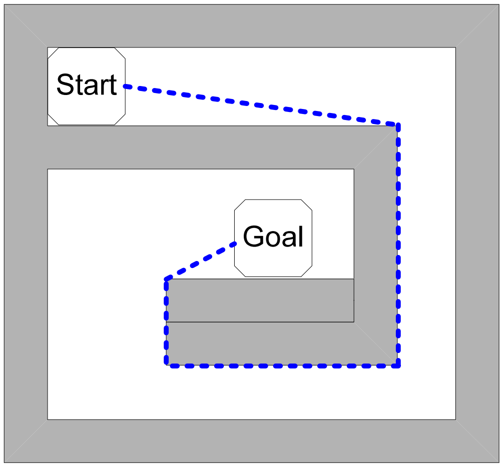


Figure 4: Configuration space method. The robot is shrunk to a point while the obstacles (walls) are increased correspondingly. The dotted line represents the shortest possible line to the goal.

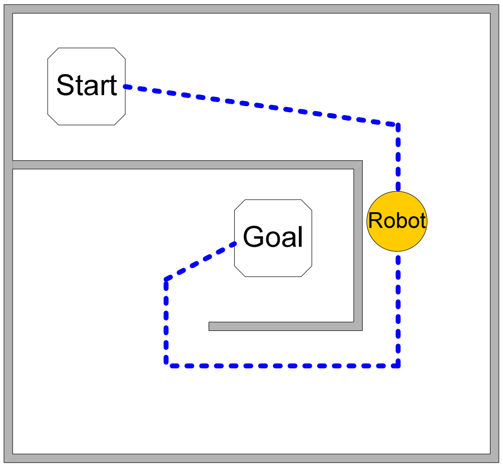


Figure 5: Configuration space method. After the path has been found, the obstacles are shrunk and the robot is grown to their normal size.

The many methods of path planning can be divided into three main categories: *roadmap approach* [9], *cell decomposition* [10], and *potential field* [11] methods. The roadmap approach is where the obstacle free area is modeled as a network of lines. This network is then searched for a path that connects the start and goal points [9]. See Figure 6, Figure 7, and Figure 8.

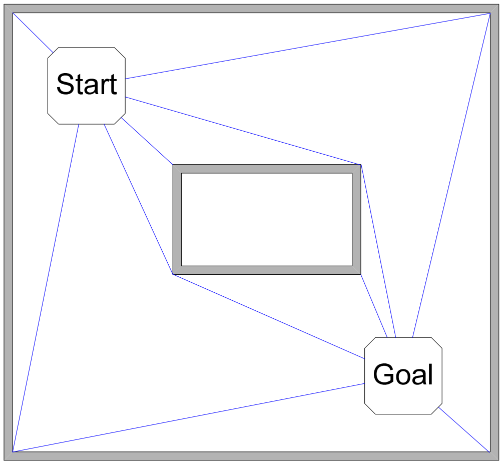


Figure 6: Roadmap approach. Lines of sight are drawn from the start and goal points to all visible vertices and corners of the map.

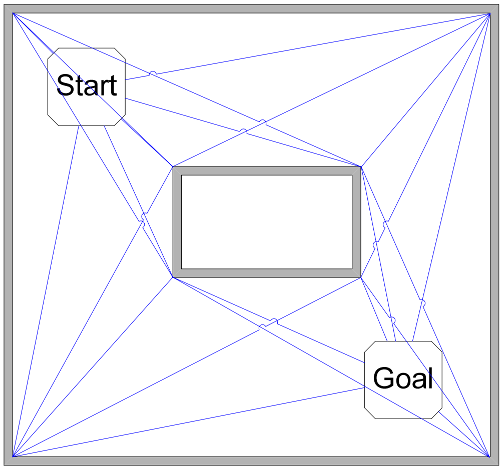


Figure 7: Roadmap approach: Lines are drawn from every corner or vertex of all obstacles to its visible line of sight.

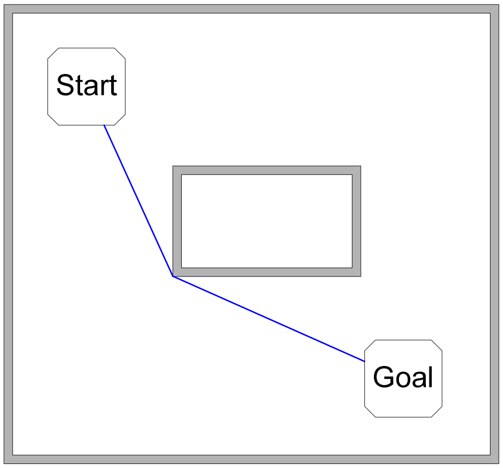


Figure 8: Roadmap approach. Based on the roadmap algorithm chosen, the lines are searched and a path connecting the start and goal points is obtained. The configuration space method could now be used to avoid the corner.

Cell decomposition is where the obstacle free area is subdivided into cells that are interconnected to each other. These cells are then searched to find a path that connects the start and goal points [10]. Figure 9 displays the free area being subdivided into cells. Figure 10 and Figure 11 demonstrate the cells being searched for a path, and a path being created respectively.

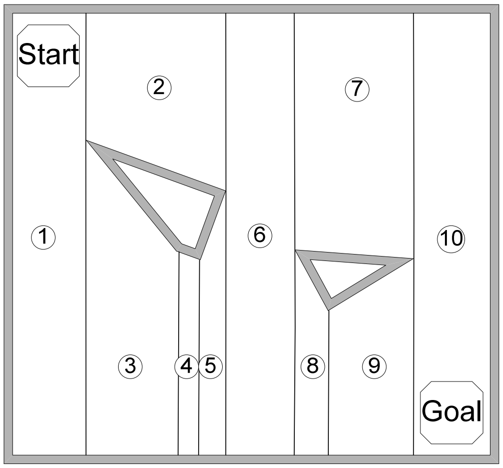


Figure 9: Cell decomposition. The free space is decomposed of cells which are created by using the vertices of the obstacles.

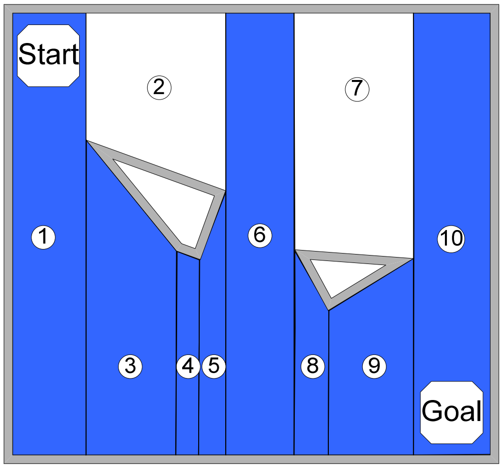


Figure 10: Cell decomposition. The graph is searched for a series of consecutive cells (shaded).

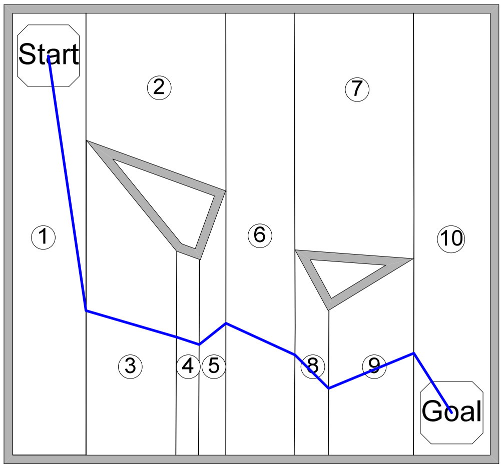


Figure 11: Cell decomposition. The sequence of consecutive cells is transformed into a path by connecting the midpoints of the intersection of two cells.

Potential field methods use imaginary forces acting on a robot. The goal position attracts the robot by pulling it towards the goal, whereas the obstacles repulse the robot by pushing it away [11].

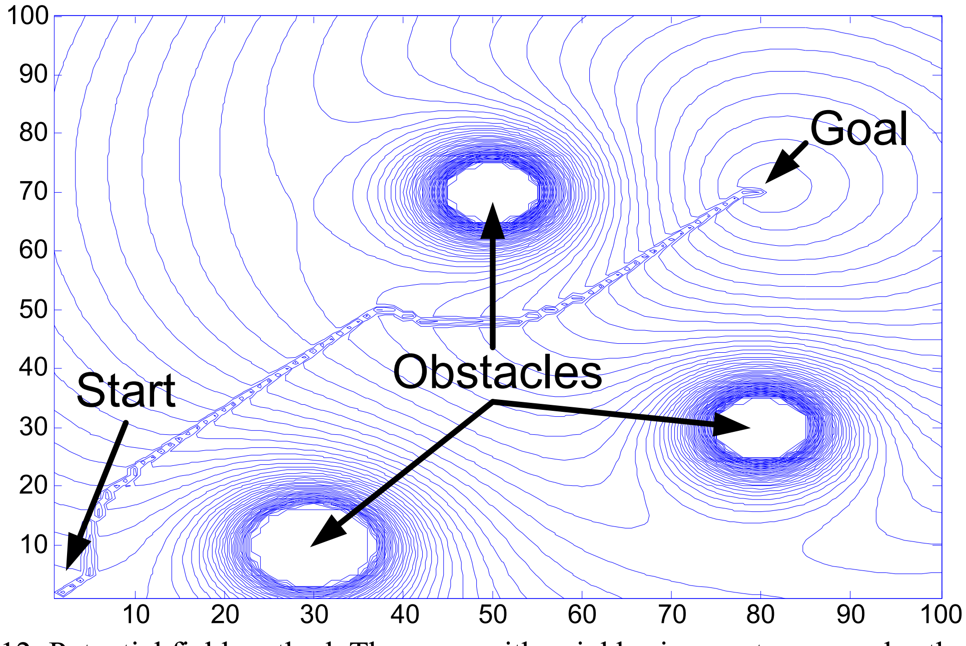


Figure 12: Potential field method. The areas with neighboring contours repulse the robot.

1.5: Outdoor Path Planning

Later works added the ability to navigate in outdoor terrains, while avoiding stationary obstacles. Outdoor environments pose a greater challenge over the typically flat indoor environments. Not only should a robot avoid obstacles, but it must also avoid steep terrain that may cause it to tip over [12]. In order to account for large terrain sizes, Cherif [13] proposed a multi-layered approach. This multi-layered approach decomposed mobile robot navigation into two levels: a global and local planner. The role of the global planner was to provide a general direction towards the target goal. The local planner was used to react to sensory data and avoid any obstacles the robot may have encountered.

Moving obstacles provide a new challenge in path planning. Qu et al. [14] introduced a technique that used a piecewise constant polynomial parameterization of all feasible trajectories in order to generate a trajectory in the presence of moving obstacles. This parameterization allowed for speed changes of obstacles as well as the appearance and the disappearance of obstacles from sensor range. Prado et al. [15] provided another method to avoid moving obstacles by modifying the velocity profile of the mobile robot. If a moving obstacle was encountered, the robot speed was reduced until the obstacle had cleared the path.

1.6: Trajectory Planning

A trajectory is a path which is an explicit function of time. To have smooth movement, the trajectory must be twice differentiable to give a continuous velocity and acceleration. As a result, curve fitting is an integral part of trajectory planning. There are a number of techniques used in curve fitting, including the use of B-splines, cubic splines

[16], clothoids, etc [17]. Simple path planning techniques assume that the robot is omni-directional and is able to execute paths flawlessly. However, in the real world, these assumptions are often not valid. Planning a trajectory which disregards the robot constraints has a profound impact on the ability of the robot to track the path [18].

Earlier works in trajectory planning added simplified kinematic constraints to plan a feasible trajectory. Louste et al. [11] used the minimum turning radius and the dimensions of a robot to plan a trajectory. The path was broken up into straight and curved segments and the velocity was individually planned for each segment. As a result the path had a smooth trajectory with a continuous first derivative. Other works [19] added dynamic constraints such as upper velocity and acceleration bounds in order to make dynamical effects negligible. This was done by setting the upper velocity and acceleration to very low values as compared to the characteristics of the robot. Munoz et al. [18] proposed a method to create a smooth trajectory subject to kinematic and dynamic constraints. The trajectory planner created the shortest obstacle free trajectory, which was smoothed out by using B-spline curves. Constraints such as velocity and acceleration limits based on the limits of the servomotors, maximum curvature, wheel ground adhesion, and a maximum lateral acceleration were added. Choi et al. [20] presented a method on how to build a trajectory of a curve. His approach split the curve into multiple sections that were modeled separately, while ensuring the continuity of motion. The constraints utilized were: battery voltage, motor current, simplified dynamics of the robot, and initial and final positions and velocities. Cherif [13] went a step further and made a detailed model of both the terrain and the robot, and used a multi-level motion planner for an outdoor setting. The algorithm consisted of a high level motion planner which minimized the distance between the present robot location and the desired end location. A local level motion planner attempted to attain the goals set by the higher level. This was done by computing wheel accelerations, contact forces, equations of motion and the new state of the deformable regions in the terrain. This algorithm also incorporated the wheel ground interaction, added a tip over constraint, and a bounded control torque constraint.

1.7: Temporal Planning

Temporal planning is to follow a given velocity profile. This may be used to minimize time of motion and to avoid moving obstacles. Fotouhi et al. [21] proposed a velocity planning algorithm for a two-link rigid manipulator. A rudimentary trajectory was created by assigning an arbitrary time to a path. Using linear time scaling, this trajectory was modified to take into account velocity and acceleration constraints of the manipulator. In order to have the manipulator end effector follow any predefined velocity profile, a non-linear time scaling technique was presented. This algorithm can be implemented on a wheeled mobile robot. Munoz et al. [22] proposed a temporal planning algorithm where the maximum possible velocity was determined by using a number of constraints. This algorithm contains three main parts. The first was the temporal planning process, where the maximum possible velocity was determined using the following constraints: velocity, acceleration, wheel slippage, moments, and a built-in safety speed. The next step was to create a smooth path using B-splines while avoiding moving obstacles. The final step was to merge the maximum velocity with the path, thereby creating a trajectory. While most other works use constant upper velocity, acceleration, and deceleration bounds, Prado et al. [23] created upper bound velocity, acceleration and deceleration functions which changed with the path. In a more recent work, Prado et al. [15] introduced an approach which created a velocity profiling technique by dividing the path into smaller segments. The velocity profile of each segment was generated by a cubic polynomial which was then integrated to obtain position and differentiated to obtain the acceleration. The profile was checked with a number of constraints and was iteratively modified until the constraints were fully satisfied. He also provided a method to avoid moving obstacles by modifying the velocity profile of the mobile robot. If a moving obstacle was encountered, the robot speed was reduced until the obstacle had cleared the path.

1.8: Research Objectives and Methodology

The research objectives are as follows

- The objective of this study was to plan a 3D trajectory (taking into account the changes in elevation of the ground) for a wheeled mobile robot, while at the same time follow a created (or given) velocity profile.
- Conduct simulations and experimental tests in an indoor setting to establish validity for the proposed algorithm.

In this project, the work of Prado et al. [15] and Munoz et al. [22] is used to enforce the dynamic and mechanical limitations, while the work of Fotouhi et al. [21] is expanded from robot manipulators to mobile robotics. The path planning stage was accomplished by using a piecewise cubic spline function to create a smooth path using kinematic constraints. An arbitrary time is assigned to each segment of the path to create an initial trajectory. Using mechanical and dynamic constraints, linear time scaling is used to make the initial trajectory a feasible one. When other moving objects existed, a method for avoiding moving obstacles was also implemented. When moving obstacles were blocking the path, the velocity profile was modified in real time to avoid a collision. Finally a non-linear scaling technique is used to follow a given velocity profile. Unlike the works done by Prado et al. [15] and Munoz et al. [22] where temporal planning was done on a flat surface, a vertical component is added to the temporal planning process. The advantage of adding the vertical component (the 3rd dimension) is that it allows for the mobile robot to be utilized in a real world environment, where the terrain is not always horizontal. The robot can only travel on ground; however, this algorithm takes into account changes in elevation thereby making it a 3D algorithm. Simulation and experimental work was implemented using the wheeled mobile robot, PowerBot, in the Robotics Lab at the Mechanical Engineering Department in the University of Saskatchewan. Successful implementation of the new algorithm for 3D temporal planning is an original contribution of this thesis.

1.9: Outline of Thesis

Chapter 1 provided an introduction to the subject and introduced a few key topics which will be covered in later chapters. Chapter 2 encompasses the theory for mobile robot temporal profiling. This includes generating the initial trajectory using cubic splines and modifying the initial trajectory to take into account a number of constraints. Chapter 3 describes the implementation of the methodology discussed in the previous section on a wheeled mobile robot. This chapter includes moving obstacle avoidance. Chapter 4 presents the simulation and experimental study. The proposed algorithm is tested using a number of different paths and conditions. Chapter 5 covers the conclusions and suggested future work.

CHAPTER 2: MOTION PLANNING

2.1: Path Planning

Let $s = \{s_1, s_2, \ldots, s_N\}$ be a route as a set of points (knots) computed by a path planner as shown in Figure 13a. The actual method for obtaining these points will not be discussed, as this has been the topic of considerable research. Each point s_j is composed of four basic elements: $x_j, y_{j,} z_j$ and θ_j where $j = 1, 2, \ldots, N$. The first three components, $x_j, y_{j,} z_j$, define position relative to a global reference frame shown in Figure 13a. The last component, θ_j, is the heading (orientation) from one point to the next measured in the $x - y$ plane and from positive x. There are several methods to join these points with a curve. The easiest method is to connect the points with straight lines; however, there are sudden changes in slope which in mobile robotics means a discontinuous velocity. Another method could be to use a high order polynomial. However, if there are numerous points, the path may become wavy due to the oscillatory nature of high order polynomials [24]. A more effective method is the use of piecewise cubic polynomials, also known as cubic splines. These cubic splines are ideal since they provide a continuity in position, heading, curvature, velocity, and acceleration [21],[24].

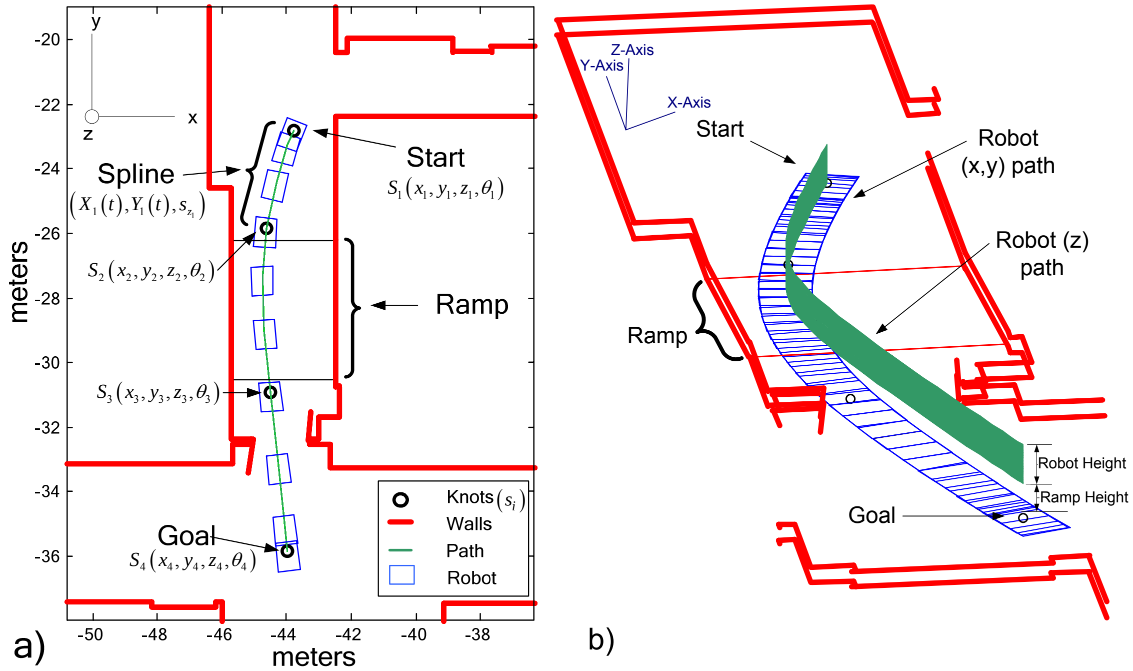


Figure 13: Three cubic splines (N=4) for experiment 4. a) 2D View. b) 3D view.

2.2: Cubic Splines

Considering the points returned by the path planner in only the x direction, s_x, the i^{th} segment can be constructed as:

$$X_i(t) = a_i\left(t-1\right)^2\left(2t+1\right) + b_i t^2(3-2t) + c_i t\left(t-1\right)^2 - d_i t^2\left(1-t\right) \tag{2.1}$$

where X_i is the parameter connecting the points s_i and s_{i+1} in the x - direction, $0 \le t \le 1$ is the relative time of motion in each segment $i = 1, 2, \ldots, N-1$ segments, and N is the total number of points (knots) used in the path. Any cubic equation can be used to construct a cubic spline by identifying the constants a_i, b_i, c_i, and d_i; however, the Hermite cubic polynomial [24] of equation (2.1) has a unique property where it satisfies all four of the following boundary conditions:

$$X_i\left(0\right) = a_i, \frac{dX_i}{dt}\left(0\right) = c_i, X_i(1) = b_i, \frac{dX_i}{dt}(1) = d_i. \tag{2.2}$$

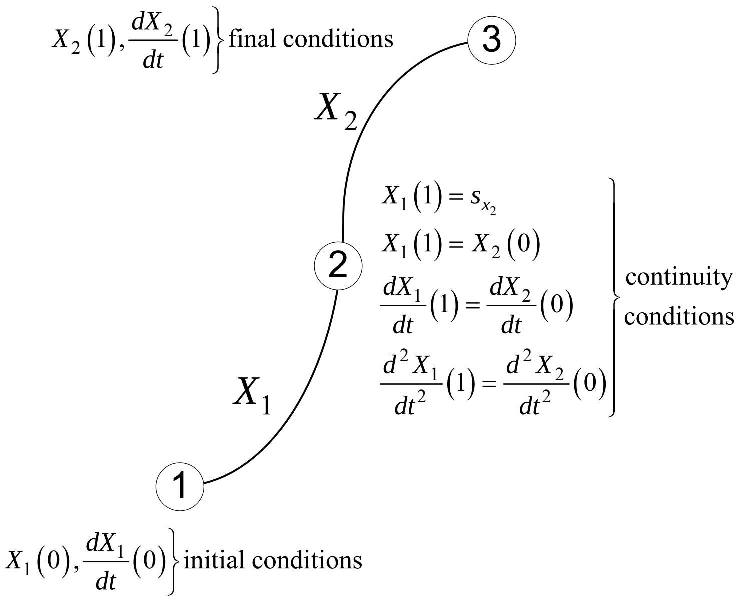
The first two boundary conditions are to assign initial values to the position and velocity, respectively. The last two boundary conditions are to assign the final position and velocity values. To ensure continuity between each polynomial, additional constraints must be set:

$$\begin{aligned}
&X_i(1) = s_{x_{i+1}} \\
&X_i(1) = X_{i+1}(0) \\
&\frac{dX_i}{dt}(1) = \frac{dX_{i+1}}{dt}(0) \\
&\frac{d^2X_i}{dt^2}(1) = \frac{d^2X_{i+1}}{dt^2}(0).
\end{aligned} \tag{2.3}$$

All of these equations give a total of $4(N-1)$ equations for $4(N-1)$ unknowns for N knots. For example, for 2 segments (3 knots), 4 boundary conditions and 4 continuity conditions exist. This equals 8 equations and 8 unknowns ($a_i, b_i, c_i, d_i; i = 1, 2$). This system of equations (equation (2.2) and equation (2.3)) can be arranged in a simple symmetric tridiagonal system [16] of equations:

$$\begin{bmatrix} 1 & & & & & \\ 1 & 4 & 1 & & & \\ & 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & & 1 & 4 & 1 \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ \vdots \\ U_{N-2} \\ U_{N-1} \end{bmatrix} = \begin{bmatrix} v_{initial} \\ 3\left(s_{x_3} - s_{x_1}\right) \\ 3\left(s_{x_4} - s_{x_2}\right) \\ \vdots \\ 3\left(s_{x_{N-1}} - s_{x_{N-3}}\right) \\ v_{final} \end{bmatrix} \tag{2.4}$$

where U is an intermediate variable, $v_{initial}$ is the initial velocity and v_{final} is the final velocity of the path. Figure 14 displays a cubic spline and its initial and boundary conditions. For an example of a 3 knot spline, see Appendix 1.



$X_2(1), \frac{dX_2}{dt}(1)$ final conditions
3
X_2
$X_1(1) = s_{x_2}$
$X_1(1) = X_2(0)$
$\frac{dX_1}{dt}(1) = \frac{dX_2}{dt}(0)$
$\frac{d^2X_1}{dt^2}(1) = \frac{d^2X_2}{dt^2}(0)$
continuity conditions
2
X_1
1
$X_1(0), \frac{dX_1}{dt}(0)$ initial conditions

Figure 14: A cubic spline and its boundary conditions for N=3 points.

Solving for the U 's allows for obtaining the coefficients of the spline:

$$\begin{aligned} a_i &= s_{x_i} \\ b_i &= s_{x_{i+1}} \\ c_i &= U_{i-1} \\ d_i &= U_i. \end{aligned} \tag{2.5}$$

Since mobile robotic motion can be in more than one dimension, a separate set of equations must be solved for each dimension, $X_i(t), Y_i(t)$. For wheeled robots, the vertical component does not require a spline since the robot always moves on the ground. Using the parametric equations $X_i(t)$ and $Y_i(t)$, and the z value in the set s, a smooth path can be created connecting all the points in s. This can be seen in Figure 13a for a 2-dimensional path (x, y) and Figure 13b for a 3-dimensional path (x, y, z).

2.3: Trajectory Planning

2.3.1: Initial Trajectory

Once a smooth path has been created using cubic splines, an arbitrary duration time of 1 second is assigned to each spline of the path. By taking the derivative of the distance with respect to time, a rudimentary velocity profile is created. Since only an arbitrary time period was assigned, this velocity profile must be modified to give a more feasible solution, which takes into account mechanical and dynamic constraints of the robot.

2.3.2: Robot Constraints

2.3.2.1: Velocity Limits

The linear velocity assigned, to each point of the trajectory, cannot be higher than the attainable velocity of the mobile robot. In the case of the PowerBot, the maximum limit is:

$$v_{robot} = 2.1 \text{ m/s} \tag{2.6}$$

as provided by the manufacturer [25] and also verified by experimentation.

2.3.2.2: Acceleration and Deceleration Limits

The maximum acceleration and deceleration limits were not provided by the robot's manufacturer, and therefore had to be determined experimentally. This was done by assigning the robot with an unattainable velocity profile as shown in Figure 15.

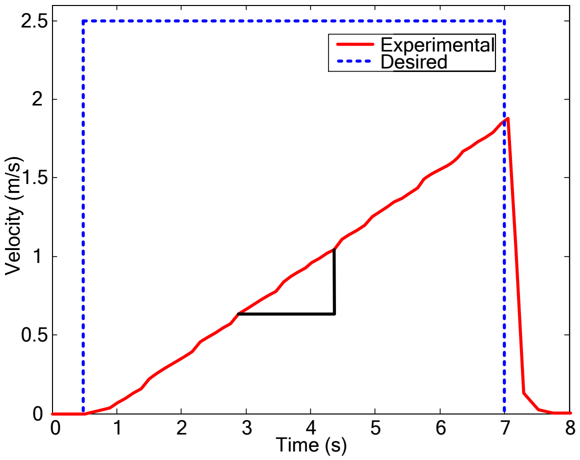


Figure 15: Experimental (actual) and desired velocity-time graph of an unattainable velocity profile used to determine the maximum acceleration and deceleration of the PowerBot mobile robot.

The acceleration of the robot was computed by taking the derivative of the actual velocity of the robot with respect to time. By taking the maximum and minimum values of the acceleration, the acceleration and deceleration limits were determined. This can be seen in Figure 16.

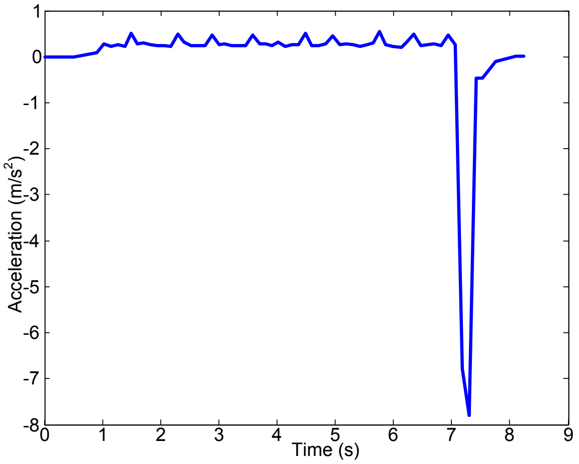


Figure 16: Experimental acceleration-time graph showing the maximum acceleration and deceleration limits of the PowerBot mobile robot.

It was determined by doing a series of two tests that the maximum acceleration is:

$$a_{robot} = 0.55 \text{ m/s}^2 \tag{2.7}$$

and the maximum deceleration is:

$$d_{robot} = -7.8 \text{ m/s}^2. \tag{2.8}$$

2.3.2.3: Maximum Velocity to Avoid Sliding Out

The maximum velocity when traveling along a curve depends on the curvature of the path and the wheel ground adhesion. Figure 17 shows the free body diagram for the robot traveling along a curved path with a radius of ρ.

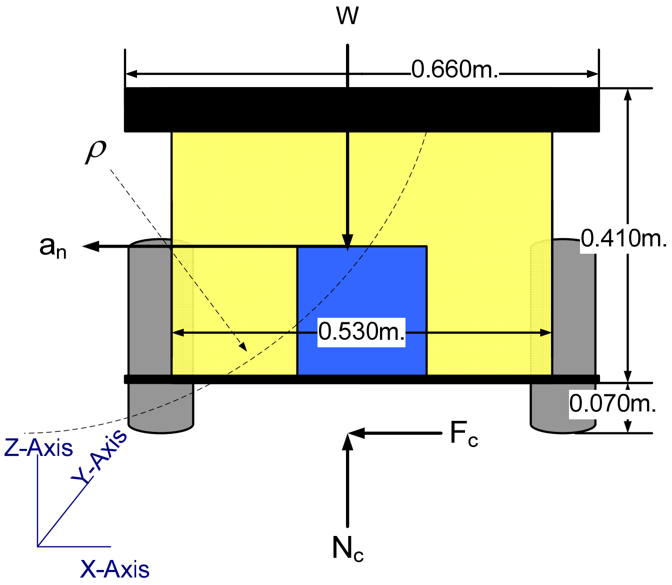


Figure 17: Free body diagram showing the PowerBot mobile robot on a curved path shown in the x-z plane (ρ is in the x-y plane).

N_c and F_c represent the resultant of the normal and frictional forces for all wheels respectively. W is the weight of the robot and a_n is the normal acceleration. The forces in the n direction can be expressed as:

$$\overset{+}{\leftarrow}\sum F_n = ma_n; \qquad F_c = ma_n. \tag{2.9}$$

F_c can be expressed in terms of the normal force N_c as shown:

$$F_c = \mu_s N_c. \tag{2.10}$$

Where μ_s is the static coefficient of friction between the ground and the wheels. This value was experimentally determined to be 0.332. See Appendix 2 for more details. The forces in the z direction can be expressed as follows:

$$+\uparrow \sum F_z = 0; \qquad W = mg = N_c. \tag{2.11}$$

The normal acceleration, a_n, can be expressed in terms of velocity, v, and the radius of curvature, ρ:

$$a_n = \frac{v^2}{\rho} \tag{2.12}$$

The radius of curvature ρ [26] can be expressed as:

$$\rho = \frac{\left(\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2\right)^{3/2}}{\left|\frac{dx}{dt}\frac{d^2y}{dt^2} - \frac{dy}{dt}\frac{d^2x}{dt^2}\right|} \tag{2-13}$$

Where $x(t)$ and $y(t)$ represent the trajectory. Combining equations (2.9), (2.10) and (2.12) results in:

$$\mu_s N_c = m\frac{v^2}{\rho} \tag{2.14}$$

Solving for v and substituting equation (2.11) into (2.14) provides the maximum velocity to avoid sliding out as shown:

$$v_{slide} = \sqrt{\mu_s g \rho_{\min}} \tag{2.15}$$

where $\rho_{\min}$ is the smallest radius of curvature of the entire path.

2.3.2.4: Wheel Ground Interaction to Avoid Slippage

The maximum acceleration to avoid wheel slippage is bounded by the wheel-ground adhesion. In order for wheel slippage not to occur, the frictional force must not overcome the normal force multiplied by the coefficient of friction. The maximum acceleration can be calculated by setting:

$$F_A = \mu_s N_A. \tag{2.16}$$

F_A is the frictional force, N_A is the normal force and μ_s is the static coefficient of friction between the wheels and the ground (See Appendix 2 for details). The free body diagram for the problem is shown in Figure 18. In order to find the maximum acceleration, the equations of motion must be derived and solved. For details in the calculation of the center of gravity, G, refer to Appendix 3. The mass of the robot (including batteries) is $m = 148kg$.

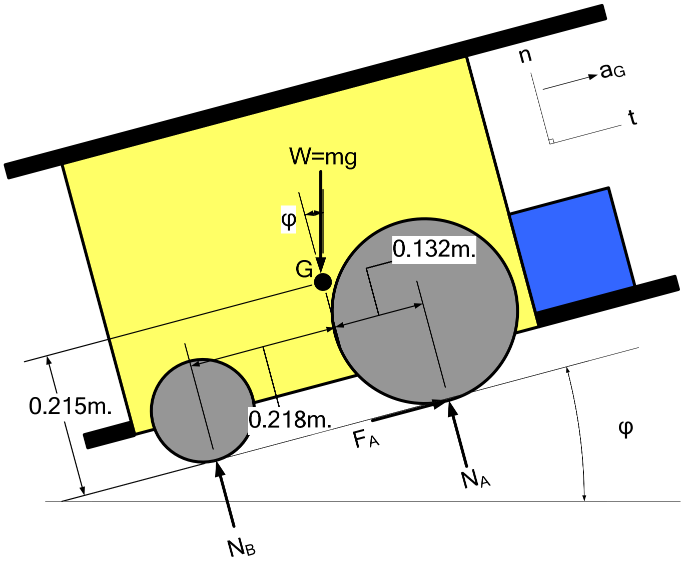


Figure 18: Free body diagram to find the maximum acceleration before wheel slippage.

The equations of motion can be expressed as follows:

$$+\nearrow \sum F_t = ma_G; \qquad F_A - 148g\sin(\varphi) = 148a_G \tag{2.17}$$

$$+\nwarrow \sum F_n = 0; \qquad N_B + N_A - 148g\cos(\varphi) = 0 \tag{2.18}$$

$$\circlearrowright + \sum M_G = 0; \qquad \left(0.218\right)N_B - \left(0.132\right)N_A - \left(0.215\right)F_A = 0 \tag{2.19}$$

N_A is the normal force acting on the driving wheels, F_A is the frictional force of the driving wheels, N_B is the normal force acting on the caster wheels, a_G is the acceleration of the center of gravity of the robot, and φ is the angle of the ground with respect to a horizontal surface. The angle is bound to $-8.53° \le \varphi \le 8.53°$, which is about a 15% grade [25]. The frictional force of the caster wheels is neglected since they are assumed to be free-rolling. The equations of motion along with equation (2.16) give a total of 4 equations for 4 unknowns; therefore, the maximum acceleration, before slippage occurs, can be calculated to be:

$$a_{slip} = a_G = \frac{-16.2\cos\left(\varphi\right)}{\mu_s + 1.63} + 9.95\cos\left(\varphi\right) - 9.81\sin\left(\varphi\right). \tag{2.20}$$

The maximum deceleration, before slippage occurs, can be calculated to be:

$$d_{slip} = -a_{slip}. \tag{2.21}$$

2.3.2.5: Maximum Acceleration to Avoid Tipping Over

The acceleration must be bounded in order to prevent the robot from tipping over. The maximum acceleration, when the robot is heading downward, can be calculated by setting the normal force of the rear wheel to:

$$N_B \approx 0. \tag{2.22}$$

The free body diagram is shown in Figure 19. In order to find the maximum acceleration, the moment equation of motion must be derived and solved.

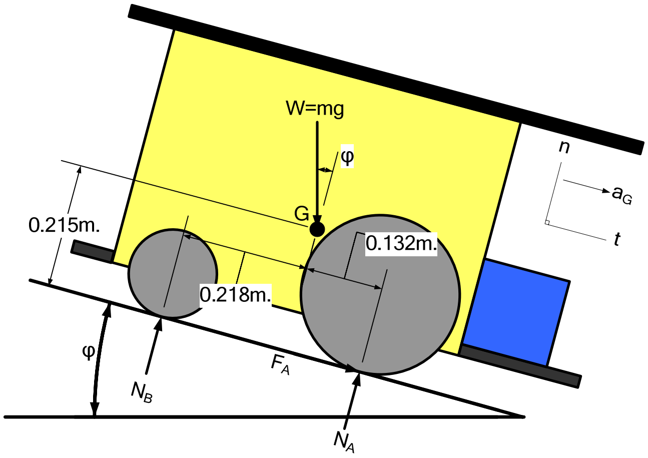


Figure 19: Free body diagram for calculating the maximum acceleration before tipping over.

The moment equation of motion [27] can be expressed as follows:

$$\begin{gathered}\circlearrowright + \sum M_A = \sum (\mathrm{M}_k)_A; \\ 148g\sin(\varphi)(0.215) - 148g\cos(\varphi)(0.132) = -148a_G(0.215)\end{gathered} \tag{2.23}$$

where $(\mathrm{M}_k)_A$ is the moment of inertia force $(-ma_G)$ with respect to point A. The maximum acceleration to prevent the robot from tipping over can be calculated to be:

$$a_{tip} = a_G = 0.614g\left(\cos(\varphi) - 1.63\sin(\varphi)\right) \tag{2.24}$$

where g is the acceleration of gravity. The maximum deceleration to prevent the robot from tipping over can be calculated to be:

$$d_{tip} = -a_{tip}. \tag{2.25}$$

2.3.2.6: Safety Speed Limit

An upper bound safety speed limit is set to avoid moving obstacles. As explained in Section 3.6, the robot is equipped with sonar sensors that check for obstacles roughly every 0.5 seconds and have a maximum range of 0.6 m. It takes another 0.5 seconds for the robot to react to the obstacle. The maximum allowed robot velocity can then be computed by:

$$v'_{safety} = \frac{0.6}{0.5+0.5} = 0.6\,{}^{m}\!/_{\mathrm{s}}\,. \tag{2.26}$$

Since sonar sensors are not flawless, velocity is reduced to 0.5 m/s as an additional safety factor; therefore:

$$v_{safety} = 0.5\,{}^{m}\!/_{s}\,. \tag{2.27}$$

2.3.3: Feasible Trajectory

Using the constraints in the previous section, the values for the maximum velocity, acceleration, and deceleration can be obtained. The velocity, acceleration, and deceleration of the robot have the following constraints:

$$v(t) \leq V_{\mathrm{lim}} \tag{2.28}$$

$$a(t) \leq A_{\mathrm{lim}} \tag{2.29}$$

$$d(t) \geq D_{\mathrm{lim}} \tag{2.30}$$

where

$$V_{\mathrm{lim}} = \min\left(v_{robot}, v_{slide}, v_{safety}\right) \tag{2.31}$$

$$A_{\mathrm{lim}} = \min(a_{robot}, a_{tip}, a_{slip}) \tag{2.32}$$

$$D_{\mathrm{lim}} = \min\left(d_{robot}, d_{tip}, d_{slip}\right) \tag{2.33}$$

where v_{robot}, v_{slide}, and v_{safety} are given in equations (2.6), (2.15), and (2.27) respectively, a_{robot}, a_{tip}, and a_{slip} are given in equations (2.7), (2.24), and (2.20) respectively, and d_{robot}, d_{tip}, and d_{slip} are provided in equations (2.8), (2.25), and (2.21) respectively.

To create a feasible trajectory, the initial trajectory is modified by linear scaling of the time as introduced in [21]. The time variable t in the initial trajectory is replaced by a scaled time variable $\bar{t}$:

$$\bar{t} = \lambda t, \tag{2.34}$$

where λ is the scaling factor. The scaling factor can be selected as follows:

$$\lambda_1 = \max\left(\left|v(t)/V_{\lim}\right|\right) \tag{2.35}$$

$$\lambda_2 = \max\left(\left|a(t)/A_{\lim}\right|\right) \tag{2.36}$$

$$\lambda_3 = \max\left(\left|d(t)/D_{\lim}\right|\right) \tag{2.37}$$

$$\lambda = \max\left(1, \lambda_1, \sqrt{\lambda_2}, \sqrt{\lambda_3}\right). \tag{2.38}$$

The resulting scaled time, velocity, and acceleration are:

$$\bar{t} = \lambda t \qquad v(\bar{t}) = {}^{1}\!/_{\lambda}\, v(t) \qquad a(\bar{t}) = {}^{1}\!/_{\lambda^2}\, a(t). \tag{2.39}$$

2.4: Following a Specified Velocity Profile

When a given velocity profile needs to be followed, a nonlinear scaling technique was used in [21]. The same method is employed as in the previous section. However, rather than scaling the time over the entire path, the time is scaled separately between each adjacent knots. The scaling factor λ_i, for each segment, can be calculated as follows:

$$\lambda_i = \max\left(\left|{}^{v_i(t)}\!\big/_{v_s(t)}\right|\right), \tag{2.40}$$

where $i = 1, 2, \ldots, N-1$ segments, $v_i(t)$ is the actual velocity, and $v_s(t)$ is the specified velocity. The resulting scaled time, velocity, and acceleration are:

$$\overline{t_i} = \lambda_i t_i \qquad v\left(\overline{t_i}\right) = {}^{1}\!/\!{}_{\lambda_i}\, v\left(t_i\right) \qquad a\left(\overline{t_i}\right) = {}^{1}\!/\!{}_{\lambda_i{}^2}\, a\left(t_i\right). \tag{2.41}$$

CHAPTER 3: IMPLEMENTATION

3.1: Mobile Robot used for Experimentation

All simulation and experiments were based on the PowerBot mobile robot in the University of Saskatchewan's Robotics Lab. This robot is an ideal platform for research on wheeled mobile robots because it is designed to run on floor and paved surfaces having a grade of up to 15% (about 8.5^{o}). The robot can reach speeds of up to 7 km/h and carry a mass of up to 100 kg on its body. It contains a number of sensors including a laser navigation finder, two arrays of sonar sensors, bumper sensors, cameras, and a GPS system. The robot platform consists of a balanced drive system (two-wheel differential with balancing casters), reversible DC motors, motor-control and drive electronics, high-resolution motion encoders, and long-life battery power, all managed by an onboard microcontroller and a mobile-robot server software [25]. Figure 20 and Figure 21 show the PowerBot robot and its dimensions.

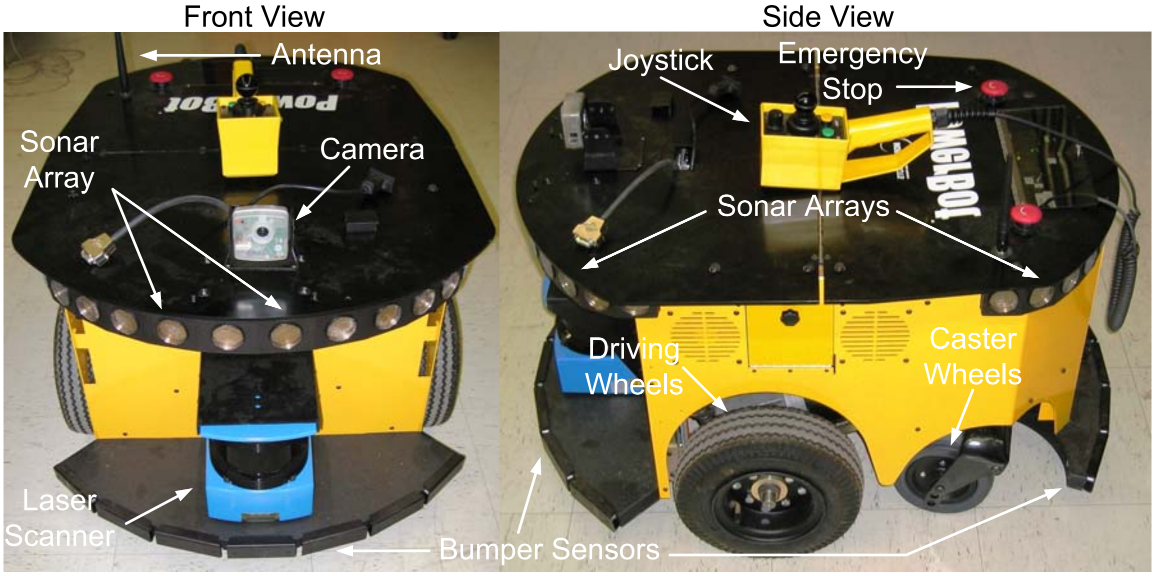


Figure 20: PowerBot wheeled mobile robot.

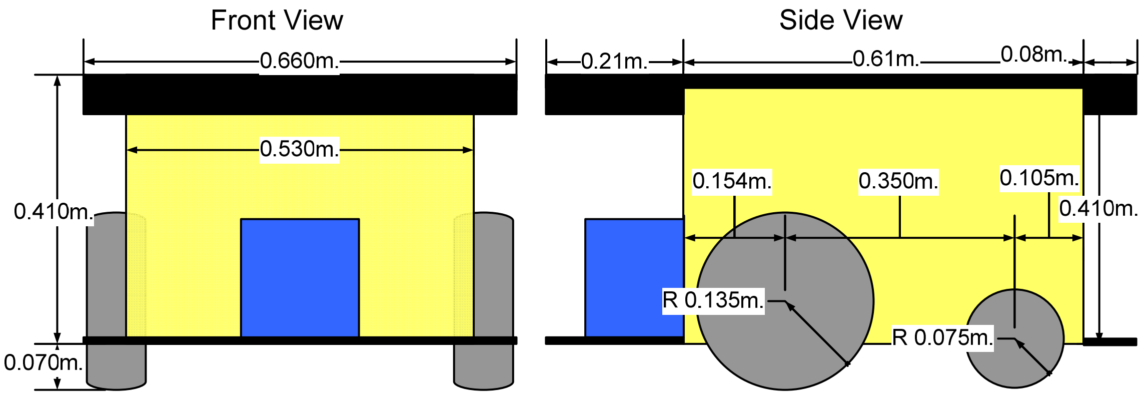


Figure 21: PowerBot wheeled mobile robot schematic.

3.2: Software Architecture

The PowerBot mobile robot contains an onboard microcontroller that handles the low level details such as maintaining the platform's drive speed and heading over uneven terrain, acquiring sensor readings, and managing attached accessories [25]. It also comes with an onboard PC computer that provides the user with a communication and control interface with the microcontroller through a serial cable. The microcontroller is controlled by the ActivMedia Robot Control and Operations Software (ARCOS).

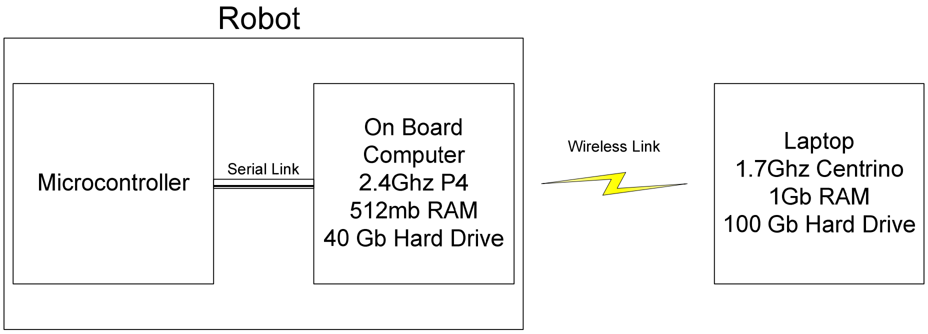


Figure 22: Computer hardware controlling the PowerBot mobile robot.

The onboard computer contains a number of software programs that allow it to control the microcontroller as well as communicate with computers that are not physically located on the robot. The software packages used in this project were:

- ActivMedia Robotics Interface for Application (ARIA) v2.4.1
- ActivMedia Robotics Navigation and Localization (ARNL) v1.4.5
- Mapper3 v1.2.1
- MobileEyes v1.3.7
- MobileSim v0.3
- MATLAB v7.0
- Microsoft Visual C++ 2003.NET

ARIA and ARNL are C++ libraries used for controlling the robot. Mapper3 is a map editing tool. MobileEyes is a multipurpose map building, navigation, and localization tool. MobileSim is a PowerBot simulator. MATLAB was used for simulations, analysis, and for converting a velocity profile into wheel velocities as input to the robot. Visual C++ was used to develop customized programs for this project. The software used and created for this project is shown in Figure 23 and is discussed in the following sections.

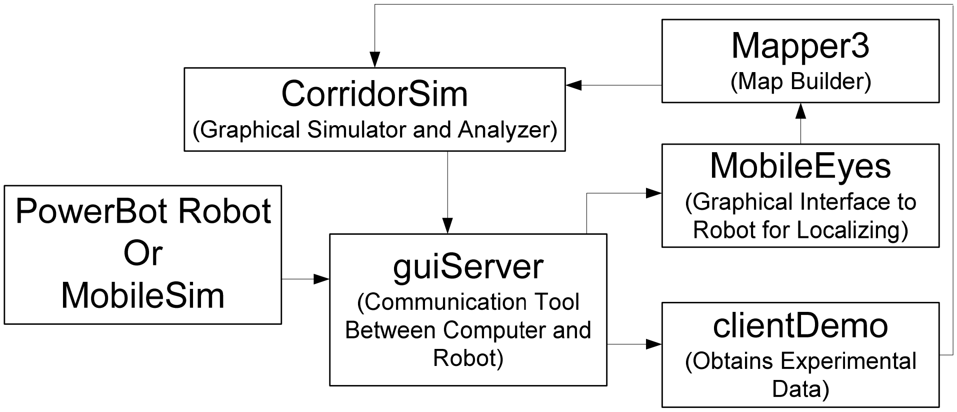

CorridorSim
(Graphical Simulator and Analyzer)
Mapper3
(Map Builder)
MobileEyes
(Graphical Interface to Robot for Localizing)
PowerBot Robot Or MobileSim
guiServer
(Communication Tool Between Computer and Robot)
clientDemo
(Obtains Experimental Data)

Figure 23: Block diagram of the software used in this project. Shows how the software interacts with each other.

3.2.1: CorridorSim Analyzer

A MATLAB program, CorridorSim, was written, specifically for this project, as a simulator for the PowerBot mobile robot. It features an easy to use graphical menu interface as shown in Figure 24.

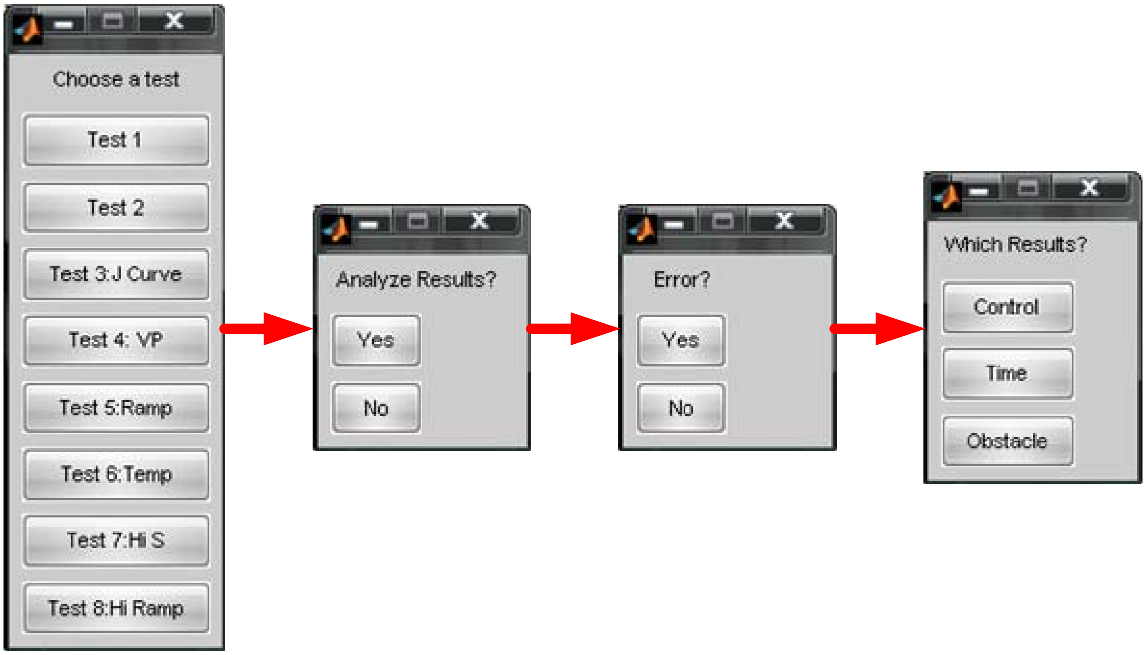
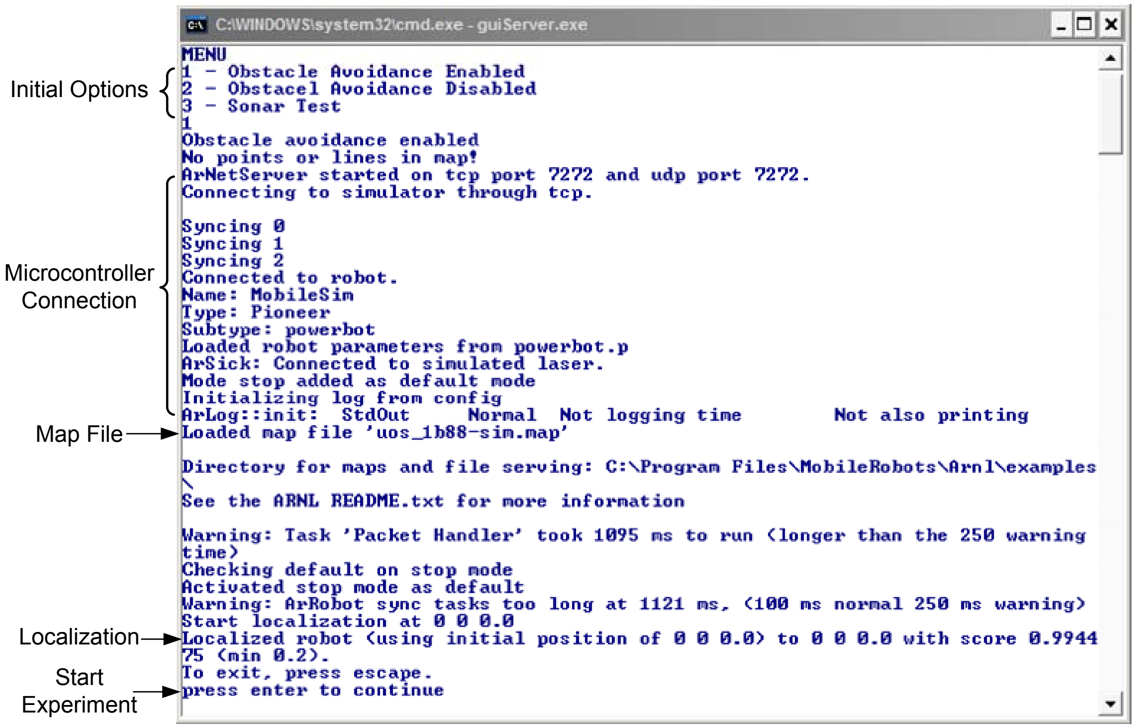


Figure 24: CorridorSim graphical user interface.

This program provides up to 20 graphs and figures for each experiment. It provides information such as the position, heading, velocity, acceleration, time, etc. It also exports the test velocity profile to the robot. The final function of this program is to import and analyze the raw data returned by the robot after experimentation has been completed.

3.2.2: guiServer

The ARNL library provided an example program, guiServer, which was modified to be specifically used in this project. GuiServer communicates directly with the robot microcontroller and imports and implements the velocity profile from the CorridorSim program. It also acts as a server, allowing other programs to connect and interact with the robot in a more user friendly manner. GuiServer has advanced features such as map making capabilities (when paired with MobileEyes), robot localization, and obstacle avoidance. Figure 25 shows the command line interface of guiServer.


C:\WINDOWS\system32\cmd.exe - guiServer.exe
Initial Options
Microcontroller Connection
Map File
Localization
Start Experiment

```
MENU
1 - Obstacle Avoidance Enabled
2 - Obstacel Avoidance Disabled
3 - Sonar Test
1
Obstacle avoidance enabled
No points or lines in map!
ArNetServer started on tcp port 7272 and udp port 7272.
Connecting to simulator through tcp.

Syncing 0
Syncing 1
Syncing 2
Connected to robot.
Name: MobileSim
Type: Pioneer
Subtype: powerbot
Loaded robot parameters from powerbot.p
ArSick: Connected to simulated laser.
Mode stop added as default mode
Initializing log from config
ArLog::init:  StdOut     Normal  Not logging time        Not also printing
Loaded map file 'uos_1b88-sim.map'

Directory for maps and file serving: C:\Program Files\MobileRobots\Arnl\examples
\
See the ARNL README.txt for more information

Warning: Task 'Packet Handler' took 1095 ms to run (longer than the 250 warning
time)
Checking default on stop mode
Activated stop mode as default
Warning: ArRobot sync tasks too long at 1121 ms, (100 ms normal 250 ms warning)
Start localization at 0 0 0.0
Localized robot (using initial position of 0 0 0.0) to 0 0 0.0 with score 0.9944
75 (min 0.2).
To exit, press escape.
press enter to continue
```

Figure 25: Command line interface of guiServer.

3.2.3: MobileEyes

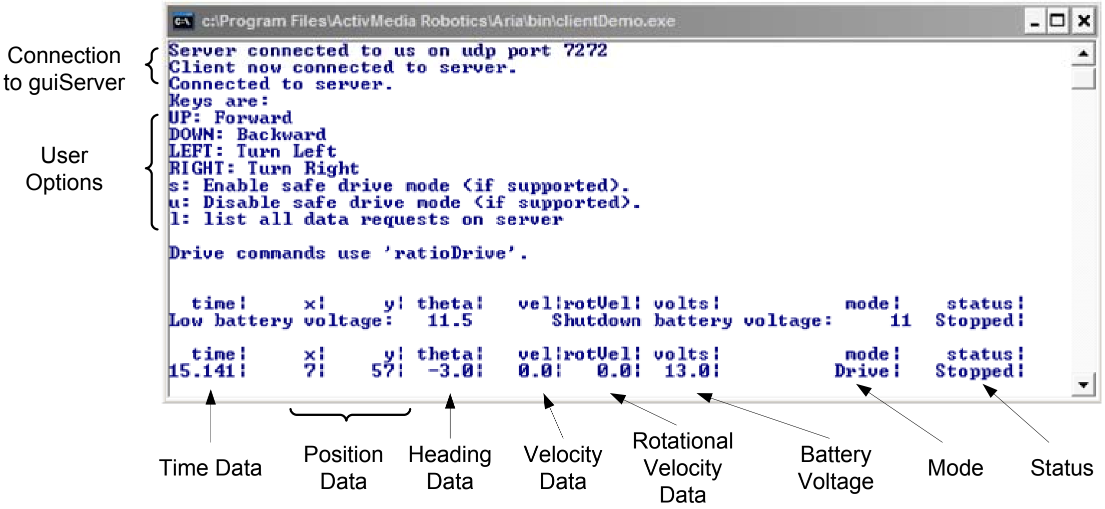
MobileEyes is a powerful and easy to use graphical interface for guiServer. In this project it is used for map creation and localization for the experiments. Localization, which means the location of the robot in the map is initially set, is a very important part of this project. The resulting robot trajectory is highly dependent on the initial starting point. MobileEyes provides an accurate way to localize (set the position and orientation) of the robot.

3.2.4: Mapper3

Mapper3 is used for editing and creating maps. Refer to chapter 3.3 for more information.

3.2.5: clientDemo

clientDemo is an ARIA library example program which was modified to be used in this project by giving it the capability of writing data into a file. clientDemo communicates with guiServer and obtains experimental information such as the time, absolute position data, velocity, heading, etc. This information is written to a file and is fed back to the CorridorSim program for analysis.

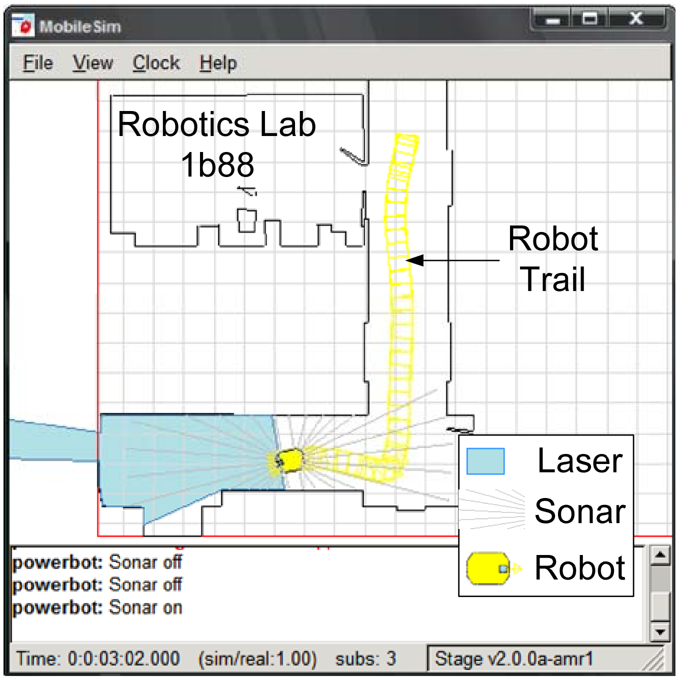


```
Server connected to us on udp port 7272
Client now connected to server.
Connected to server.
Keys are:
UP: Forward
DOWN: Backward
LEFT: Turn Left
RIGHT: Turn Right
s: Enable safe drive mode (if supported).
u: Disable safe drive mode (if supported).
l: list all data requests on server

Drive commands use 'ratioDrive'.

  time|     x|     y| theta|   vel|rotVel| volts|           mode|    status|
Low battery voltage:   11.5       Shutdown battery voltage:     11  Stopped|

  time|     x|     y| theta|   vel|rotVel| volts|           mode|    status|
15.141|     7|    57|  -3.0|   0.0|   0.0|  13.0|          Drive|   Stopped|
```

Figure 26: Command line interface of clientDemo program showing raw data that is output.

3.2.6: MobileSim

MobileSim is a PowerBot simulator provided by the manufacturer. It was used for the development stage since the code developed can be tested immediately without implementing on the robot; however, it is not accurate, and therefore is not used as the main simulator for this project. Figure 27 displays a typical path using this program.



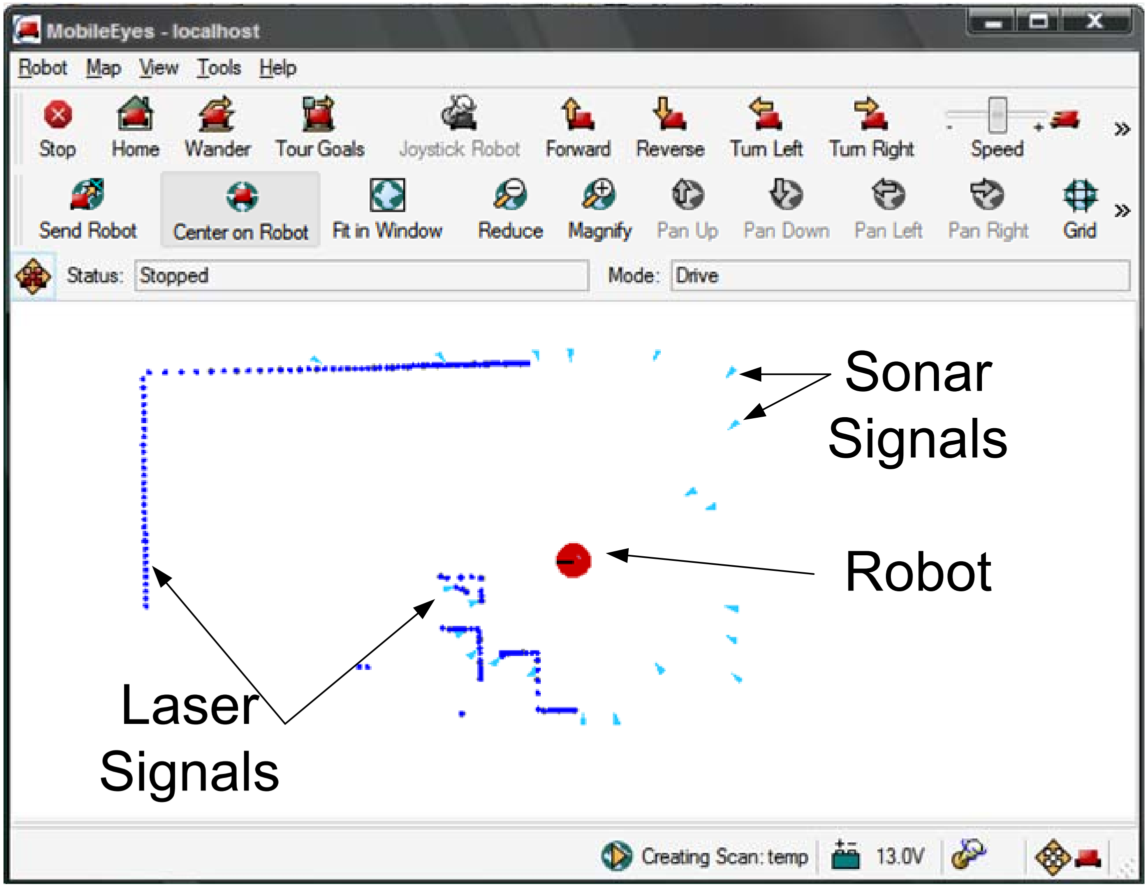
```
powerbot: Sonar off
powerbot: Sonar off
powerbot: Sonar on
```

```
Time: 0:0:03:02.000   (sim/real:1.00)   subs: 3   Stage v2.0.0a-amr1
```

Figure 27: MobileSim simulator. Shows the PowerBot robot navigating a path.

3.3: Map Creation

Before any experimentation could be completed, a detailed map for the setting of the robot had to be created. Map creation was accomplished by the Mapper3 and MobileEyes programs and by using the mobile robot's laser rangefinder and sonar systems. The laser's range is up to 50 m and its field of view is 180 degrees and sonar array provides a range of 6 m around the robot. Together, these two systems allow for the creation of detailed and accurate maps using the MobileEyes and Mapper3 software [28]. Figure 28 displays MobileEyes being used to create a map of Robotics Lab 1b88.



Status: Stopped

Mode: Drive

Creating Scan: temp

13.0V

Figure 28: MobileEyes used in scanning a 2D map of Robotics Lab 1b88.

After MobileEyes has scanned the area, the Mapper3 program is used to convert the scan into a map file as is displayed in Figure 29. Since this system only scans in a 2D environment, changes in elevation where ramps exist had to be incorporated manually; that is by physically measuring and entering the data.

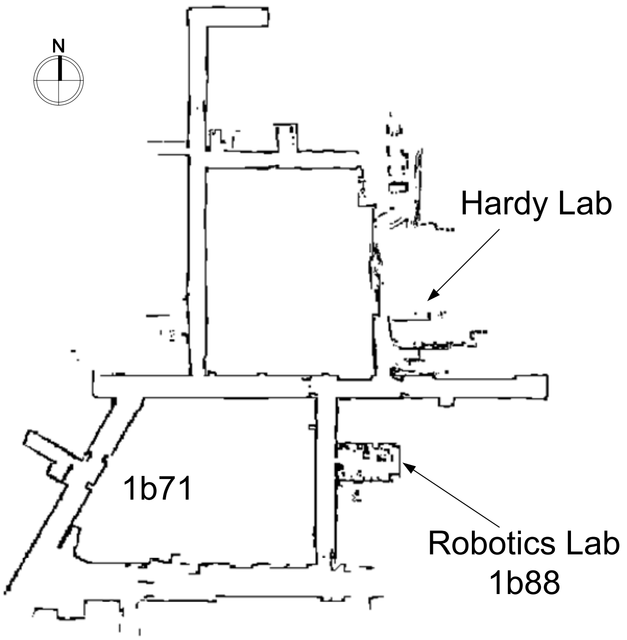
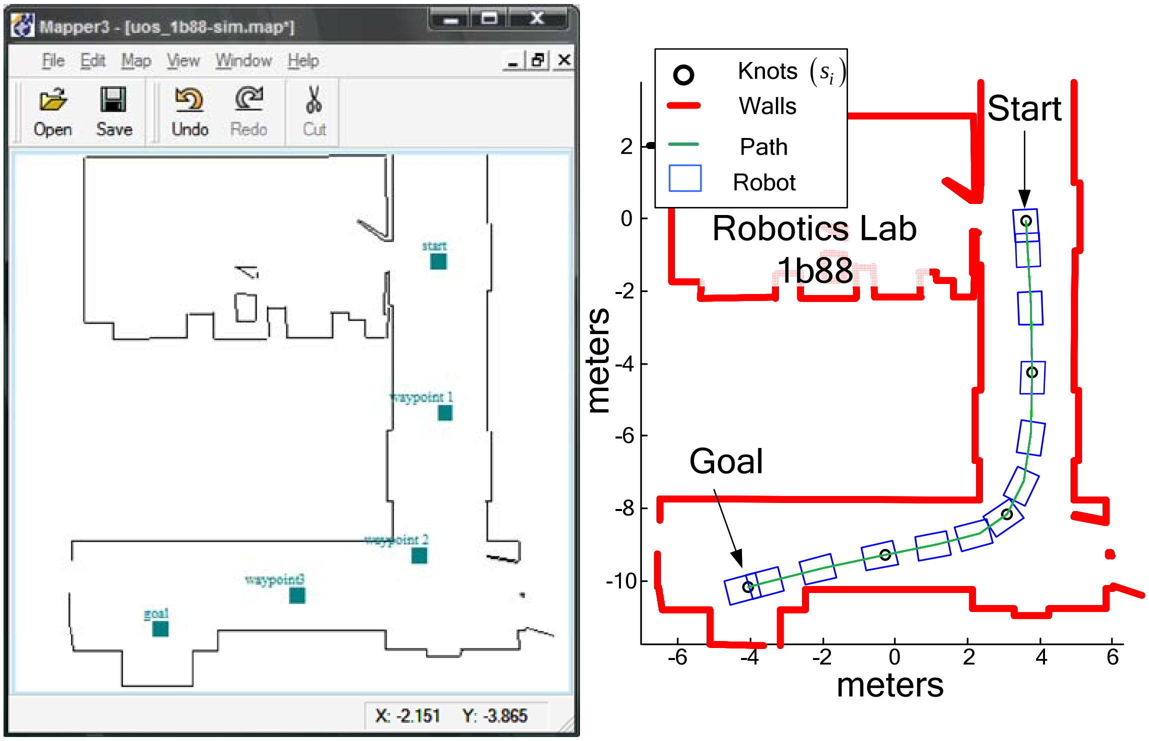


Figure 29: Mapper3 used to create a map of a section of the first floor of the U of S Engineering Building.

3.4: Trajectory Generation

Once a map has been created, a set of knots (as described in Section 2.1) are entered into the map file, created in Section 3.3, by using the Mapper3 program. The map file is then imported into the custom MATLAB program and a path is created using cubic splines. See Figure 30 for an example of a generated path.



X: -2.151 Y: -3.865

Figure 30: Path of experiment 2 (see Section 4.2). Knots in Mapper3 (left). Path created in MATLAB (right).

If a predefined velocity profile is not provided, a velocity profile is created, as outlined in Section 2.3.

3.5: Implementation of the Velocity Profile on the Robot.

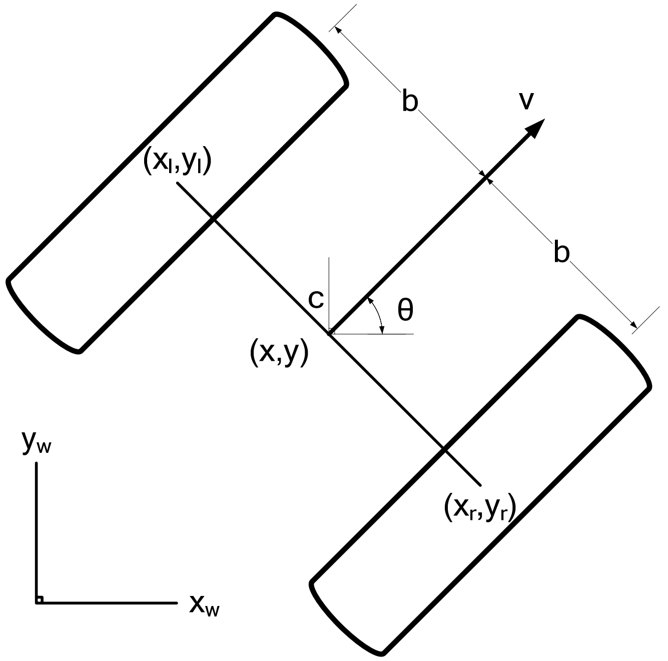


Figure 31: Top view of driving wheels of the robot.

In order to implement the velocity profile on the robot, the linear velocity of the center of the robot must be related to the wheels of the robot. As shown in Figure 31, the position of the right wheel can be expressed in terms of the center of the driving wheels as shown:

$$\begin{aligned} x_r &= x + b\sin(\theta) \\ y_r &= y - b\cos(\theta). \end{aligned} \tag{3.1}$$

where (x, y) is the position of point c, (x_r, y_r) is the position of the right wheel, b is the distance from the center point, c, to the center of each wheel, and θ is the orientation of the robot with respect to the fixed coordinates (x_w, y_w). Similarly the position of the center of the left wheel can be expressed as:

$$\begin{aligned} x_l &= x - b\sin(\theta) \\ y_l &= y + b\cos(\theta). \end{aligned} \tag{3.2}$$

The velocity of the center point, v, of the driving wheels can be expressed as:

$$v = \dot{x}\cos(\theta) + \dot{y}\sin(\theta) \tag{3.3}$$

Therefore, the velocity of the left and right wheels, respectively, can be expressed as:

$$\begin{aligned} v_l &= \dot{x}_l \cos(\theta) + \dot{y}_l \sin(\theta) \\ v_r &= \dot{x}_r \cos(\theta) + \dot{y}_r \sin(\theta) \end{aligned} \tag{3.4}$$

Substituting equations (3.1), (3.2) into (3.4) and simplifying results in:

$$\begin{aligned} v_l &= -b\dot{\theta} + \underbrace{\dot{x}\cos\left(\theta\right) + \dot{y}\sin\left(\theta\right)}_{v} \\ v_r &= b\dot{\theta} + \underbrace{\dot{x}\cos\left(\theta\right) + \dot{y}\sin\left(\theta\right)}_{v} \end{aligned} \tag{3.5}$$

Equation (3.5) can be further simplified by using equation (3.3):

$$\begin{aligned} v_l &= -b\dot{\theta} + v \\ v_r &= b\dot{\theta} + v \end{aligned} \tag{3.6}$$

where v_L and v_R are the left and right linear wheel velocities respectively, and $\dot{\theta}$ is the rate of change of the robot's orientation with respect to time.

Since the velocity is a continuous function it cannot be directly given to the robot, as it requires time for computation and therefore must be digitized. Let:

$$M_{\text{int}} \cong \frac{t_i}{T_s}, \tag{3.7}$$

where M_{int} is the number of intervals in segment i, and is rounded to the nearest positive integer, t_i is the time required for the robot to travel segment i, and T_s is the sampling rate for providing a discrete velocity to the robot. The velocity can then be shown as:

$$V_{ik} = v_i(kT_s), \tag{3.8}$$

where V_{ik} is the discrete velocity for each segment i, and each interval k, with

$k = 0,1,2,\ldots,M_{\text{int}}$. The discrete time can be expressed as:

$$T_{ik} = t_i\left(kT_s\right), \tag{3.9}$$

where T_{ik} is the discrete time for each segment i, and each interval k.

The sample rate T_s, was determined experimentally by running a series of tests using the same velocity profile. The criteria used in choosing the ideal sample rate were how well the robot followed the velocity profile and the path. The path chosen for this experiment is the path for experiment 2, shown in Figure 30. The simulated velocity profile for this path is shown in Figure 32. To compare the differences of the experimental and simulated velocity profile of the robot, the following Mean Square Error Norm (MSEN) was used:

$$\|E\| = \sqrt{\frac{1}{Q}\sum_{i=1}^{Q}\left(v_{d_i} - v_{e_i}\right)^2} \tag{3.10}$$

where E is the mean squared error, Q is the total number of data points returned by the robot, v_{d_i} is the simulated velocity, and v_{e_i} is the experimental velocity of the driving wheels of the robot. Table 1 shows the results of four different sample rates, T_s. To compare the final destination of the robot at the end of its maneuver the following distance relation was used:

$$d = \sqrt{\left(x_e - x_d\right)^2 + \left(y_e - y_d\right)^2 + \left(z_e - z_d\right)^2} \tag{3.11}$$

where $\left(x_e, y_e, z_e\right)$ is the destination point of the experimental path and $\left(x_d, y_d, z_d\right)$ is the destination point of the simulated path. These results are shown in Table 2. As can be seen from these tables, a sample rate of about $T_s \cong 0.5$ seconds is the optimum sampling time for updating the robots velocity. A sample rate of less than 0.5 seconds resulted in the robot’s inability to keep up with the computations required, while a rate of greater than 0.5 seconds added considerable error into the system.

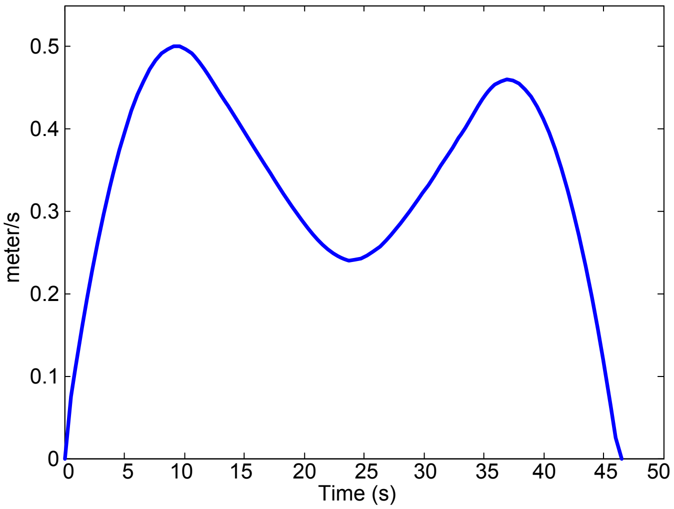


Figure 32: Desired velocity profile for the path shown in Figure 30.

Table 1: Sample rate and the corresponding mean squared error norms.

Sample Time (seconds)	**MSEN from Eq. (3.10) (m/s)**
0.3	0.014
0.4	0.022
0.5	**0.010**
0.6	0.014

Table 2: Sample rate and the corresponding distance values.

Sample Time (seconds)	**Error in Final Destination (m)**
0.3	1.60
0.4	1.68
0.5	**1.39**
0.6	1.77

3.6: Moving Obstacles Avoidance

If the static obstacles are avoided in the path planning stage, the robot should have the ability to follow the path without any collisions. However, a moving obstacle cannot be accounted for in the path planning stage without prior knowledge of the obstacles in the path. The proposed strategy modifies the velocity profile in real time, with no changes made to the path itself [15]. In the presence of moving obstacles, the velocity profile is modified by reducing the velocity of the robot, and stopping if necessary, until the obstacle has passed. The robot's previous velocity, before detecting the obstacle, is resumed, taking into account the distance traveled during the reduction of the velocity.

The PowerBot robot is equipped with sonar sensors that allow for the detection of obstacles. The sensors are set to detect obstacles at 60 cm away with a scanning time equal to the sampling time of roughly 0.5 seconds If a moving obstacle is detected, the time interval is immediately halved and the velocity is reduced to half of its value in the second half of the time interval. If the obstacle has not passed within the time interval, the robot will eventually stop. Once the obstacle has passed (more than 60 cm from the robot), the robot velocity is increased to its original value incrementally as shown in Figure 33.

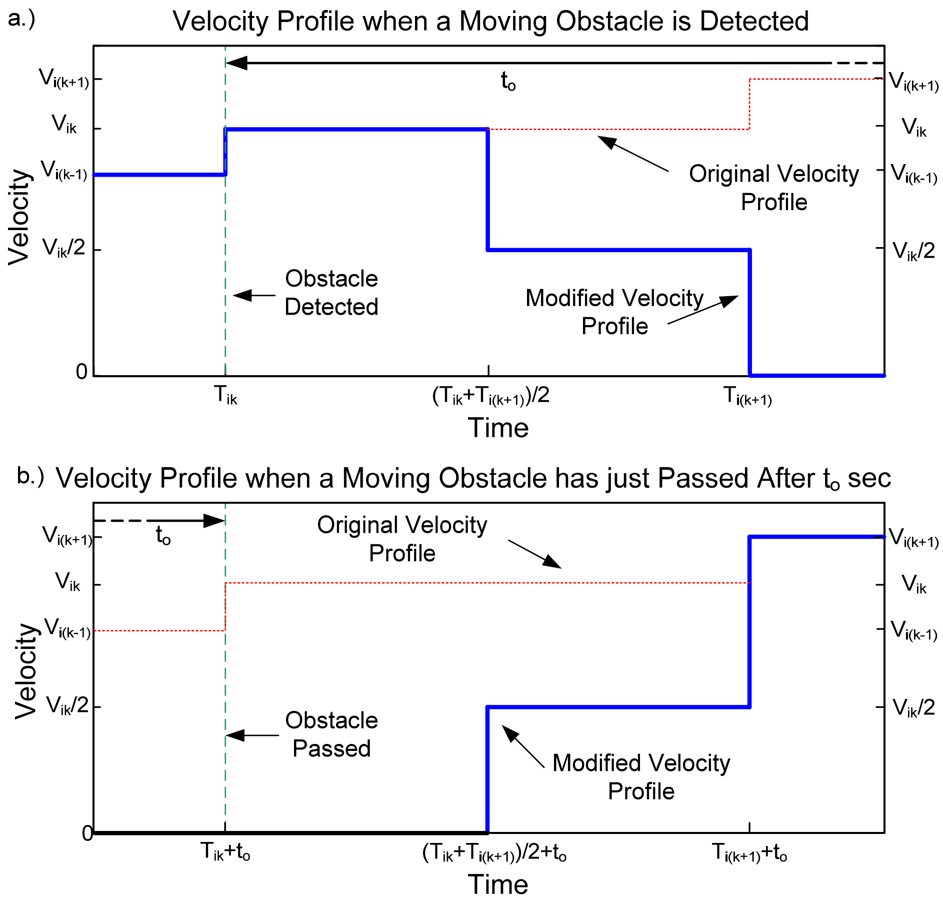


Figure 33: Modified velocity profile when a moving obstacle is detected.

It can be shown that the distance traveled by the robot, when no obstacle is present, is the same as the distance traveled after modifying the velocity profile in the case of a moving obstacle detected. The distance traveled with the original velocity profile is shown below:

$$D_{ik} = V_{ik}\left(T_{i(k+1)} - T_{ik}\right), \tag{3.12}$$

where D_{ik} is the distance traveled on the i^{th} segment and k^{th} interval, V_{ik} is the velocity, and $T_{i(k+1)}$ is the starting time of the next interval. Using the modified velocity profile, this distance can be calculated as follows:

$$\begin{aligned} D_{ik} &= V_{ik}\left(\frac{\left(T_{ik}+T_{i(k+1)}\right)}{2}-T_{ik}\right)+\frac{V_{ik}}{2}\left(T_{i(k+1)}-\frac{\left(T_{ik}+T_{i(k+1)}\right)}{2}\right) \\ &\quad+\frac{V_{ik}}{2}\left(\left(T_{i(k+1)}+t_o\right)-\left(\frac{\left(T_{ik}+T_{i(k+1)}\right)}{2}+t_o\right)\right) \\ &= V_{ik}\left(T_{i(k+1)}-T_{ik}\right), \end{aligned} \tag{3.13}$$

where t_o is the time the obstacle was detected until it has cleared the robot's path.

CHAPTER 4: SIMULATION AND EXPERIMENTAL RESULTS

A total of six experimental and simulation case studies have been conducted in an attempt to establish validity of the proposed temporal planning algorithm. For each case study, a brief outline, the physical parameters, and the results are provided. The physical parameters for all cases are as follows:

- Path.

The start point, in experimentation, is set by using the MobileEyes software (see Section 3.2.3), while the goal point is obtained by integrating the experimental velocity profile and using the experimental orientation data. MobileEyes cannot be used to locate the goal point because it is unable to correctly localize the robot when the robot travels on a ramp.

- Velocity profile and orientation.

The experimental velocity profile and orientation are measured by using the clientDemo program (see Section 3.2.5). The PowerBot robot contains high resolution encoders on each driveshaft that allow for a high precision speed sensing and direction [25].

- Distance traveled.

The distance traveled, for both simulation and experimentation, is obtained by integrating the experimental and simulated velocity profiles with respect to the time of motion.

- Time of motion.

The time of motion is measured by using the onboard computer. This provides accuracy of up to 0.0001 seconds.

Since a large amount of data was collected, only one set of graphs for each experiment were presented. The minimum error is the main criterion for choosing the graphing data. In this project, MSE norm (mean squared error norm) of the velocity, orientation, and change in orientation with respect to time is the main method of quantifying the error. The distance, orientation, and time of motion between the experimental and desired goal points are also compared.

4.1: Experiment 1

Experiment 1 contains a path with two turns and is located on the main hallway north of the robotics lab (1B88). The simulated and experimental paths for this experiment are shown in Figure 34. As can be seen from this figure, this path is composed of 5 knots. For this experiment it was determined that $M_{\text{int}} = 14$ intervals between each knot resulted in a sample rate, $T_s = 0.510$ seconds, which was acceptable as compared to the optimum value of 0.5 seconds.

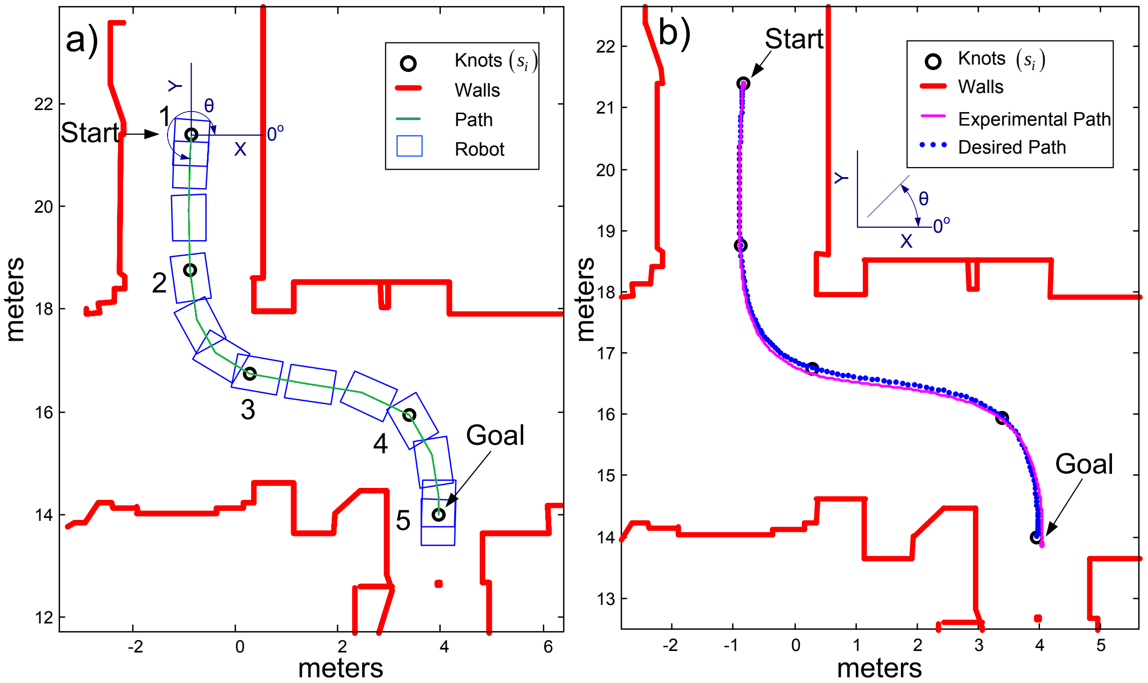


Figure 34: Experiment 1, path located in the hallway north of the Robotics Lab, 1B88. a) Simulated (desired). b) Experimental.

Three tests were completed for this experiment. Test 2 had the lowest difference in simulated and desired destination point of 0.050 m, as can be seen from Table 5, and therefore its results were graphed. Table 3 shows a number of physical parameters for each test and the simulated results.

Table 3: Experiment 1, physical parameters.

Test	1	2	3	Simulated
Start Point (x,y,z) (m)*	(-0.840,21.4,0)	(-0.886,21.4,0)	(-0.86,21.4,0)	(-0.840,21.4,0)
Goal Point (x,y,z) (m)	(3.77,13.9,0)	(3.99,14.0,0)	(4.08,13.9,0)	(3.96,14.0,0)
Distance Travelled (m)	10.3	10.3	10.3	10.5
Time of Motion (s)	28.2	28.1	28.1	28.5
Initial Orientation (deg)*	-93.0	-93.0	-94.0	-93.8
Final Orientation (deg)	-99.0	-96.0	-100	-94.7

* Initially specified by the user. All others are experimental results (tests 1 to 3)

An arbitrary time of motion of 4 seconds was assigned to the path to convert it into a rudimentary trajectory. This rudimentary trajectory was made feasible by using the mobile robot constraint limits and stretching the time of motion to 28.5 seconds (refer to Section 2.3). The constraint limits are shown in Table 4. In that table v_{robot}, v_{slide}, and

v_{safety} are originally defined in equations (2.6), (2.15), and (2.27) respectively, a_{robot}, a_{tip}, and a_{slip} are given in equations (2.7), (2.24), and (2.20) respectively, and d_{robot}, d_{tip}, and d_{slip} are provided in equations (2.8), (2.25), and (2.21) respectively. Figure 35 displays the scaled acceleration and velocity profiles along with the velocity, acceleration, and deceleration constraints.

Table 4: Experiment 1, velocity, acceleration and deceleration limits.

Velocity Limits (m/s)		Acceleration Limits (m/s^2)		Deceleration Limits (m/s^2)	
v_{safety}	0.500	a_{robot}	0.550	d_{robot}	-7.80
v_{robot}	2.10	a_{tip}	6.02	d_{tip}	-6.02
v_{slide}	1.99	a_{slip}	1.70	d_{slip}	-1.70
V_{lim}	**0.500**	**A_{lim}**	**0.550**	**D_{lim}**	**-1.70**

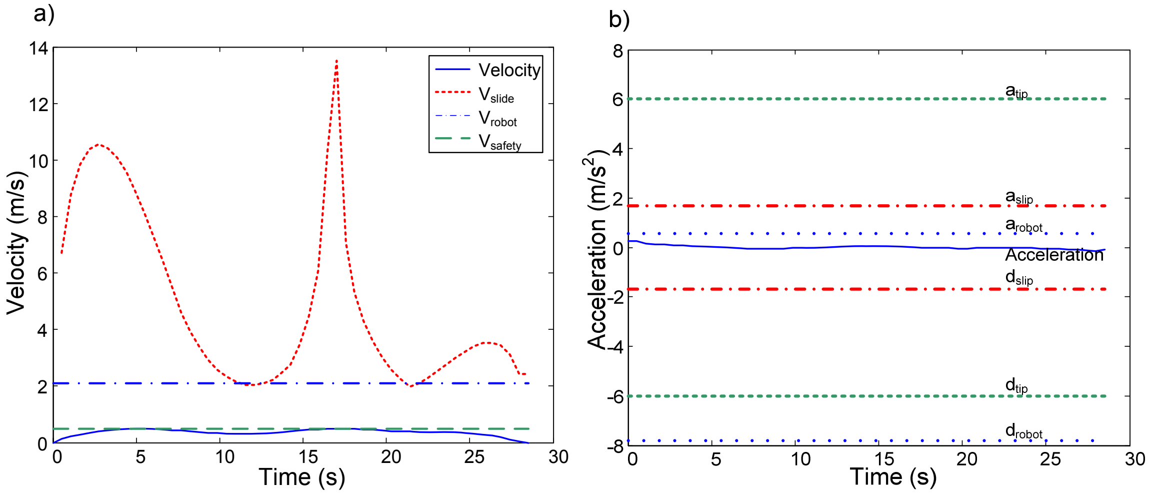


Figure 35: Experiment 1, a) Robot velocity – time with constraints. b) Robot acceleration – time with constraints.

The un-scaled and scaled velocity, acceleration, and deceleration profiles are shown in Figure 36. The left-hand side of Figure 36 shows a total maneuver time of 4.00 seconds, maximum velocity of 3.57 m/s, maximum acceleration of 12.7 m/s^2, and a maximum deceleration of -7.10 m/s^2. All three values are beyond the robot limits shown in Table 4. As it can be observed, on the right-hand side of Figure 36, setting the maximum velocity to V_{lim}=0.500 m/s by increasing the time of motion to 28.5 seconds, resulted in a maximum acceleration of 0.250 m/s^2, and a maximum deceleration of -0.140 m/s^2. The acceleration is well below the limit of A_{lim}=0.550 m/s^2 and the deceleration is above the limit of D_{lim}=-1.70 m/s^2.

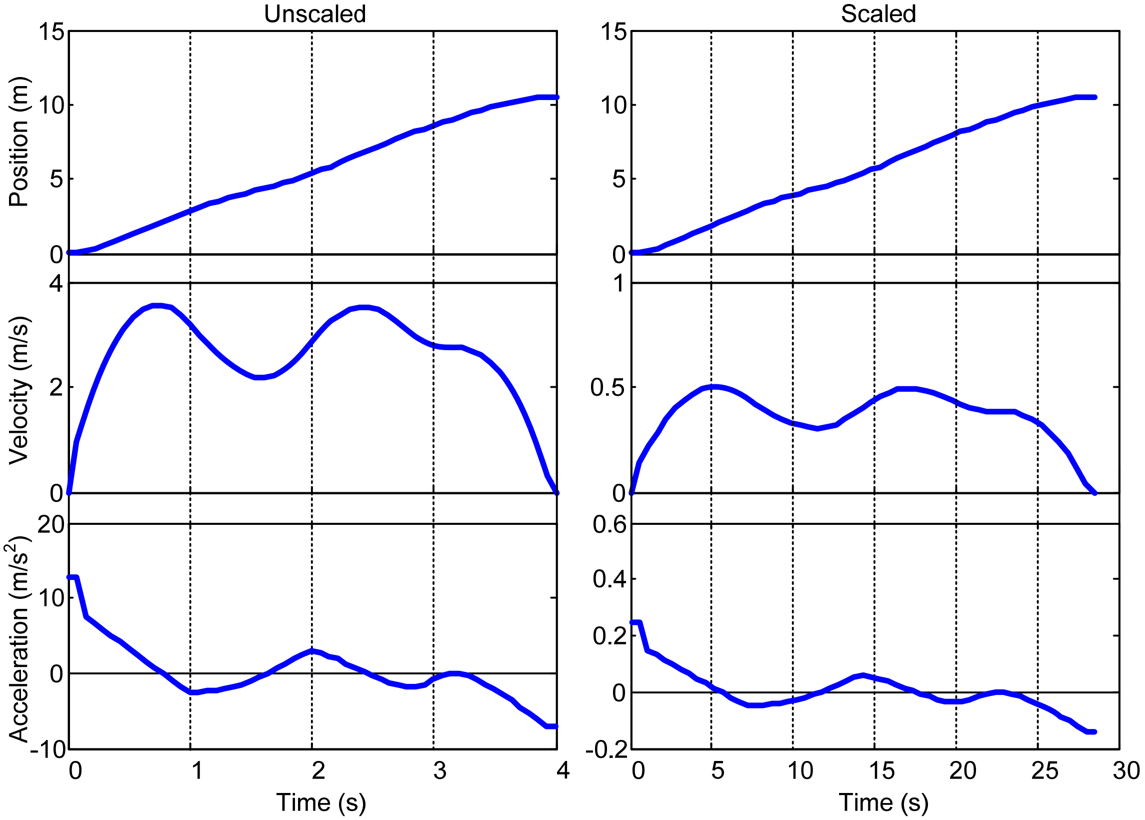


Figure 36: Experiment 1, simulated position, velocity, and acceleration. Un-scaled (left) and scaled (right).

Once a feasible velocity profile was created in the simulation stage, it is necessary to calculate the left and right wheel velocities in order to implement the profile on the robot. Figure 37 displays the linear velocities of both the robot and the wheels.

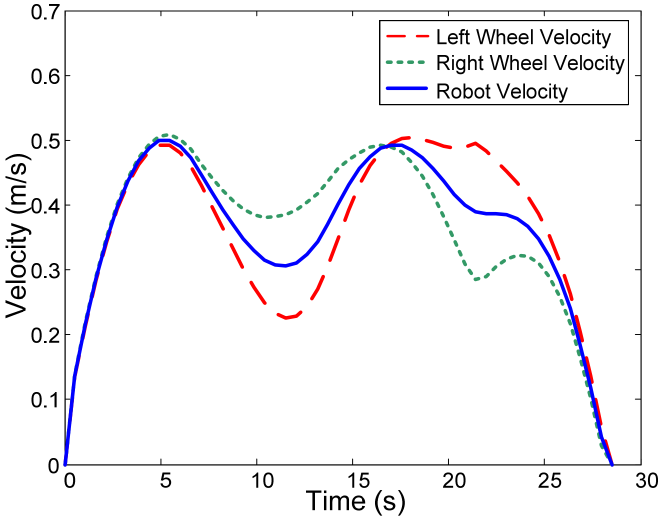


Figure 37: Experiment 1, linear velocities of the center of the robot and the center of the left and right wheels.

Figure 38 shows the experimental and simulated (desired) velocity profiles with respect to time and distance. As can be seen from the figure, the experimental and desired velocity profiles are very close, having a mean squared error of only $\|E\| = 0.028$ m/s as calculated from equation (3.10). Figure 39 shows the experimental and desired orientation. The experimental and simulated paths were shown in Figure 34.

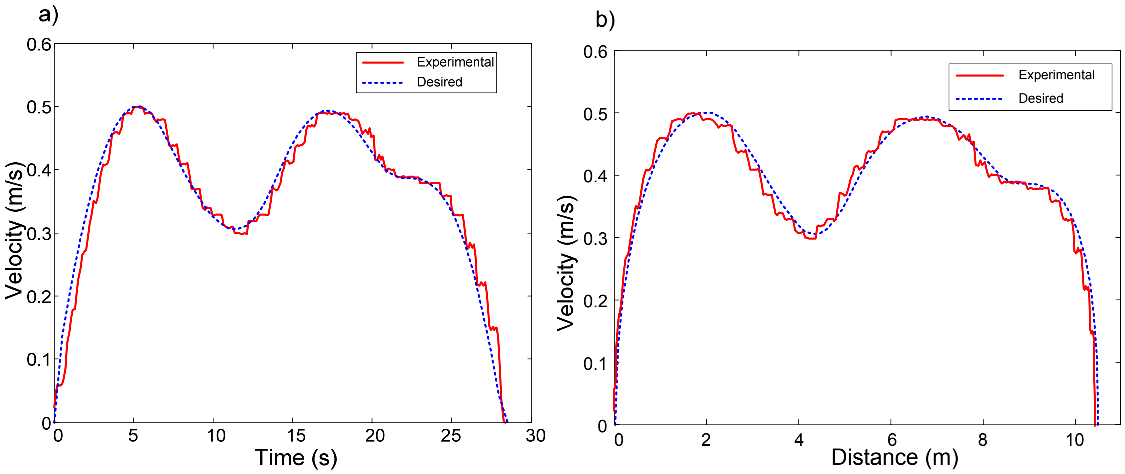


Figure 38: Experiment 1, experimental and simulated (desired) linear robot velocities. a) Versus time. b) Versus distance.

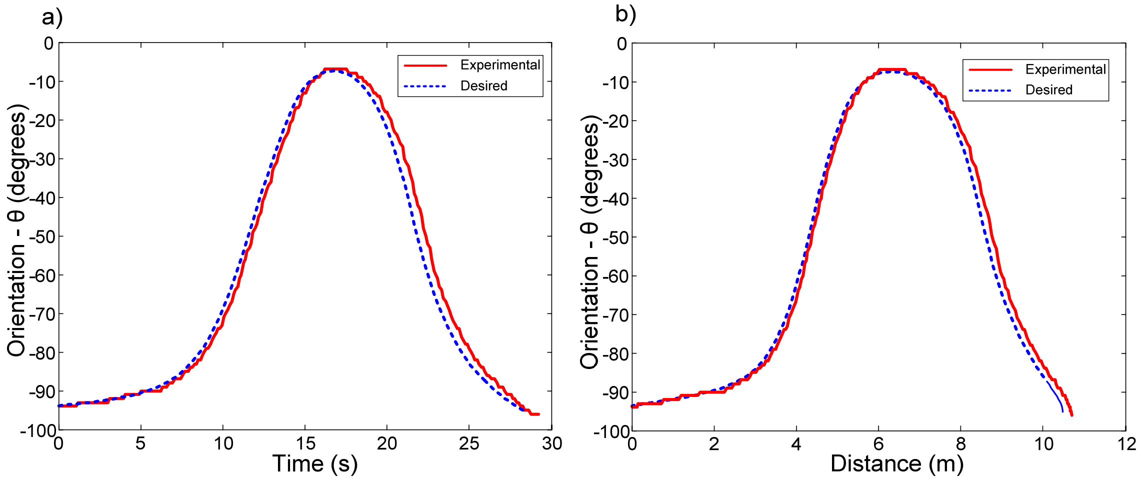


Figure 39: Experiment 1, experimental and simulated (desired) orientation. a) Versus time. b) Versus distance.

Table 5 displays the difference between the experimental and simulated (desired) results of experiment 1. In all cases, the velocity profiles had an error of around 0.02 – 0.03 m/s which is relatively low as compared to the maximum velocity of the path (0.5 m/s). The orientation in test 1, test 2, and test 3 had a MSE norm of 3.45°, 3.31°, and 4.95° respectively, and a difference in goal position of 0.214 m, 0.050 m, and 0.394 m respectively. From this data it appears that the orientation has a great impact on the ability of the robot to track the path.

Table 5: Experiment 1, difference in experimental and simulated (desired) results.

Test	**1**	**2**	**3**
MSE of Velocity Profile (m/s)	0.024	0.028	0.032
MSE of Orienation (deg)	3.45	3.31	4.95
Difference in E and S* Start Position (m)	0.025	0.046	0.016
Difference in E and S* Goal Position (m)	**0.214**	**0.050**	**0.394**
Difference in E and S* Time (s)	0.305	0.414	0.443
Difference in E and S* Initial Orientation (deg)	0.755	0.755	0.245
Difference in E and S* Final Orientation (deg)	4.35	1.35	5.35

* Experimental and Simulated

4.2: Experiment 2

The path for experiment 2 starts from the outside door of the Robotics Lab (1B88) and ends at the entrance door of the Hardy lab. It contains one turn and has a traveling distance of 15.8 m. This experiment is also used for moving obstacle avoidance as discussed in experiment 3. The simulated (desired) and experimental paths for this experiment are shown in Figure 40. As can be seen from this figure, this path is composed of 5 knots. For this experiment it was determined that $M_{\text{int}} = 23$ intervals between each knot resulted in a sample rate, $T_s = 0.506$ seconds, which was acceptable when compared to the optimum value of 0.5 seconds.

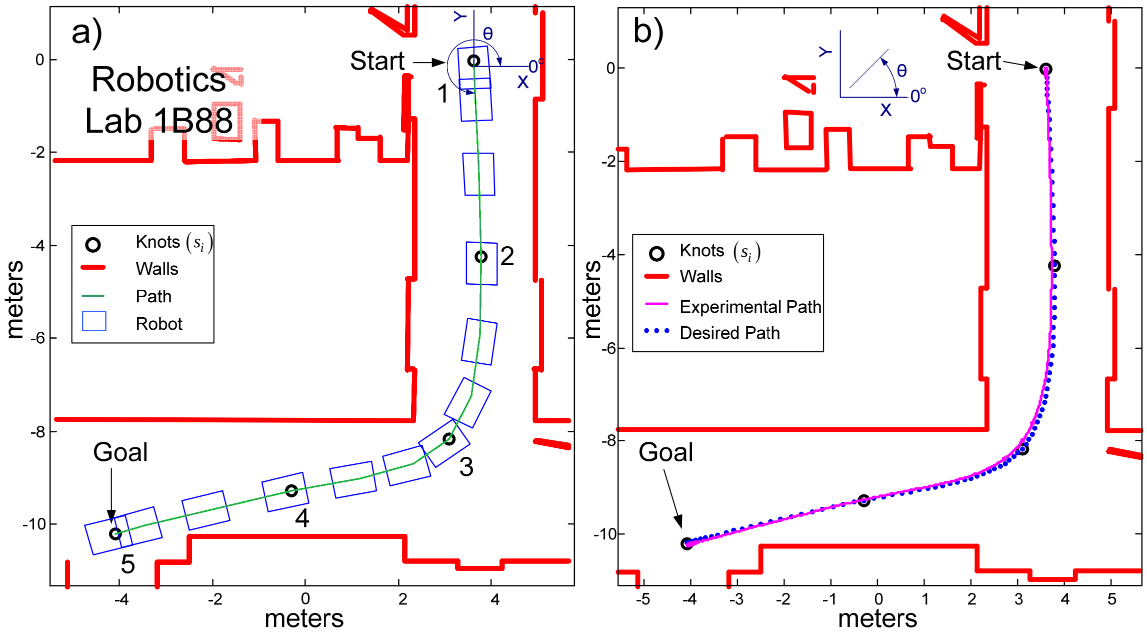


Figure 40: Experiment 2, path located in the hallway outside the Robotics Lab. a) Simulated (desired). b) Experimental.

Three tests were completed for this experiment. Test 3 had the lowest difference in experimental and simulated destination position of 0.030 m, as can be seen from Table

8, and therefore its results were used. Table 6 shows a number of physical parameters for each test and the simulated results.

Table 6: Experiment 2, physical parameters.

Test	1	2	3	Simulated
Start Point (x,y,z) (m)*	(3.61,-0.022,0)	(3.61,-0.053,0)	(3.61,-0.033,0)	(3.62,-0.037,0)
Goal Point (x,y,z) (m)	(-4.02,-10.3,0)	(-3.92,-10.4,0)	(-4.09,-10.2)	(-4.07,-10.2,0)
Distance Travelled (m)	15.8	15.8	15.8	15.8
Time of Motion (s)	46.4	46.5	46.5	46.5
Initial Orientation (deg)*	-86.0	-86.0	-86.0	-86.3
Final Orientation (deg)	-165	-164	-166	-165

* Initially specified by the user. All others are experimental results (tests 1 to 3)

An arbitrary time of motion of 4 seconds was assigned to the path to convert it into a rudimentary trajectory. This rudimentary trajectory was made feasible by using the mobile robot constraint limits and stretching the time of motion to 46.5 seconds (refer to Section 2.3). The constraint limits are shown in Table 7. Figure 41 displays the scaled acceleration and velocity profiles, along with the velocity, acceleration and deceleration constraints.

Table 7: Experiment 2, velocity, acceleration and deceleration limits.

Velocity Limits (m/s)		Acceleration Limits (m/s^2)		Deceleration Limits (m/s^2)	
v_{safety}	0.500	a_{robot}	0.550	d_{robot}	-7.80
v_{robot}	2.10	a_{tip}	6.02	d_{tip}	-6.02
v_{slide}	2.31	a_{slip}	1.70	d_{slip}	-1.70
V_{lim}	**0.500**	**A_{lim}**	**0.550**	**D_{lim}**	**-1.70**

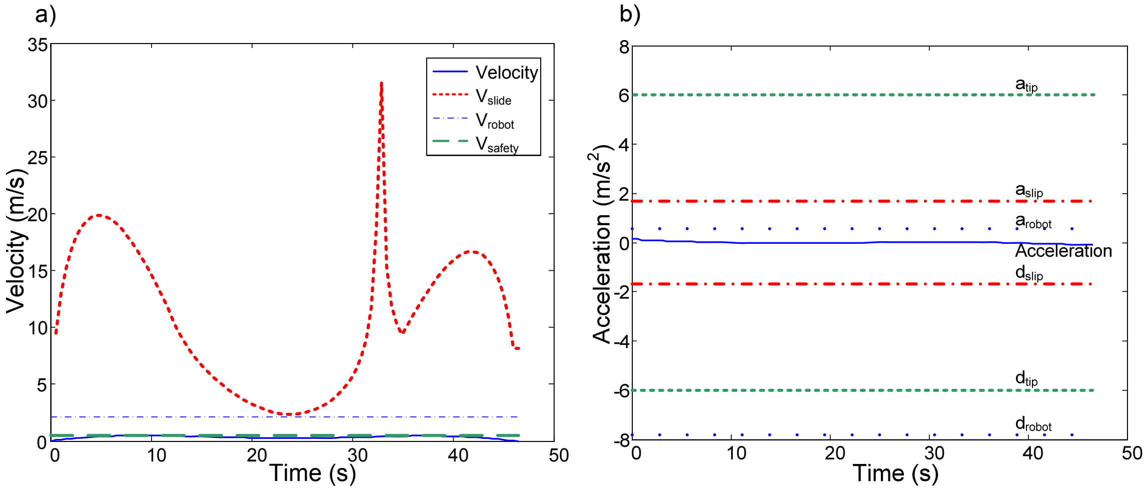


Figure 41: Experiment 2, a) Robot velocity – time with constraints. b) Robot acceleration – time with constraints

The un-scaled and scaled velocity, acceleration, and deceleration profiles are shown in Figure 42. The left-hand side of Figure 42 shows a total maneuver time of 4.00 seconds, maximum velocity of 5.82 m/s, maximum acceleration of 20.0 m/s^2, and a maximum deceleration of -12.6 m/s^2. All these values (velocity, acceleration, and deceleration) are beyond the robot limits as shown in Table 7. As it can be observed, on the right-hand side of Figure 42, setting the maximum velocity to V_{lim}=0.500 m/s by increasing the time of motion to 46.5 seconds, resulted in a maximum acceleration of 0.145 m/s^2, and a maximum deceleration of -0.093 m/s^2. The acceleration is well below the limit of A_{lim}=0.550 m/s^2 and the deceleration is above the limit of D_{lim}=-1.70 m/s^2.

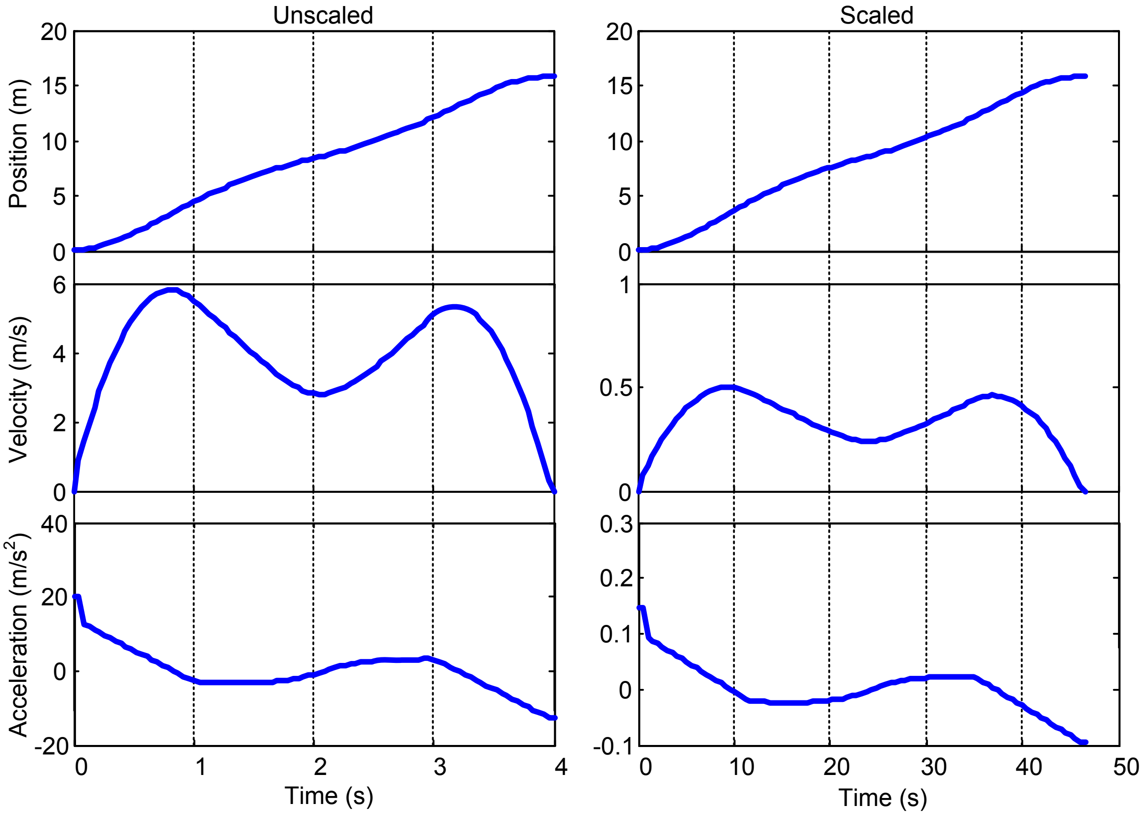


Figure 42: Experiment 2, simulated position, velocity, and acceleration. Un-scaled (left) and scaled (right).

Figure 43 displays the linear velocities, which were calculated to be implemented on the mobile robot, of both the robot and the wheels.

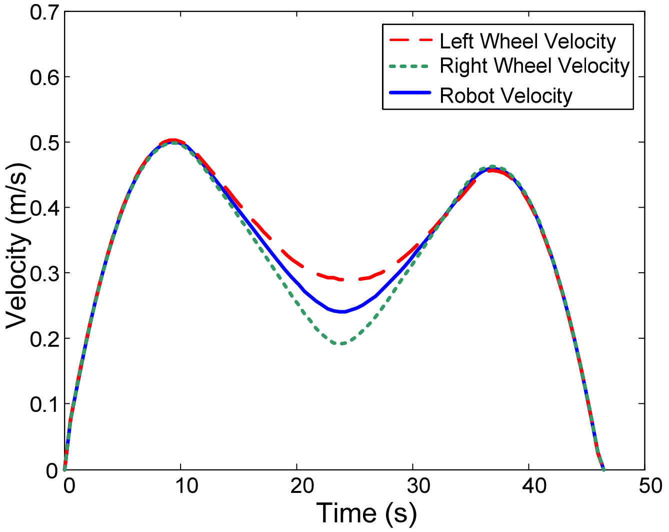


Figure 43: Experiment 2, linear velocities of the center of the robot and the center of the left and right wheels.

Figure 44 shows the experimental and simulated (desired) velocity profiles both in respect to time and distance. As can be seen from the figure, the experimental and desired velocity profiles are very close having a MSE of $\|E\| = 0.011$ m/s as calculated from equation (3.10). Figure 45 shows the experimental and desired orientation.

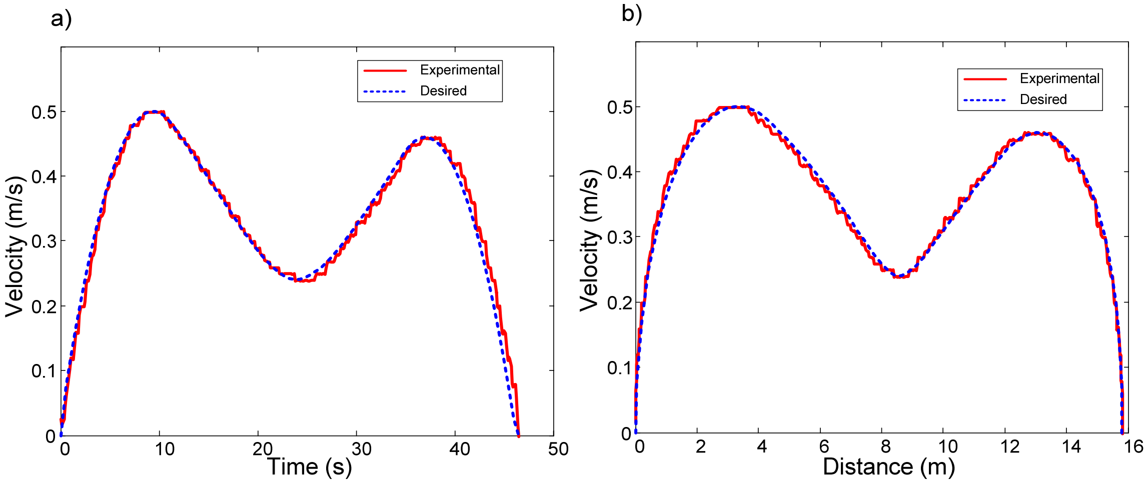


Figure 44: Experiment 2, experimental and simulated (desired) linear robot velocities. a) Versus time. b) Versus distance.

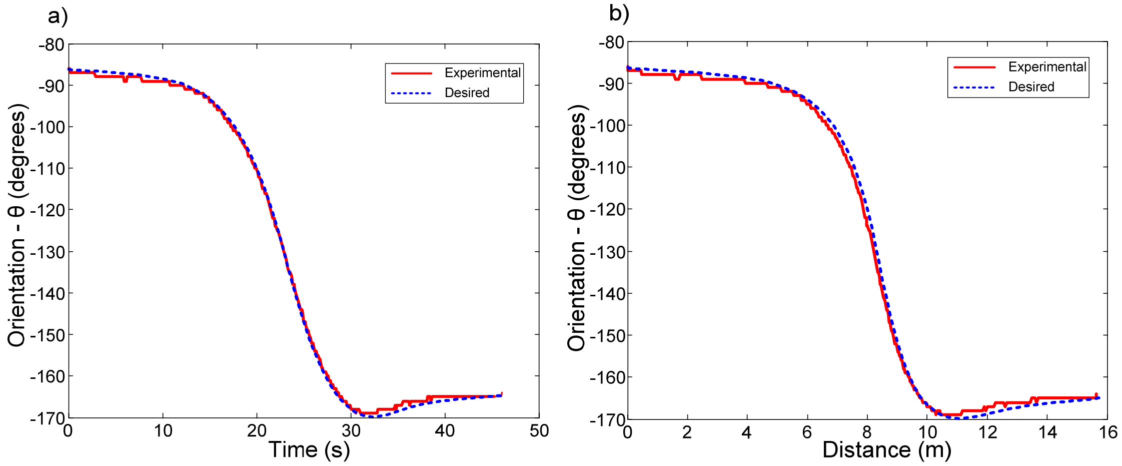


Figure 45: Experiment 2, experimental and simulated (desired) orientation. a) Versus time. b) Versus distance.

For all three tests performed, Table 8 shows the difference between the experimental and simulated results of experiment 2. In all cases, the velocity profiles had an error of around 0.01 m/s which is relatively low as compared to the maximum velocity of the path (0.5 m/s). The orientation in tests 1, 2, and 3 had a MSE norm of 0.903°, 1.22°, and 1.00 ° respectively, and a difference in goal position of 0.096 m, 0.221 m, and 0.030 m respectively. From this data it appears that both the orientation as well as the velocity profile have a large impact on the ability of the robot to track the path.

Table 8: Experiment 2, difference in experimental and simulated (desired) results.

Test	**1**	**2**	**3**
MSE of Velocity Profile (m/s)	0.013	0.012	0.011
MSE of Orienation (deg)	0.903	1.22	1.00
Difference in E and S* Start Position (m)	0.016	0.017	0.014
Difference in E and S* Goal Position (m)	**0.096**	**0.221**	**0.030**
Difference in E and S* Time (s)	0.162	0.035	0.014
Difference in E and S* Initial Orientation (deg)	0.295	0.295	0.295
Difference in E and S* Final Orientation (deg)	0.250	0.750	1.25

* Experimental and Simulated

4.3: Experiment 3

When a moving obstacle crosses the robot's path, the velocity profile is modified as discussed in Section 3.6. Experiment 3 is an implementation of this aspect of the developed algorithm. The path of this experiment is the same as the one in experiment 2, except that it has two obstacles crossing the path as shown in Figure 46.

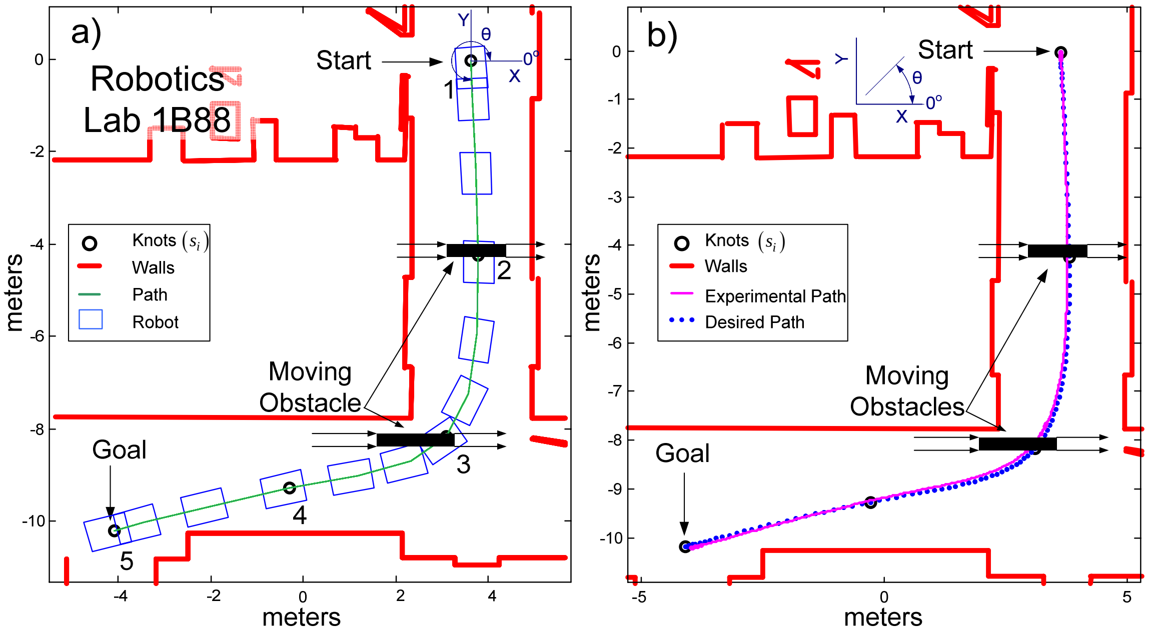


Figure 46: Experiment 3, two moving obstacles blocking the path of the robot, located in the hallway outside the Robotics Lab. a) Simulated (desired). b) Experimental.

One test was completed for this experiment since it is very difficult to synchronize the moving obstacle's movement with the mobile robot's movement. Table 9 shows a number of physical parameters for the test and the simulated results.

Table 9: Experiment 3, physical parameters.

Test	1	Simulated
Start Point (x,y,z) (m)*	(3.62,-0.038,0)	(3.62,-0.037,0)
Goal Point (x,y,z) (m)	(-3.96,-10.2,0)	(-4.07,-10.2,0)
Distance Travelled (m)	15.6	15.8
Time of Motion (s)	49.3	46.5
Initial Orientation (deg)*	-86.0	-86.3
Final Orientation (deg)	-164	-165

* Initially specified by the user. All others are experimental results (test 1)

See experiment 2 (Figure 41 to Figure 43) for the constraints, wheel velocities, scaled, and un-scaled velocity profiles. At $T_{ik} = 9.95$ seconds, an obstacle came into the robot's path and the velocity profile was modified in real time. It took $t_o = 0.59$ seconds for the obstacle to clear the robot's path and therefore the robot only slowed down and did not stop. At $T_{ik} = 21.2$ seconds, another obstacle came into the robot's path. It took $t_o = 2.74$ seconds for the obstacle to clear the path. Figure 47 shows the experimental and simulated (desired) velocity profiles both versus time and distance. Figure 48 shows the experimental and desired orientation.

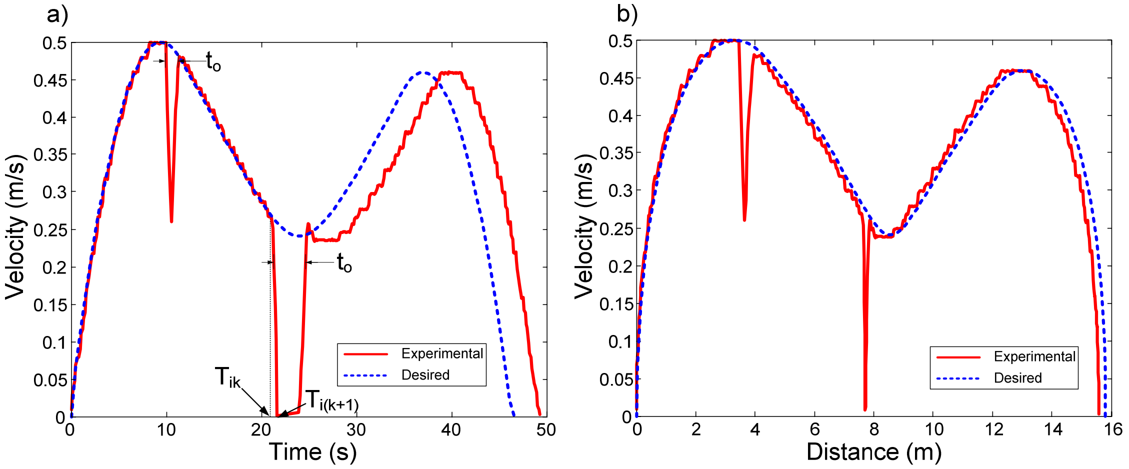


Figure 47: Experiment 3, experimental and simulated (desired) linear robot velocities with moving obstacles present. a) Versus time. b) Versus distance.

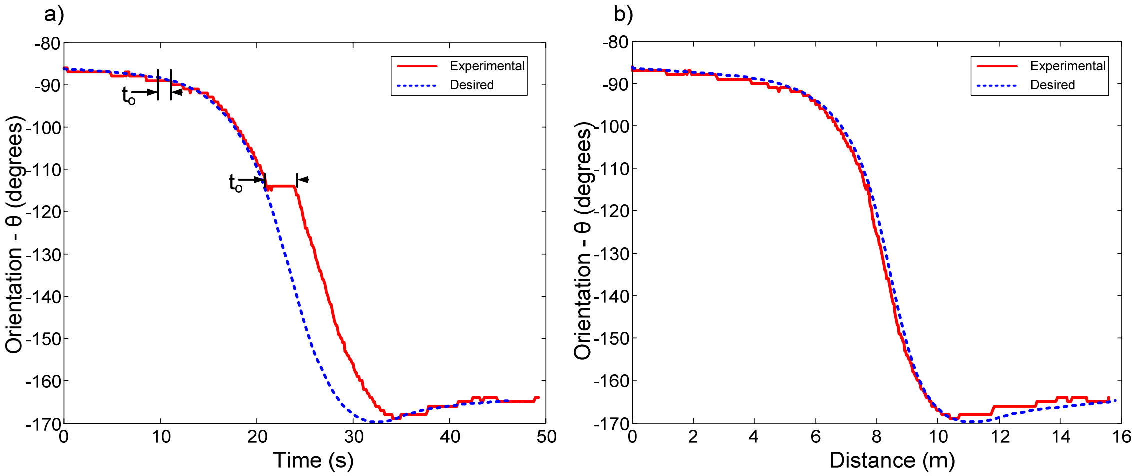


Figure 48: Experiment 3, experimental and desired orientation with moving obstacles present. a) Versus time. b) Versus distance.

Table 10 displays the difference between the experimental and simulated results of experiment 3. The MSE can not be calculated since there are no simulated results for obstacle avoidance. As a comparison with experiment 2, the difference in goal position in this experiment is 0.111 m which is similar to the results in experiment 2 which were between 0.030 m. and 0.221 m.

Table 10: Experiment 3, difference in experimental and desired results.

Test	**1**
MSE of Velocity Profile (m/s)	N/A
MSE of Orienation (deg)	N/A
Difference in E and S* Start Position (m)	0.005
Difference in E and S* Goal Position (m)	**0.111**
Difference in E and S* Time (s)	2.72
Difference in E and S* Initial Orientation (deg)	0.295
Difference in E and S* Final Orientation (deg)	1.00

* Experimental and Simulated

4.4: Experiment 4

This experiment is designed to simulate a healthcare environment, such as a hospital ward. The location chosen for this experiment included a *ramp*, a corridor, doorway and level ground. The ramp has a total length of 4.29 m, a height of 0.450 m and a grade of 5.99^{o}. The safety speed limit is set to $v_{safety} = 0.500$ m/s (roughly 24% of the robots maximum velocity) due to the limitations in the obstacle avoidance algorithm. In future research, it may be possible to increase the safety limit to be coincident with the robot's velocity limit, v_{robot}. In this experiment, the safety speed limit is not enforced to test the limits of the robot. The experimental and simulated 2D and 3D views for this path are shown in Figure 49. As can be seen from this figure, this path is composed of 6 knots. For this experiment it was determined that $M_{\text{int}} = 13$ intervals between each knot resulted in a sample rate, $T_s = 0.483$ seconds, which was acceptable as compared to the optimum value of 0.5 seconds.

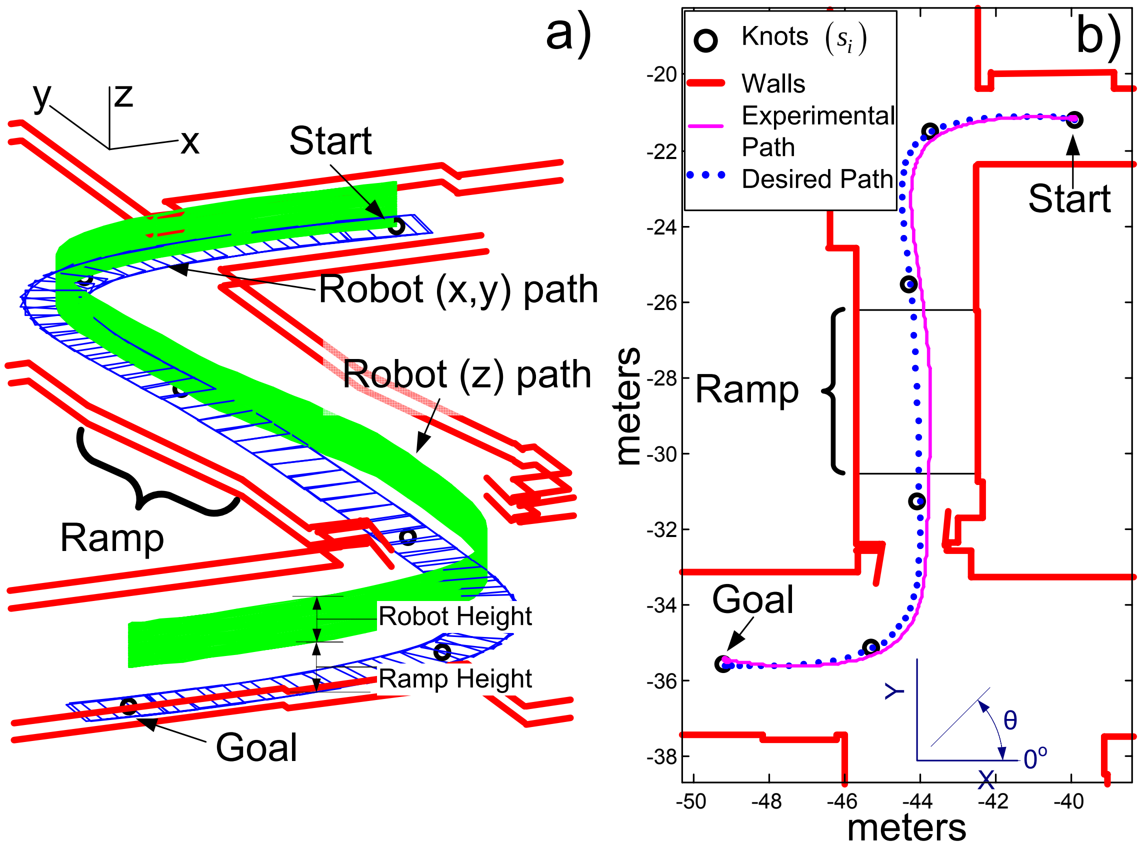


Figure 49: Experiment 4, path located in the Chemical Engineering department. a) 3D simulated (desired) view. b) 2D experimental view.

Two tests were completed for this experiment. As can be seen from Table 13, test 2 had the lowest difference in simulated and experimental destination position of 0.119 m, and therefore its results were used. Table 11 shows a number of physical parameters for each test and the simulated results.

Table 11: Experiment 4, physical parameters.

Test	1	2	Simulated
Start Point (x,y,z) (m)*	(-39.9,-21.2,0)	(-39.9,-21.2,0)	(-39.3,21.2,0)
Goal Point (x,y,z) (m)	(-49.0,-35.5,0.45)	(-49.2,-35.4,0.45)	(-49.2,-35.6,0.45)
Distance Travelled (m)	22.1	22.2	22.2
Time of Motion (s)	31.6	31.9	31.4
Initial Orientation (deg)*	-185	-187	-186
Final Orientation (deg)	-190	-190	-183

* Initially specified by the user. All others are experimental results (tests 1 and 2)

An arbitrary maneuver time of 5 seconds was assigned to the path to convert it into a rudimentary trajectory. This rudimentary trajectory was made feasible by using the mobile robot constraint limits and stretching the time of motion to 31.4 seconds (refer to Section 2.3). The constraint limits are shown in Table 12. Figure 50 displays the scaled acceleration and velocity profiles along with the velocity, acceleration and deceleration constraints. The effect of the ramp can be seen in Figure 50, where the acceleration and deceleration limits are affected.

Table 12: Experiment 4, velocity, acceleration and deceleration limits.

Velocity Limits (m/s)		Acceleration Limits (m/s^2)		Deceleration Limits (m/s^2)	
v_{safety}	N/A	a_{robot}	0.550	d_{robot}	-7.80
v_{robot}	2.10	a_{tip}	4.97	d_{tip}	-4.97
v_{slide}	2.08	a_{slip}	0.666	d_{slip}	-0.666
V_{lim}	**2.08**	**A_{lim}**	**0.550**	**D_{lim}**	**-0.666**

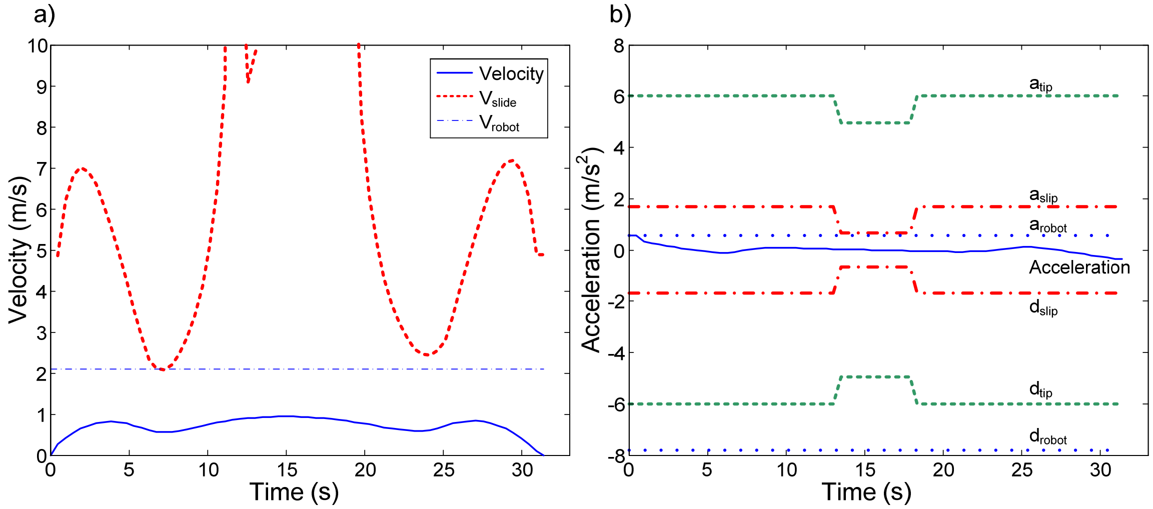


Figure 50: Experiment 4, a) Robot velocity – time with constraints. b) Robot acceleration – time with constraints.

The un-scaled and scaled velocity, acceleration, and deceleration profiles are shown in Figure 51. The left side of Figure 51 shows a total maneuver time of 5 seconds, maximum velocity of 5.89 m/s, maximum acceleration of 21.7 m/s^2, and a maximum deceleration of -13.8 m/s^2. All these values are beyond the robot limits as shown in Table 12. As it can be observed on the right-hand side of Figure 51, increasing the time of motion to 31.4 seconds resulted in a maximum velocity of 0.939 m/s, a maximum acceleration of 0.550 m/s^2, and a maximum deceleration of -0.351 m/s^2. The velocity is well below the limit of V_{lim}=2.08 m/s, the acceleration is equal to the limit of A_{lim}=0.550 m/s^2 and the deceleration is above the limit of D_{lim}=-0.666 m/s^2. Notice that the velocity is well above the safety limit of 0.500 m/s, which was not enforced in this experiment.

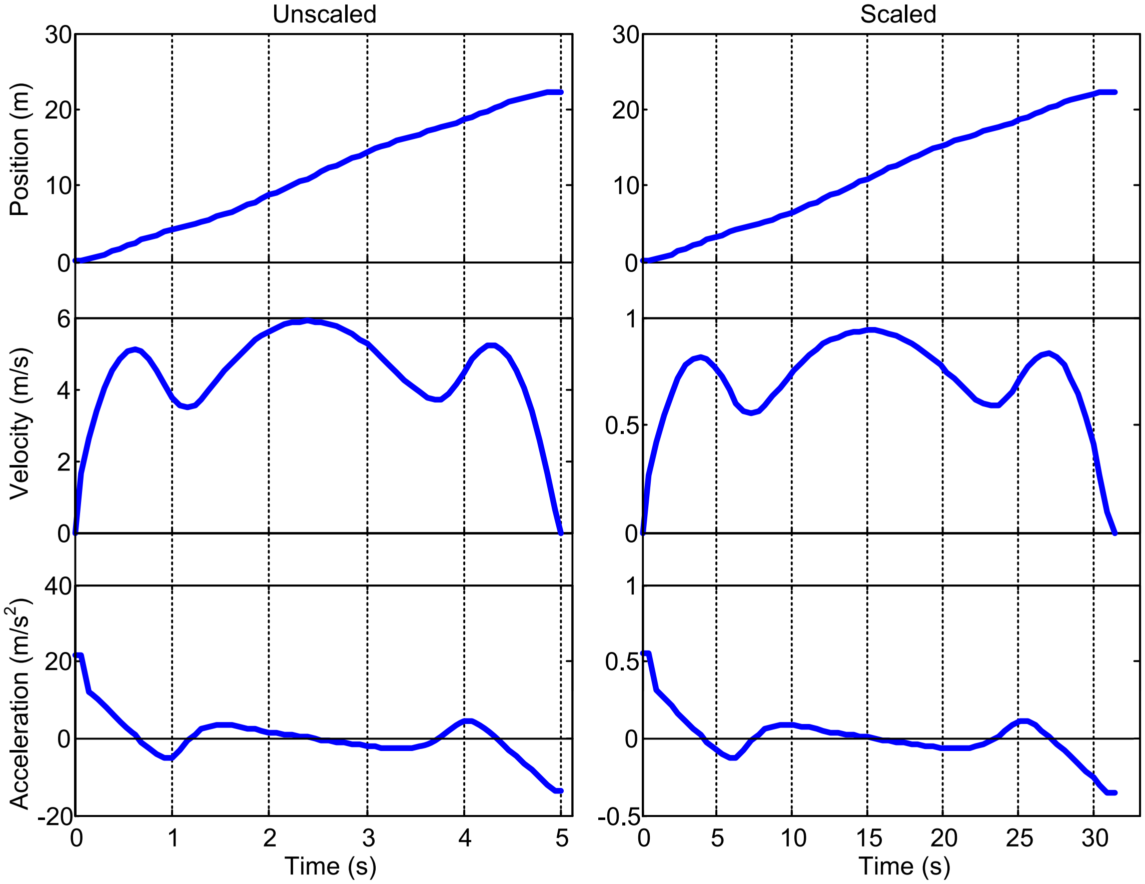


Figure 51: Experiment 4, simulated position, velocity, and acceleration. Un-scaled (left) and scaled (right).

Figure 52 displays the linear velocities of both the robot and the wheels.

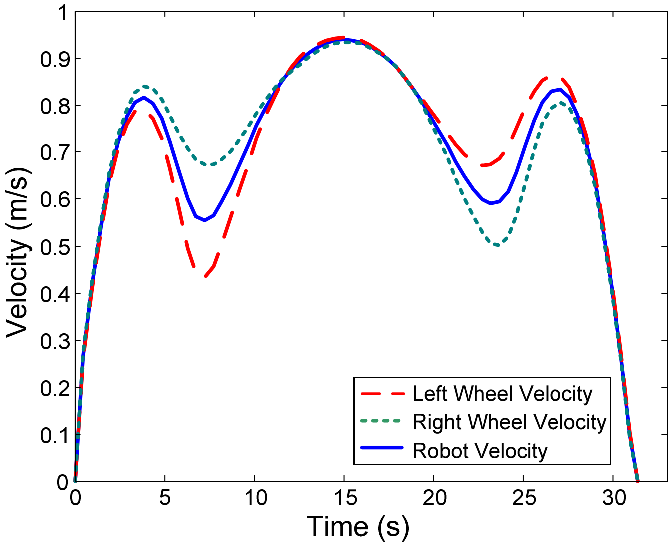


Figure 52: Experiment 4, linear velocities of the center of the robot and the center of the left and right wheels.

Figure 53 shows the experimental and simulated (desired) velocity profiles both versus time and distance. Figure 54 shows the experimental and simulated (desired) orientation.

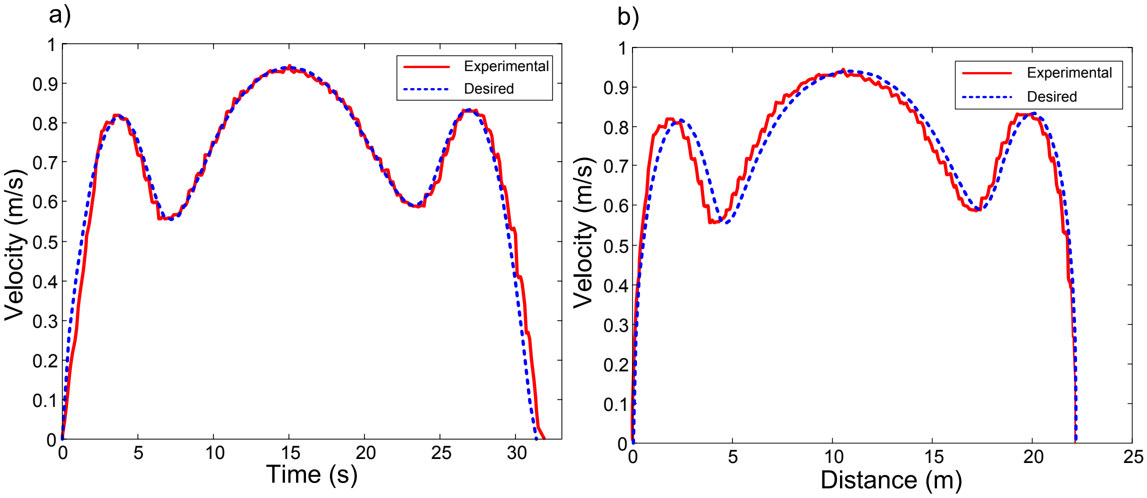


Figure 53: Experiment 4, experimental and simulated (desired) linear robot velocities. a) Versus time. b) Versus distance.

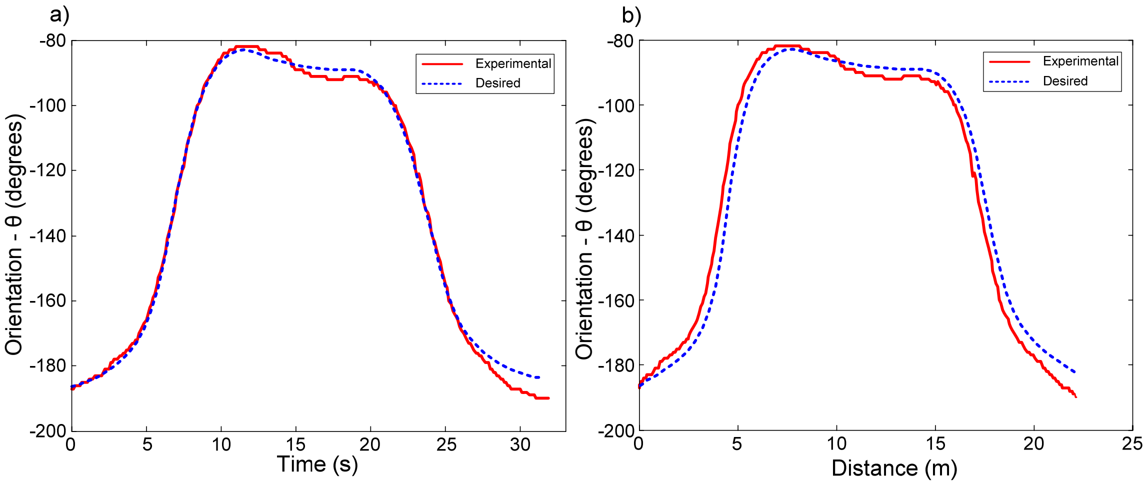


Figure 54: Experiment 4, experimental and simulated (desired) orientation. a) Versus time. b) Versus distance.

Table 13 displays the difference between the experimental and simulated results for all tests performed. In both cases, the velocity profiles had an error of around 0.02 m/s which is relatively low as compared to the maximum velocity of the path (0.939 m/s). The orientation in both tests had a MSE of 3.31^{o} and 2.98^{o} respectively, and a difference in goal position of 0.168 m, and 0.119 m respectively. As can be seen from the results, the robot was able to correctly navigate the ramp and did not have a large impact on the ability of the robot to track the path.

Table 13: Experiment 4, difference in experimental and simulated (desired) results.

Test	**1**	**2**
MSE of Velocity Profile (m/s)	0.022	0.025
MSE of Orienation (deg)	3.31	2.98
Difference in E and S* Start Position (m)	0.018	0.010
Difference in E and S* Goal Position (m)	**0.168**	**0.119**
Difference in E and S* Time (s)	0.254	0.473
Difference in E and S* Initial Orientation (deg)	1.00	1.00
Difference in E and S* Final Orientation (deg)	7.00	7.00

* Experimental and Simulated

4.5: Experiment 5

Experiment 5 contains two turns and is located on the main hallway north of the robotics lab (1B88). Like in experiment 4, in this experiment, the safety speed limit is removed to test the limits of the robot. The experimental and simulated paths for this experiment are shown in Figure 55. As can be seen from this figure, this path is composed of 6 knots. It was determined that $M_{\text{int}} = 9$ intervals between each knot resulted in a sample rate $T_s = 0.520$ seconds, which was acceptable as compared to the optimum value of 0.5 seconds.

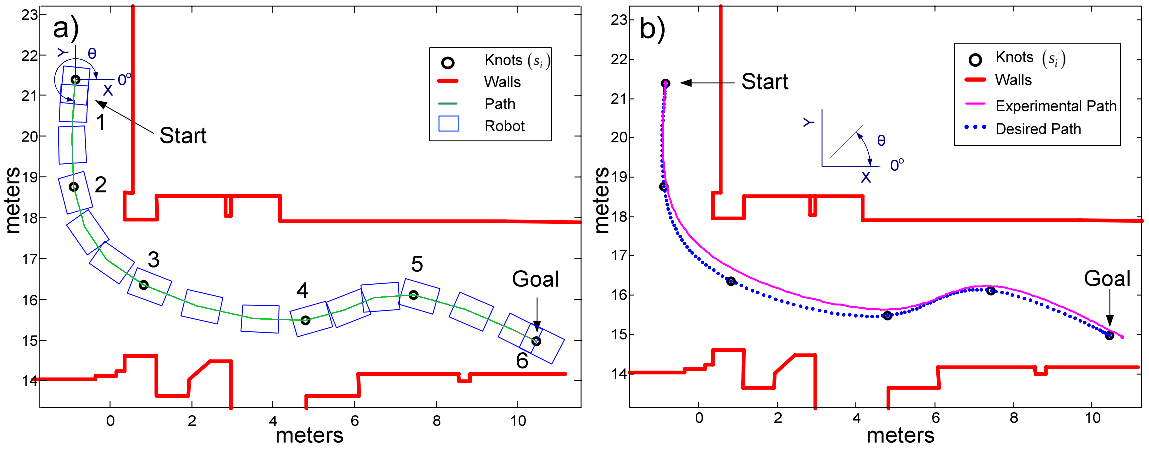


Figure 55: Experiment 5, path located in the hallway north of the Robotics Lab 1B88. a) Simulated (desired). b) Experimental.

Three tests were completed for this experiment. As can be seen from Table 16, test 1 had the lowest error in final destination of 0.328 m., and therefore its results were graphed. Table 14 shows a number of physical parameters for each test and the simulated results.

Table 14: Experiment 5, physical parameters.

Test	1	2	3	Simulated
Start Point (x,y,z) (m)*	(-0.841,21.4,0)	(-0.838,21.4,0)	(-0.829,21.4,0)	(-0.840,21.4,0)
Goal Point (x,y,z) (m)	(10.8,14.9,0)	(10.9,15.8,0)	(10.9,15.1,0)	(10.5,15.0,0)
Distance Travelled (m)	15.9	15.9	15.9	15.8
Time of Motion (s)	25.1	25.2	25.0	23.4
Initial Orientation (deg)*	-97.0	-96.0	-97.0	-96.4
Final Orientation (deg)	-28.0	-27.0	-28.0	-27.2

* Initially specified by the user. All others are experimental results (tests 1 to 3)

An arbitrary time of motion of 5 seconds was assigned to the path to convert it into a rudimentary trajectory. This rudimentary trajectory was made feasible by using the mobile robot constraint limits and stretching the time of motion to 23.4 seconds. The constraint limits are shown in Table 15. Figure 56 displays the scaled acceleration and velocity profiles along with the velocity, acceleration and deceleration constraints.

Table 15: Experiment 5, velocity, acceleration and deceleration limits.

Velocity Limits (m/s)		Acceleration Limits (m/s^2)		Deceleration Limits (m/s^2)	
v_{safety}	N/A	a_{robot}	0.550	d_{robot}	-7.80
v_{robot}	2.10	a_{tip}	6.02	d_{tip}	-4.97
v_{slide}	2.60	a_{slip}	1.70	d_{slip}	-1.70
V_{lim}	**2.10**	**A_{lim}**	**0.550**	**D_{lim}**	**-1.70**

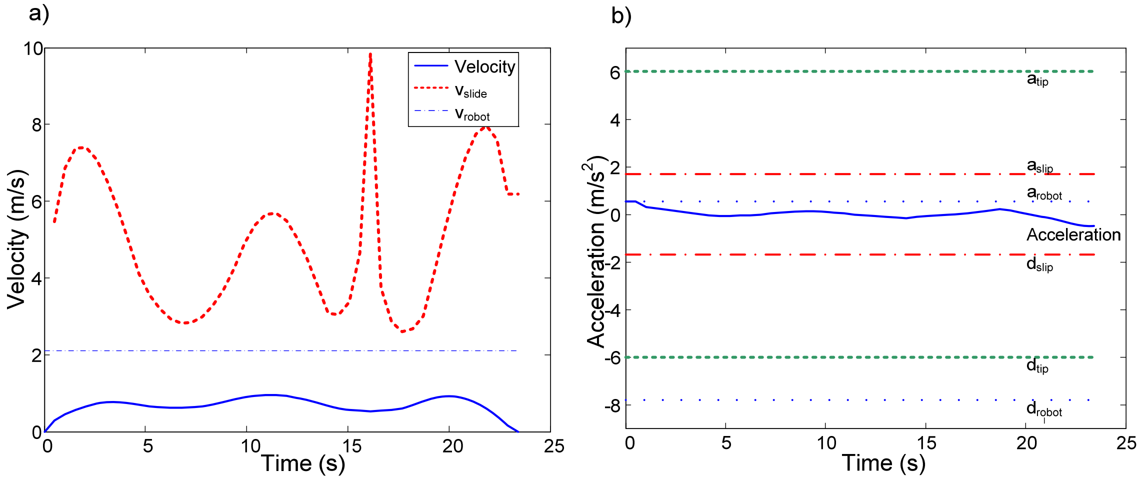


Figure 56: Experiment 5, a) Robot velocity – time with constraints. b) Robot acceleration – time with constraints.

The un-scaled and scaled velocity, acceleration, and deceleration profiles are shown in Figure 57. The left-hand side of Figure 57 shows a total maneuver time of 5 seconds, a maximum velocity of 4.43 m/s, a maximum acceleration of 12.05 m/s^2, and a maximum deceleration of -10.56 m/s^2. All three values are beyond the robot limits shown in Table 15. The right-hand side (constrained) of Figure 57 displays a total time of 28.5 seconds, a maximum velocity of 0.950 m/s, a maximum acceleration of 0.550 m/s^2, and a maximum deceleration of -0.480 m/s^2, all of which are below the constraint limits of V_{lim}=2.1 m/s, A_{lim}=0.550 m/s^2, and D_{lim}=-1.70 m/s^2.

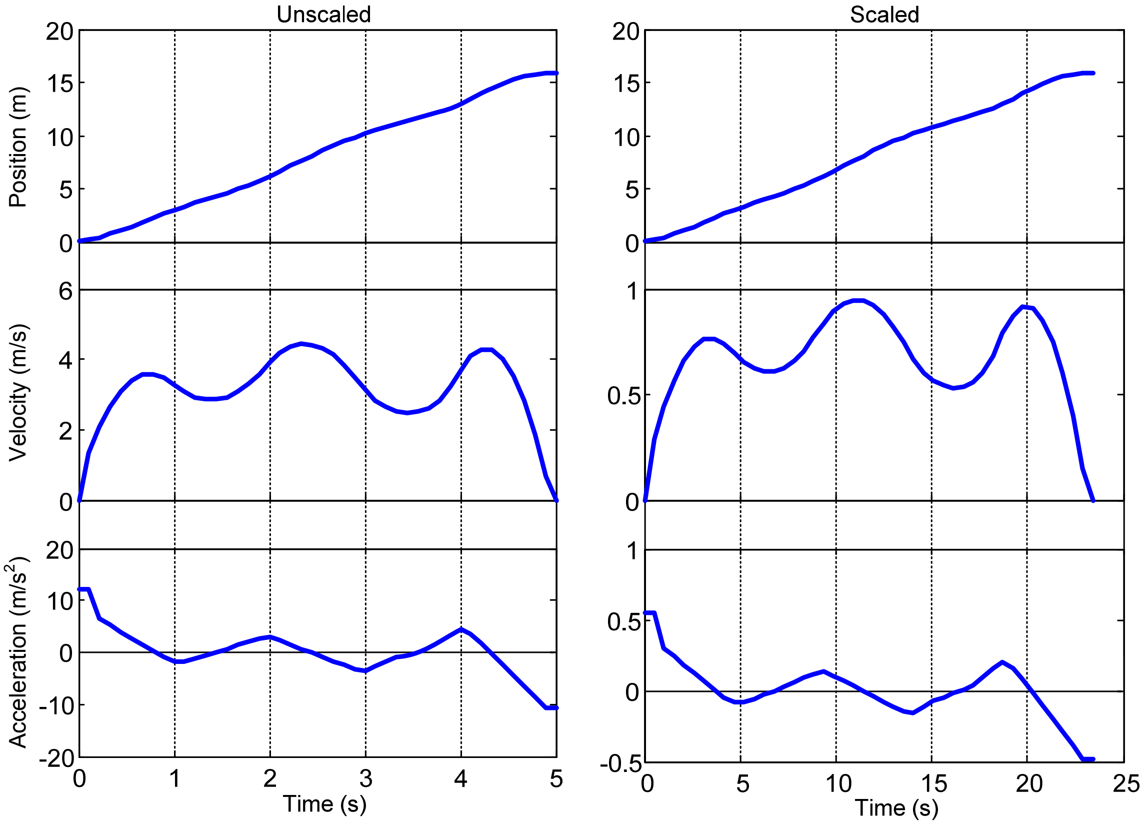


Figure 57: Experiment 5, simulated position, velocity, and acceleration. Un-scaled (left) and scaled (right).

Figure 58 displays the linear velocities of both the robot and the wheels.

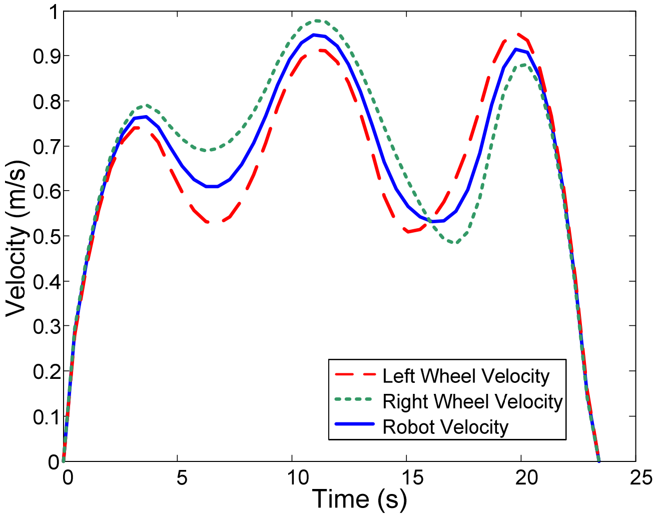


Figure 58: Experiment 5, linear velocities of the center of the robot and the center of the left and right wheels.

Figure 59 shows the experimental and simulated (desired) velocity profiles both in respect to time and distance. Figure 60 shows the experimental and desired orientation.

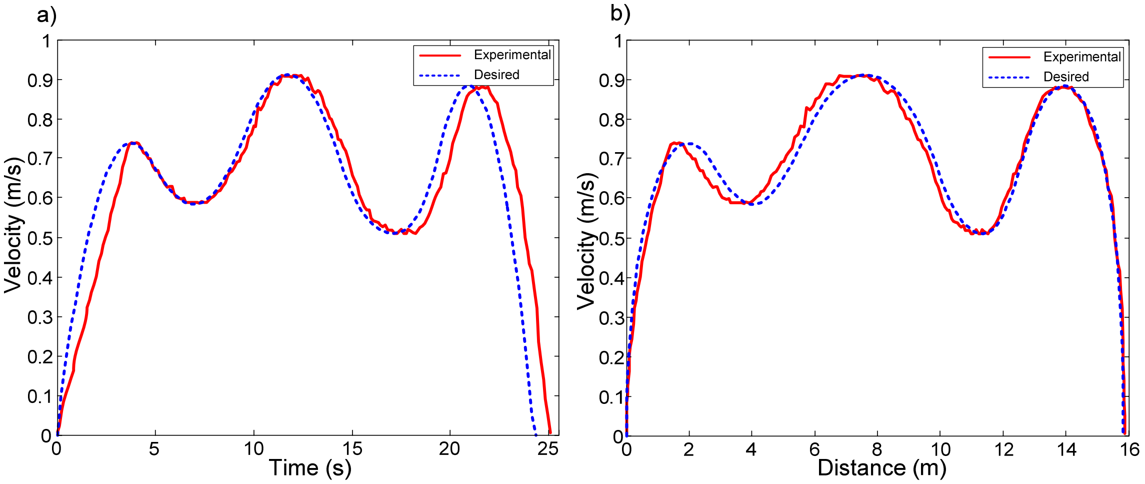


Figure 59: Experiment 5, experimental and simulated (desired) linear robot velocities. a) Versus time. b) Versus distance.

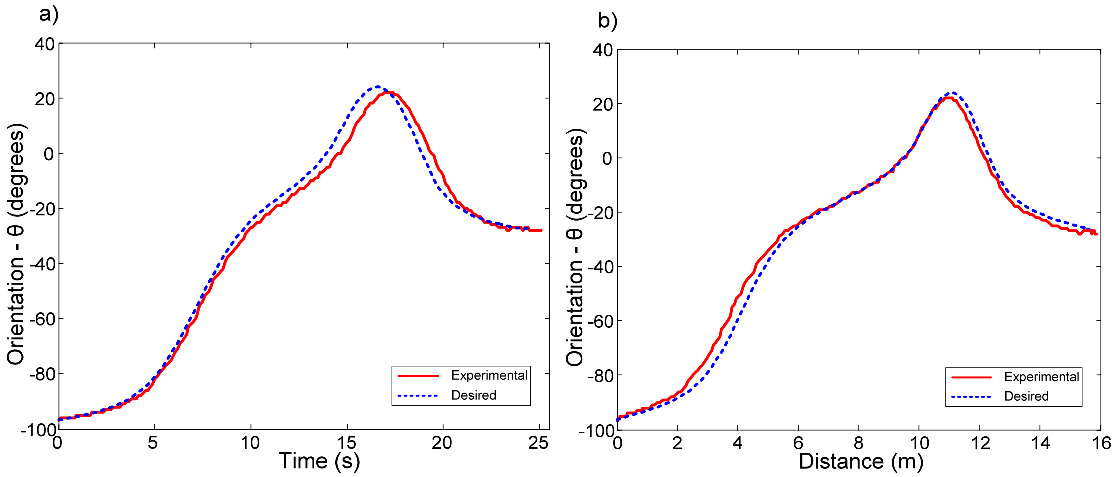


Figure 60: Experiment 5, experimental and desired orientation. a) Versus time. b) Versus distance.

Table 16 displays the difference between the simulated and desired results for all tests that were performed. In all cases, the velocity profiles had an error of around 0.09 m/s which is relatively low as compared to the maximum velocity of the path (0.95 m/s). The orientation in tests 1, 2 and 3 had a MSE of 3.65°, 3.72°, and 3.61° and a difference in goal position of 0.328 m, 0.473 m and 0.460 m respectively. Once again from this data it appears that the error in orientation and the error in the velocity profile have a large impact on the ability of the robot to track the path.

Table 16: Experiment 5, difference in experimental and simulated (desired).

Test	**1**	**2**	**3**
MSE of Velocity Profile (m/s)	0.092	0.095	0.097
MSE of Orienation (deg)	3.65	3.72	3.61
Difference in E and S* Start Position (m)	0.017	0.011	0.022
Difference in E and S* Goal Position (m)	**0.328**	**0.473**	**0.460**
Difference in E and S* Time (s)	0.816	0.923	0.763
Difference in E and S* Initial Orientation (deg)	0.579	0.421	0.579
Difference in E and S* Final Orientation (deg)	0.785	0.215	0.785

* Experimental and Simulated

4.6: Experiment 6

In this experiment, the velocity profile is given and a straight line path composed of 131 knots is used as shown in Figure 61. The given velocity profile is specified as follows:

$$v_s = \begin{cases} 0.066i & 0 \le i \le 15 \\ 1 & 15 < i \le 50 \\ 1 - 0.066(i-50) & 50 < i \le 60 \\ 0.33 & 60 < i \le 90 \\ 0.33 + 0.033(i-90) & 90 < i \le 100 \\ 0.66 & 100 < i \le 120 \\ 0.66 - 0.066(i-120) & 120 < i \le 130 \end{cases} . \tag{4.1}$$

Where v_s is the specified linear velocity of the mid-point of the two driving wheels and i is the number of knots. The specified velocity profile and the simulated velocity profile are shown in Figure 62. In this experiment $M_{\text{int}} = 5$, and as observed in Figure 62, this results in a jagged velocity profile.

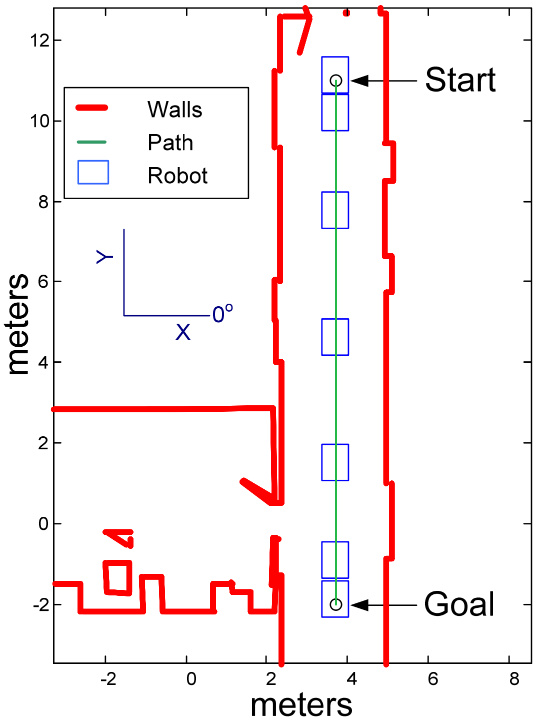


Figure 61: Experiment 6, simulated (desired) path located in the hallway outside of the robotics lab.

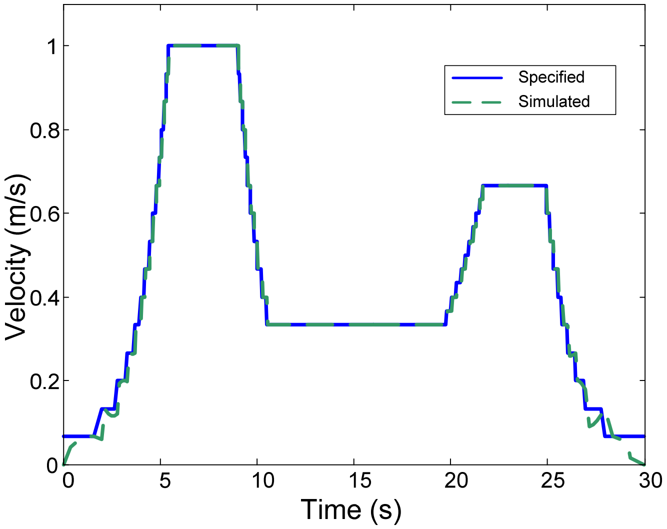


Figure 62: Experiment 6, specified and simulated velocity profile.

Table 17 shows a number of physical parameters for this test and the simulated results.

Table 17: Experiment 6, physical parameters.

Test	1	Simulated
Start Point (x,y,z) (m)*	(3.70,11.0,0)	(3.70,11.0,0)
Goal Point (x,y,z) (m)	(3.91,-2.29,0)	(3.70,-2.00,0)
Distance Travelled (m)	13.3	13.0
Time of Motion (s)	29.8	30.0
Initial Orientation (deg)*	-90.0	-90.0
Final Orientation (deg)	-89.0	-90.0

* Initially specified by the user. All others are experimental results (test 1)

The position with respect to time can be obtained by integrating the specified velocity profile and the acceleration can be obtained by taking the derivate of the velocity profile. Figure 63 shows the position, velocity, and acceleration with respect to time. The velocity has a maximum speed of 1.00 m/s, the maximum acceleration is 3.33 m/s^2, and the maximum deceleration is 3.11 m/s^2. The acceleration is beyond the robot's mechanical limits; therefore, a source of error is added into this experiment.

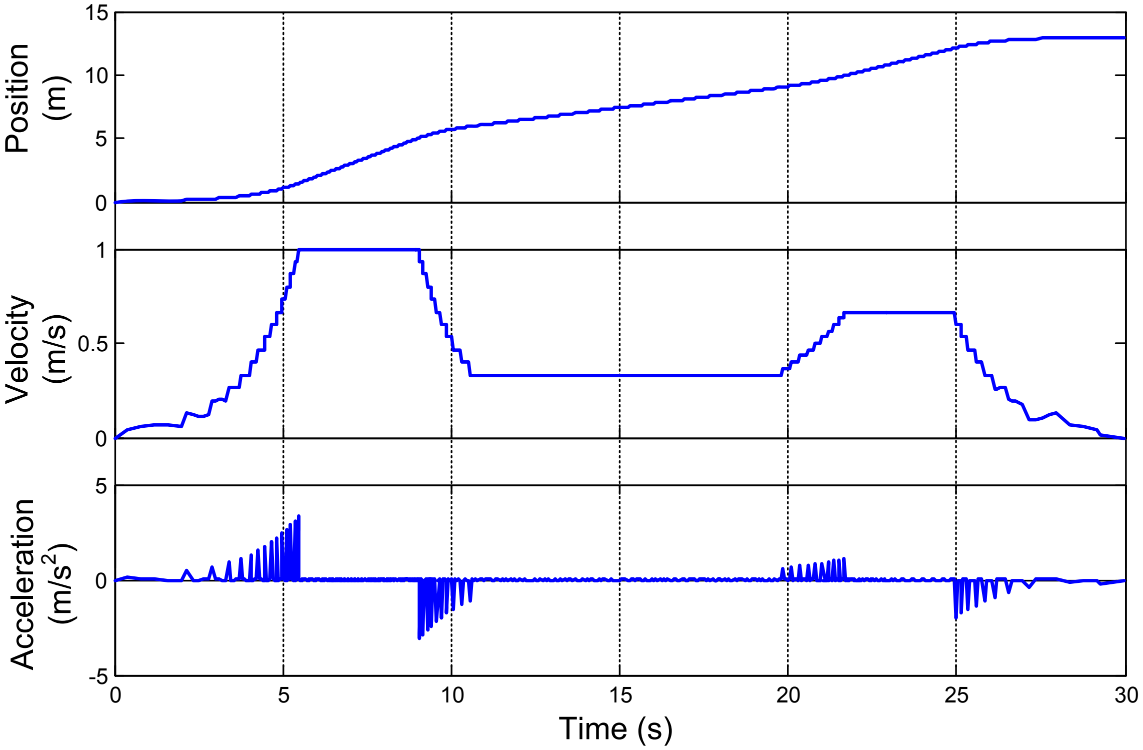


Figure 63: Experiment 6, simulated position, velocity, and acceleration versus time graphs.

Figure 64 displays the experimental and simulated (desired) velocity profiles versus time and distance.

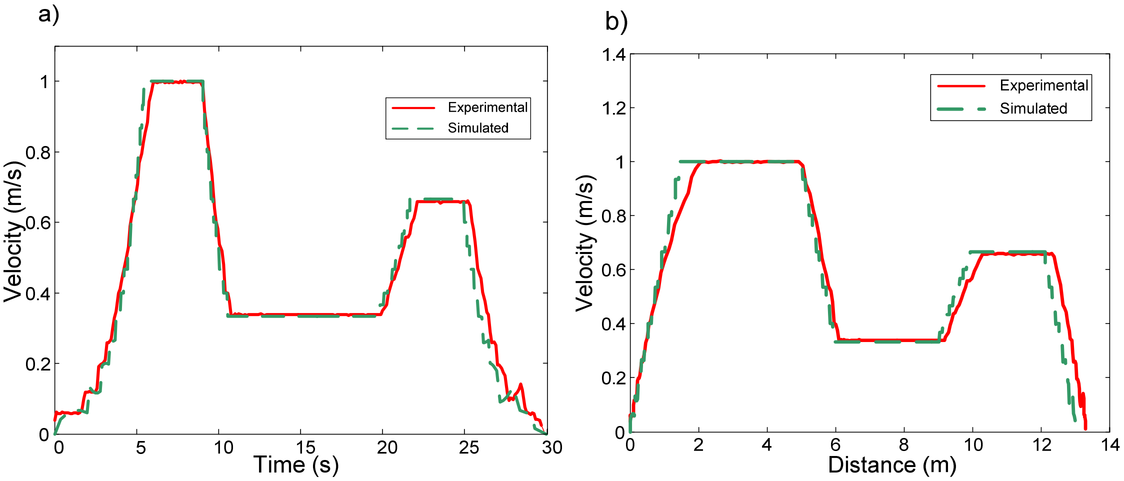


Figure 64: Experiment 6, experimental and simulated (desired) linear robot velocities. a) Versus time. b) Versus distance.

Table 18 shows the difference between the experimental and desired results. The velocity profile has a MSE of about 0.04 m/s. The MSE of orientation was not provided since the orientation change is 0. The large difference in goal position can be attributed to the inability of the robot to reach the maximum acceleration of 3.33 m/s.

Table 18: Experiment 6, difference in experimental and desired results.

Test	**1**
MSE of Velocity Profile (m/s)	0.043
MSE of Orienation (deg)	N/A
Difference in E and S* Goal Position (m)	0.023
Difference in E and S* Goal Position (m)	**0.928**
Difference in E and S* Time (s)	0.158
Difference in E and S* Initial Orientation (deg)	0
Difference in E and S* Final Orientation (deg)	1.00

* Experimental and Simulated

4.7: Analysis

4.7.1: Velocity Profile

In all cases, the mobile robot was able to follow the velocity profile very well. In experiments 1 and 2, the MSE for the velocity profile ranged from 0.011 m/s to 0.032 m/s, which is 2.20% to 6.40% of the maximum speed of 0.500 m/s. In experiment 4, the velocity profile ranged from 0.022 m/s to 0.025 m/s, which is 2.34% to 2.66% of the maximum speed of 0.939 m/s. In experiment 5, the MSE for the velocity profile ranged from 0.092 m/s to 0.097 m/s, which is 9.68% to 10.2% of the maximum speed of 0.950 m/s. Experiment 6 had a MSE of 0.043 m/s for the velocity profile. This value is only 4.3% of the maximum value of 1.00 cm/s.

4.7.2: Orientation

The MSE of the orientation varied considerably depending on how many turns and how large the turning radius was. Experiment 1 had two turns and the MSE of the orientation varied from 3.31^{o} to 4.95^{o} between the three tests. The MSE ranged from 0.903^{o} to 1.00^{o} for orientation in experiment 2, which had only 1 turn. Experiment 4 had a slight curve and the MSE for orientation varied from 1.30^{o} to 1.48^{o} degrees. Experiment 5 had two turns and it's MSE for orientation ranged from 3.61^{o} to 3.72^{o} degrees.

4.7.3: Goal Position

The difference in experimental and simulated goal position varied considerably between each test in each experiment. The greatest change was in experiment 1, where the difference varied from 0.050 m to 0.394 m for different tests. One general trend that was observed was that as both the MSE in orientation and MSE in the velocity profile increased, the farther the robot was from the desired goal point. This trend can be seen in Figure 65.

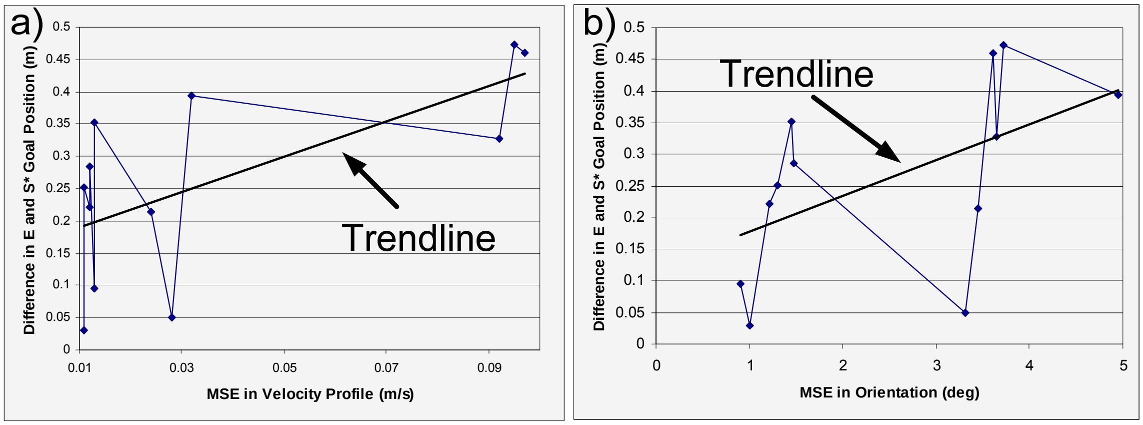


Figure 65: Difference in experimental and simulated goal position versus a) MSE in velocity profile b) MSE in orientation.

The reasons of the disparity in the goal position could be due to: incorrect tire pressure, error in the measuring devices, the onboard computer being unable to keep up with computations, etc.

4.7.4: Obstacle Avoidance

The obstacle avoidance algorithm was relatively successful, as can be seen from experiment 3. Table 8 and Table 10 show the difference in the final goal position of experiment 3 (0.111 m) is relatively close to the difference in the final goal position in experiment 2 (between 0.030 m to 0.221 m).

4.7.5: Predefined Velocity Profile

In the case where a predefined velocity profile was provided, the robot was able to follow the profile relatively well. In experiment 6, due to the fact that the acceleration was beyond the robot's limits, error was added into the system.

CHAPTER 5: CONCLUSIONS AND FUTURE WORK

5.1: Major Conclusions

- Previous works have mainly dealt with temporal planning (generating a velocity profile) in a 2D environment. This project presented a new algorithm for 3D temporal planning. The PowerBot mobile robot was purchased by the U of S Robotics Lab (funded by the Canadian Foundation of Innovation, CFI). All simulations and experiments were conducted on this platform.
- Six different simulations and tests were completed. In all experimental tests, the mobile robot was able to follow the desired velocity profile reasonably well. The time of motion in all experiments was only a fraction different from the simulated case.
- The difference between experimental and simulated goal position varied with the error in orientation and velocity profile. The general trend was the lower the error in the orientation and velocity profile, the closer the robot was to the desired goal position.
- In a healthcare type environment (ramp, door, corridor), the robot was able to track the path and follow the velocity profile exceptionally well.
- The obstacle avoidance algorithm was successful in avoiding moving obstacles without adding significant error into the system.
- If a predefined velocity profile was provided to the robot, the robot was able to follow the profile relatively well.

5.2: Limitations

- The size of the path must be kept relatively small since only open loop control is used in this experiment; therefore, error is not bounded.
- The initial start point and orientation of the robot in experimentation, is determined only by the MobileEyes software package. This package does not provide accuracy below 10 cm. This adds an error into the system before the experimentation has even started.
- Currently moving obstacle avoidance can only be done if the maximum velocity of the robot is 0.5 m/s. Obstacle avoidance could only be tested in an experimental setting since it is very difficult to synchronize two moving robots at the same time.
- The dynamic constraints are only valid for environments that contain a similar tiled floor as the U of S Engineering Building. This is due to the fact that the coefficient of friction between the tires and the surface is calculated only for this material.

5.3: Future Work

- In order to get the difference in experimental and desired goal position as low as possible, instead of using an open loop control system, a closed loop control system should be applied to this algorithm. This would allow for the correction of any initial start point errors, and would allow for the robot to track very large paths.
- Optimization of the algorithm based on a given cost measure (time, accuracy).
- Update the obstacle avoidance algorithm to use both laser and sonar sensors and to remove the 0.5 m/s velocity limit.
- The long term goal of this research is to extend this work into an outdoor setting.

Work is currently underway to extend this algorithm to a mobile manipulator (mobile robot with a manipulator arm) [29].

REFERENCES

[1] J. Craig, “Introduction to Robotics: Mechanism and Control”, Addison-Wesley Publishing Co., 2nd Edition, 1989

[2] I. Waheed, R. Fotouhi, “Trajectory / Temporal Planning of an Intelligent Healthcare Wheeled Mobile Robot”, UVS Canada Conference, Montebello, Quebec, Nov. 7-10, 2006.

[3] J. C. Mackinnon, “The arithmetic of health care”, Canadian Medical Association Journal, 171(6), pp. 603-604 (2004)

[4] T. Hatase, Y. Wakamatsu, H. Nishimura, H. Yamamoto, “Intelligent Wheelchair Robot”, Fujitsu, vol. 57, no. 3, pp. 263-8 (2006)

[5] F. Bley, M. Rous, U. Canzler, K.-F. Kraiss, “Supervised Navigation and Manipulation for Impaired Wheelchair Users”, Proc. IEEE Conference of Systems, Man and Cybernetics (The Hague, Netherlands, Oct. 10-13, 2004), vol. 3, pp. 2790-2796.

[6] G. Pacnik, “Voice Operated Intelligent Wheelchair – VOIC”, IEEE International Symposium on Industrial Electronics (Dubrovnik, Croatia, Jun 20-23, 2005), pp. 1221-1226.

[7] C. Gurtler, “Trajectory Planning for Mobile Robots Based on Dynamical Models”, Proc. IEEE Int. Conf. Intelligent Engineering Systems (Budapest, Hungary, 1997) pp. 171-174.

[8] K. Jiang, L.D. Seneviratne, S.W.E Earles, “Time-optimal smooth-path motion planning for a mobile robot with kinematic constraints”, Robotica, vol. 15, pp. 547-553 (1997).

[9] S. Aydin, H. Temeltas, “Fuzzy-Differential Evolution Algorithm for Planning Time-Optimal Trajectories of a Unicycle Mobile Robot on a Predefined Path.”, Advanced Robotics, vol. 18, no. 7, pp. 725-748 (2004).

[10] A. Stentz, “Optimal and Efficient Path Planning for Unknown and Dynamic Environments”, International Journal of Robotics and Automation, vol. 10, no. 3, (1995).

[11] C. Louste, A.Liegeois, “Path Planning for Non-holonomic Vehicles: a Potential Viscous Fluid Field Method”, Robotica, vol. 20, pp. 291-298 (2002).

[12] A. Yahja, S. Singh, A. Stentz, “An Efficient On-Line Path Planner for Outdoor Mobile Robots”, Robotics and Autonomous Systems, vol. 32, pp. 129-143 (2000).

[13] M. Cherif, “Motion Planning for All-Terrain Vehicles: A Physical Modeling Approach for Coping with Dynamic and Contact Interaction Constraints”, IEEE Transactions on Robotics and Automation, vol. 15, no. 2, pp. 202-218 (1999).

[14] Z. Qu, J. Wang, C. Plaisted, “A New Analytical Solution to Mobile Robot Trajectory Generation in the Presence of Moving Obstacles”, IEEE Transactions on Robotics, vol. 20, no. 6, pp. 978-993 (2004).

[15] M. Prado, A. Simon, E. Carabias, A. Perez, F. Ezquerro, “Optimal Velocity Planning of Wheeled Mobile Robots on Specific Paths in Static and Dynamic Environments”, Journal of Robotic Systems, vol. 20, no. 12, pp. 737-754 (2003).

[16] R.E. Parking, “Applied Robotic Analysis”, Prentice Hall, New Jersey, 1991.

[17] G. Paolo, “Technique to Analytically Formulate and to Solve the 2-Dimensional Constrained Trajectory Planning Problem for a Mobile Robot”, Journal of Intelligent & Robotic Systems, vol. 27, no. 3, pp. 237-262 (2000).

[18] V. Munoz, A. Ollero, M. Prado, A. Simon, “Mobile Robot Trajectory Planning with Dynamic and Kinematic Constraints”, Proceedings 1994 IEEE International Conference on Robotics and Automation, pt. 4, pp. 2802-7, vol. 4 (1994).

[19] F. Lamiraux, S. Sekhavat, J.P. Laumond, “Motion Planning and Control for Hilare Pulling a Trailer”, IEEE Transactions on Robotics and Automation, vol. 15, no. 4, pp. 640-652 (1999).

[20] J.S. Choi, B.K. Kim, “Near-Time-Optimal Trajectory Planning for Wheeled Mobile Robots with Translational and Rotational Sections”, IEEE Transactions on Robotics and Automation, vol. 17, no. 1, pp. 85-90 (2001).

[21] R. Fotouhi, W. Szyszkowski, P. Nikiforuk, “Trajectory planning and speed control for a two-link rigid manipulator”, Journal of Mechanical Design, Transactions of the ASME, v124, n 3 (2002).

[22] V.F. Munoz, A. Garcia-Cerezo, A. Cruz, “A Mobile Robots Trajectory Planning Approach under Motion Restrictions”, Integrated Computer-Aided Engineering, v 6, n4, pp. 331-347 (1999).

[23] M. Prado, A. Simon, F. Ezquerro, “Velocity, Acceleration and Deceleration Bounds for a Time-Optimal Planner of a Wheeled Mobile Robot”, Robotica, vol. 20, pp. 181-193 (2002).

[24] Gilbert Strang, “Introduction to Applied Mathematics”, Wellesley College, January 1986.

[25] ActivMedia Robotics, “PowerBot™ AGV Operations Manual”, version 5, March, 2005.

[26] URL: http://mathews.ecs.fullerton.edu/n2003/curvature/CurvatureMod/Links/CurvatureMod_lnk_9.html.

[27] R. Hibbeler, “Engineering Mechanics Dynamics”, Prentice-Hall, 8th Edition, 1998.

[28] ActivMedia Robotics, “Advanced Robotics Navigation and Localization Installation and Operations Manual”, version 2, January 2006.

[29] I. Waheed, R. Fotouhi, “Trajectory / Temporal Planning of Mobile Robot Manipulators”, Canadian Space Agency Workshop, Montreal, Quebec, Oct. 17-18, 2006.

APPENDIX 1: CUBIC SPLINE EXAMPLE

An example showing the steps in creating a simple trajectory by connecting the points, (0,0), (1,0), and (2,2), using cubic splines is discussed below (only the x axis case will be shown in detail).

1. The Hermite cubic will be used in this example as discussed in Section 2.2.

$$X_i(t) = a_i\left(t-1\right)^2\left(2t+1\right) + b_i t^2(3-2t) + c_i t\left(t-1\right)^2 - d_i t^2\left(1-t\right). \qquad \text{(A:1-1)}$$

2. Define the initial and final conditions.

   The initial and final x axis positions are 0 and 2 respectively. The initial and final velocities are both set to 0.

3. Define the boundary conditions:

$$\begin{aligned} &X_1\left(1\right) = X_2\left(0\right) = s_{x_2} = 1 \\ &\frac{dX_1}{dt}\left(1\right) = \frac{dX_2}{dt}\left(0\right) \\ &\frac{d^2X_1}{dt^2}\left(1\right) = \frac{d^2X_2}{dt^2}\left(0\right) \end{aligned} \qquad \text{(A:1-2)}$$

4. Since there are 3 points, 2 segments will be required to join the points. These 2 segments have 4 boundary conditions and 4 continuity conditions, that is 8 equations and 8 unknowns. Rearranging the 8 equations into a matrix format (see equation (2.4)) results in:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} U_0 \\ U_1 \\ U_2 \end{bmatrix} = \begin{bmatrix} v_{initial} = 0 \\ 3\left(s_{x_3} - s_{x_1}\right) = 6 \\ v_{final} = 0 \end{bmatrix} \qquad \text{(A:1-3)}$$

5. Solving for U results in:

$$\begin{bmatrix} U_0 \\ U_1 \\ U_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1.5 \\ 0 \end{bmatrix} \tag{A:1-4}$$

6. The coefficients of the equations are found by the following relation:

$$\begin{aligned} a_i &= s_{x_i} \\ b_i &= s_{x_{i+1}} \\ c_i &= U_{i-1} \\ d_i &= U_i. \end{aligned} \tag{A:1-5}$$

Therefore:

$$\begin{aligned} a_1 &= s_{x_1} = 0 & a_2 &= s_{x_2} = 1 \\ b_1 &= s_{x_2} = 1 & b_2 &= s_{x_3} = 2 \\ c_1 &= U_0 = 0 & c_2 &= U_1 = 1.5 \\ d_1 &= U_1 = 1.5 & d_2 &= U_2 = 0 \end{aligned} \tag{A:1-6}$$

The coefficients of the y axis can also be computed using the same method:

$$\begin{aligned} a_1 &= 0 & a_2 &= 0 \\ b_1 &= 0 & b_2 &= 2 \\ c_1 &= 0 & c_2 &= 1.5 \\ d_1 &= 1.5 & d_2 &= 0 \end{aligned} \tag{A:1-7}$$

Graphing these equations yields:

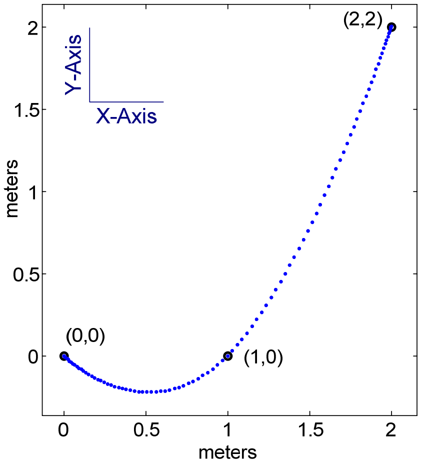


Figure 66: Appendix 3, example of 2 cubic splines.

APPENDIX 2: CALCULATION OF THE COEFFICIENT OF FRICTION

Calculating the static coefficient of friction (μ_s) between the robot drive wheels and the tiles used in the U of S Engineering Building was accomplished by running a series of tests. The robot was placed on a flat clean surface and a cable was attached to the robot. Engaging the robot brakes ensured that the drive wheels would not rotate during the tests. A spring scale was attached to the cable to record the amount of force required to cause the robot to begin sliding. Three tests were completed and the results are shown in Table 19. Figure 67 displays the forces acting on the robot.

Table 19: Appendix 2, force required for the movement of the robot.

Trial	Force Required for Movement (lbs)
1	62
2	58
3	60
Avgerage:	**60**

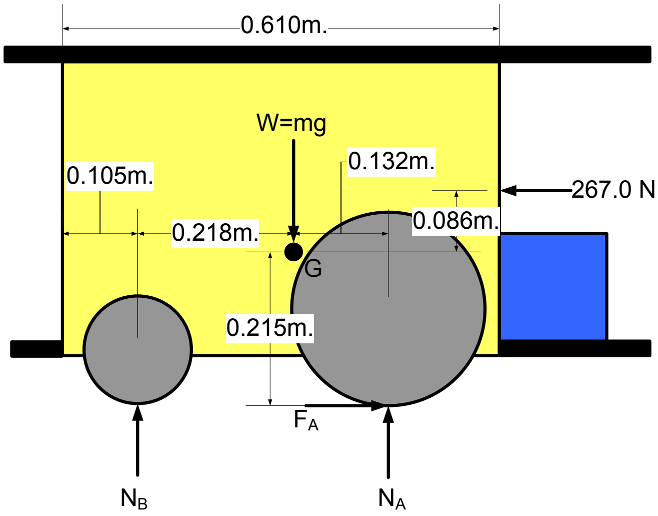


Figure 67: Appendix 2, schematic of the robot showing the forces acting on it to cause it to move.

The coefficient of friction can be determined by the following equation:

$$F_A = \mu_s N_A \qquad \text{(A:2-1)}$$

F_A is the frictional force resisting movement of the driving wheels, and N_A is the normal force of the driving wheels. The frictional and normal force can be obtained by solving the equations of motion as shown:

$$\overset{+}{\leftarrow} \sum F_n = 0; \qquad 267.0 - F_A = 0 \qquad \text{(A:2-2)}$$

$$+\uparrow \sum F_t = 0; \qquad N_A + N_B - 148g = 0 \qquad \text{(A:2-3)}$$

$$+\circlearrowleft \sum M_G = 0; \qquad 0.132N_A + 0.215F_A - 0.218N_B + 267.0\left(0.086\right) = 0 \qquad \text{(A:2-4)}$$

Where N_B is the normal force of the castor wheels and g is the acceleration due to gravity. The equations can be solved to give:

$$\begin{aligned} F_A &= 267.0N \\ N_A &= 804.1N \\ N_B &= 647.8N \end{aligned} \qquad \text{(A:2-5)}$$

Referring to (A:2-1), the coefficient of friction can be calculated as follows:

$$\begin{aligned} F_A &= \mu_s N_A \\ \mu_s &= 0.332 \end{aligned} \qquad \text{(A:2-6)}$$

APPENDIX 3: CALCULATION OF THE CENTER OF GRAVITY

Since so many dynamic calculations are dependent on the location of the center of gravity, the location should be as accurate as possible (see Chapter 2). This problem is broken into two main parts. In the first part, the horizontal position of the center of gravity is calculated. The second part calculates the vertical position of the center of gravity.

A3.1: Calculation of the Horizontal Component of the Center of Gravity

The mass of the robot is experimentally determined to be 148 kg (326 lb) by using a scale. The scale is then used for experimentally determining the normal force of the driving wheels as can be seen from Figure 68.

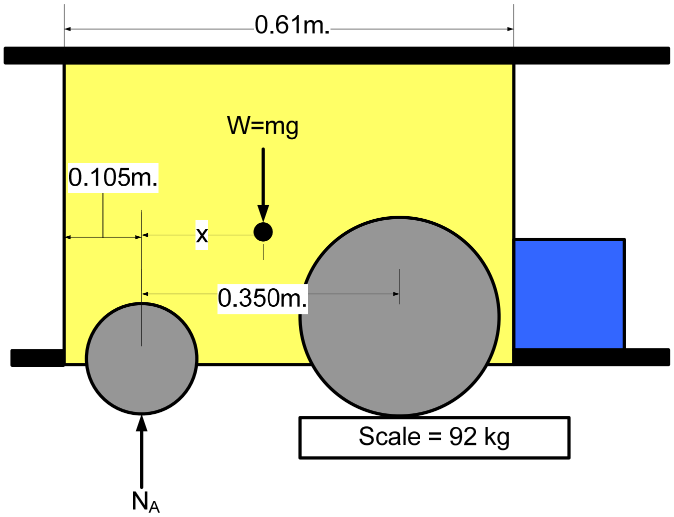


Figure 68: Appendix 1, calculation of the horizontal component of the center of gravity.

Taking the summation of the moments at point "A" and solving for x gives the horizontal coordinate for the center of gravity as shown:

$$+\circlearrowleft\sum M_A = 0; \qquad 148gx - 92g\left(0.350\right) = 0$$
$$x = 0.218m \qquad \text{(A:3-1)}$$

The center of gravity of the horizontal component calculated from the lower left corner of the robot is calculated to be:

$$x_{cg} = x + 0.105 = 0.323m. \qquad \text{(A:3-2)}$$

A3.2: Calculation of the Vertical Component of the Center of Gravity

The vertical component of the center of gravity is accomplished by raising the robot and experimentally determining the normal force of the driving wheels. Figure 69 displays a schematic of a section of the robot after the castor wheels have been raised 0.062 meters by placing a wooden block beneath them.

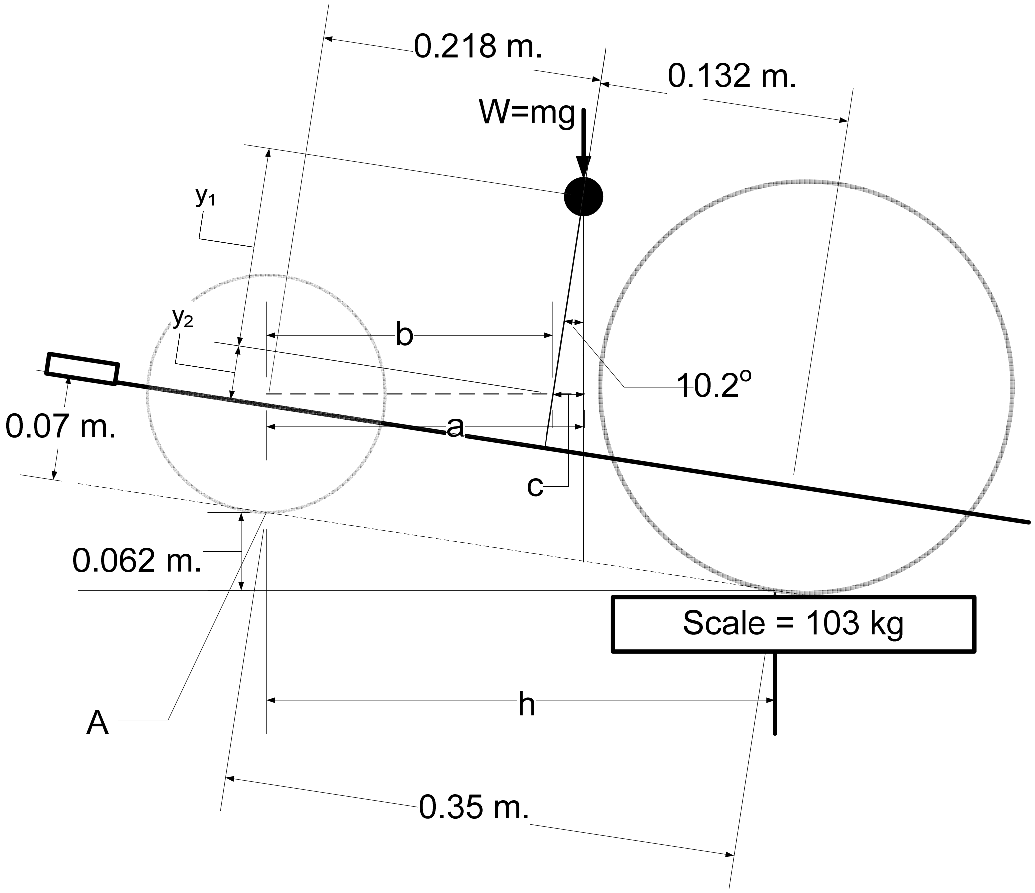


Figure 69: Appendix 1, schematic of a section of the robot after it has been raised 0.062 meters.

Solving the center of gravity problem is accomplished by using geometry. The first step is calculating the horizontal distance between the drive and castor wheels:

$$h = 0.35\cos(10.2) = 0.344m. \qquad \text{(A:3-3)}$$

The perpendicular distance between the castor wheel and the center of gravity can be calculated as shown:

$$\begin{aligned} &+\circlearrowleft\sum M_A = 0; \qquad 103gh - 148ga = 0 \\ &a = 0.240m \end{aligned} \qquad \text{(A:3-4)}$$

Referring to Figure 69, the vertical component of center of gravity from the ground can be shown as:

$$y_{cg} = y_1 + y_2 + 0.07 \qquad \text{(A:3-5)}$$

In order to calculate y_1, the distances (labeled b and c in Figure 69) must be calculated as shown:

$$b = {}^{0.218}\!\!\diagup_{\cos(10.2)} = 0.222m \qquad \text{(A:3-6)}$$

$$c = a - b = 0.018m \qquad \text{(A:3-7)}$$

y_1 and y_2 can be calculated as follows:

$$y_1 = {}^{c}\!\diagup_{\sin(10.2)} = 0.105m \qquad \text{(A:3-8)}$$

$$y_2 = {}^{0.218}\!\!\diagup_{\tan(10.2)} = 0.039m \qquad \text{(A:3-9)}$$

The vertical center of gravity, from the ground, is calculated to be:

$$y_{cg} = y_1 + y_2 + 0.07 = 0.215m \qquad \text{(A:3-10)}$$