MOTION PLANNING OF MOBILE ROBOT IN DYNAMIC ENVIRONMENT USING POTENTIAL FIELD AND ROADMAP BASED PLANNER

A Thesis

by

WAQAR AHMAD MALIK

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2003

Major Subject: Mechanical Engineering

MOTION PLANNING OF MOBILE ROBOT IN DYNAMIC ENVIRONMENT USING POTENTIAL FIELD AND ROADMAP BASED PLANNER

A Thesis

by

WAQAR AHMAD MALIK

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Sooyong Lee (Chair of Committee)	Darbha Swaroop (Member)
John E. Hurtado (Member)	Dennis L. O'Neal (Head of Department)

August 2003

Major Subject: Mechanical Engineering

ABSTRACT

Motion Planning of Mobile Robot in Dynamic Environment Using Potential Field and Roadmap Based Planner. (August 2003)

Waqar Ahmad Malik, B.Tech., Indian Institute of Technology, Kharagpur

Chair of Advisory Committee: Dr. Sooyong Lee

Mobile robots are increasingly being used to perform tasks in unknown environments. The potential of robots to undertake such tasks lies in their ability to intelligently and efficiently locate and interact with objects in their environment. My research focuses on developing algorithms to plan paths for mobile robots in a partially known environment observed by an overhead camera. The environment consists of dynamic obstacles and targets. A new methodology, *Extrapolated Artificial Potential Field*, is proposed for real time robot path planning. An algorithm for probabilistic collision detection and avoidance is used to enhance the planner. The aim of the robot is to select avoidance maneuvers to avoid the dynamic obstacles.

The navigation of a mobile robot in a real-world dynamic environment is a complex and daunting task. Consider the case of a mobile robot working in an office environment. It has to avoid the static obstacles such as desks, chairs and cupboards and it also has to consider dynamic obstacles such as humans. In the presence of dynamic obstacles, the robot has to predict the motion of the obstacles. Humans inherently have an intuitive motion prediction scheme when planning a path in a crowded environment. A technique has been developed which predicts the possible future positions of obstacles. This technique coupled with the generalized Voronoi diagram enables the robot to safely navigate in a given environment.

To my Mother and Father.

ACKNOWLEDGMENTS

I gratefully acknowledge the support, guidance and encouragement that I received from my advisor, Dr. Sooyong Lee, throughout the course of my studies. He managed to give me the confidence and the tools necessary to complete this work. I would like to thank my committee member, Dr. Darbha Swaroop, for his help, advice and encouragement. I would like to express my gratitude to Dr. John E. Hurtado for his careful reading of my thesis.

I want to thank Dr. N. K. Anand, the Graduate Program Director, for his support during my studies at Texas A&M University. He has been kind enough to act as a substitute in my thesis defense on very short notice.

This work benefited greatly from the interaction with all the members of the Robotics Laboratory at Texas A&M University. In particular I want to thank Mr. JaeYong Lee for his contribution in Gaussian estimation and collision avoidance.

Finally, I thank my parents for their love and support. I thank them for giving me the constant love, encouragement, inspiration and motivation without which this research would not be possible.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

A. Motivation

The common environments where robots are used can be categorized as dynamic or time varying. In these environments, there are several moving objects. Common examples of these environments will be manufacturing industry with machinery parts as the moving objects, an office or a museum where the moving objects will be humans or a road-system where the moving objects will be the vehicles. The robot has to negotiate and avoid these obstacles while moving towards its destination.

If mobile robots are to be used to solve various problems, it is necessary that they be able to autonomously compute a safe trajectory. Mobile robots have been used as museum guides and office assistants. They have also been used for *search and rescue* operations and for exploration of hazardous environments. For these systems it becomes important that the robot finds an autonomous path with least chance of collision with any objects and humans. While navigating in a crowded environment, a dynamic obstacle can block the path of the robot. The robot has to stop its motion until the obstacle moves away. To plan a successful path, the robot has to make predictions about the motion of the obstacle and change its path before it faces this situation.

Motion planning in an environment containing moving obstacles is difficult and computationally intensive, since it requires simultaneously solving the path planning and the velocity planning problems. The robot path planning problem can be stated as the computation of collision free paths between two locations in an environment

---

The journal model is *IEEE Transactions on Automatic Control.*

containing obstacles. Velocity planning consists of computing the velocity profile along a given path that minimizes motion time or energy and satisfies system dynamics and actuator limits.

The following section surveys the previous work related to this research. The last section presents the outline of the following chapters.

B. Previous Work

There are many difficulties associated with enabling the robot to see and interact with its environment. The robot has to recognize the objects around it and also has to identify them as obstacles and targets. In addition to this information, the robot has to find its own position and orientation to enable it to make intelligent decisions so as to solve the path planning problem. This problem of localization of mobile robots is an active research field and many techniques using laser range finders [1], sonar range finders, ultrasonic sensors [2], infrared sensors, dead reckoning [3], GPS and vision sensors have been used. In vision based sensing, the camera can be on-board or off-board. A network of cameras has also been used, providing a larger field of view and data fusion over space and time. This research uses a single overhead sensor placement. Such systems have been used in robot soccer competitions. Overhead camera placement is convenient for visual tracking because all objects are currently in view. Another great advantage of overhead camera is the self localization of the robot. There is no need for a separate localization algorithm.

After finding the position of robot and a description of its working environment, the robot path planning problem is solved. The robot path planning problem can be stated as the computation of collision free paths between two locations in an environment containing obstacles. Various collision free path planning algorithms

have been proposed in [4], [5]. The simplest case is of one robot present in a static and known environment which has to be moved from its current position to a final goal position. This thesis deals with an environment consisting of multiple, dynamic obstacles and a dynamic target.

The potential field method has been used extensively for mobile robot path planning in the last two decades:[6], [7]. The basic concept of the potential field approach is to compute a artificial potential field in which the robot is attracted to the target and repulsed from the obstacles [8]. The artificial potential field is used due to its computational simplicity. In [9], the mobile robot applies a force generated by the artificial potential field as the control input to its driving system. Potential field method for path planning has some limitations [10], namely, local minima, oscillations in the presence of obstacles, absence of passage between closely spaced obstacles and oscillations in narrow passages. There also exists a problem of goal non-reachability with obstacles nearby [11]. A hydrodynamic potential field has been utilized in [12] to guide a mobile robot towards the goal while avoiding obstacles. The workspace of the robot is compared with a flow field, with the path corresponding to a streamline. A potential field method for non-spherical body which simulates steady-state heat transfer with variable thermal conductivity has been proposed in [13].

This research extends the artificial potential approach to the case of dynamic motion planning. Future estimated positions of the obstacles also add to the repulsive potential, thus providing a path which considers the effect of dynamic obstacle. Future estimated position of the obstacles will have an uncertainty associated with it. Miura et al. [14] considers uncertainty only in linear velocity of the obstacles, thus giving rise to a one-dimensional probability distribution. In [15], the planner predicts future motion of obstacles by assuming that they will continue to move at the current velocity, and plans the next best action in space-time. Fiorini and Shiller

[16] used the concept of linear velocity obstacles for local motion planning. [17], [18] formulated a statistical estimation technique which generated the error ellipse based on the Gaussian probability distribution. These ellipses are used in estimating future robot position when the path is predefined.

Motion planning is important in the operation of autonomous robots. Simultaneous solution of the path planning and the velocity planning problems are required. The environment could be completely known (when the trajectory of the obstacles are known) or partially known (when obstacle trajectory is unknown or information about it is incomplete). In our case we have multiple, dynamic obstacles in a partially known environment. Due to the uncertain nature of the environment, a solution computed at time t_0 may be infeasible at a later time [19].

The geometric path calculated in the path planner does not contain any timing information but includes only spatial positions. Motion planning in dynamic environments was originally addressed by adding the time dimension to the robot's configuration space, assuming bounded velocities and known trajectories of the obstacles [20], [21], [22]. Reif and Sharir [20] solved the problem by searching a visibility graph [8]. Kant and Zucker [23] proposed that the avoidance of the moving obstacles can be done by adjusting the speed along the geometric path. Lee and Lee [24] developed a similar approach for two cooperating robots, and compared the effects of delay and velocity reduction on motion time. Fraichard [25] considered acceleration bounds and used a search in a state-time space to compute a velocity profile yielding a minimum-time trajectory. Reister and Pin [26], Renaud and Fourquet [27] and Yamamoto et al [28] solved the problem to find the time-optimal motion of two independently driven wheels type robots. Pledel and Bestaoui [29] found an optimal motion problem subject to various actuator constraints while the motion is constrained to an arbitrary path. Fiorini and Shiller [16] proposed a planning method based on velocity obstacles,

which maps the dynamic environment into the robot velocity space.

When the environment becomes more complex, such as presence of narrow paths, the potential field method of robot navigation is not so effective. In such a case roadmap based method is better to implement. An example of a roadmap based method is the generalized Voronoi diagram, the locus of points equidistant to two or more obstacles. The generalized Voronoi diagram is an extension of the Voronoi diagram, the set of points equidistant to two or more points in the plane. Generalized Voronoi diagrams have long been used as a basis for motion planning algorithms [30], [31]. The Generalized Voronoi diagrams has been used to guide the potential field planner in [32]. By following the boundary of a Voronoi cell, the robot will be guaranteed to remain at a maximum clearance from the obstacles enabling the robot to move in narrow pathways.

C. Organization

The thesis first discusses about the new Extrapolated Potential Field method for the trajectory generation of a mobile robot in the presence of dynamic obstacles and dynamic target. Comparison is made with the traditional artificial potential method. The chapter following that talks about probabilistic collision prediction and avoidance. Conditions for collision alarms are stated and error ellipses' for the mobile robot and the obstacles are generated and the combined probability distribution is found. Various kinematic and dynamic constraints of a differential-drive mobile robot are discussed and a velocity planning scheme is presented.

The problem of a mobile robot navigating in a cluttered environment is discussed next. This is solved with the generalized Voronoi diagram. Method for improvement in the Voronoi path is discussed. Method for long-term prediction of obstacle's posi-

tion is developed and planning in the presence of dynamic obstacles is considered.

The last chapter provides an overview of the experimental setup and provides the experimental and simulation results.

CHAPTER II

THE POTENTIAL FIELD BASED PATH PLANNER

A. Introduction

The potential field approach has been used extensively for mobile robot path planning. This method has been used by researchers because of its ability to find an elegant solution and its mathematical and computational simplicity. Although many form of potentials have been studied, the concept behind them is relatively simple. The basic concept of the potential field method is to fill the workspace with an artificial potential field in which the goal exerts an attractive force on the robot and every obstacle exerts a repulsive force. The vector sum of all forces give the resultant direction and speed of the robot's motion at any given position. Potential field method for path planning has some limitations, namely, local minima, oscillations in the presence of obstacles, absence of passage between closely spaced obstacles and oscillations in narrow passages.

B. The Traditional Artificial Potential Field Method

In the traditional artificial potential field methods, an obstacle is considered as a point of highest potential, and a goal as a point of lowest potential. The mobile robot always moves from a high potential point to a low potential point. The robot is assumed to be a point mass and moves in a two-dimensional workspace. Its position in the workspace is denoted by $\mathbf{q} = [x \quad y]^T$. The most commonly used attractive potential has the form, [8] [4] [10]

$$U_{att}(\mathbf{q}) = \frac{1}{2}\xi\rho^m(\mathbf{q}, \mathbf{q}_{goal}) \tag{2.1}$$

where ξ is a positive scaling factor, $\rho(\mathbf{q}, \mathbf{q}_{goal}) = \|\mathbf{q}_{goal} - \mathbf{q}\|$ is the distance between the body $\mathbf{q}$ and the goal $\mathbf{q}_{goal}$, and m=1 or 2.

For m=1, the attractive potential is conic in shape and the resulting attractive force has constant amplitude except at the goal where its the potential is singular. For m=2, the attractive potential is parabolic in shape. The corresponding attractive force is given by the negative gradient of the attractive potential

$$\mathbf{F}_{att}(\mathbf{q}) = -\nabla_q U_{att}(\mathbf{q}) = \xi\rho(\mathbf{q}_{goal} - \mathbf{q}) \tag{2.2}$$

which converges linearly towards zero as the robot approaches the goal.

The commonly used repulsive potential function has the form:

$$U_{rep}(\mathbf{q}) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(\mathbf{q},\mathbf{q}_{obs})} - \frac{1}{\rho_o}\right)^2, & \text{if } \rho(\mathbf{q}, \mathbf{q}_{obs}) \leq \rho_o \\ 0, & \text{if } \rho(\mathbf{q}, \mathbf{q}_{obs}) > \rho_o \end{cases} \tag{2.3}$$

where η is a positive scaling factor, $\rho(\mathbf{q}, \mathbf{q}_{obs})$ denotes the distance from the robot $\mathbf{q}$ to the obstacle, $\mathbf{q}_{obs}$denotes the position of the obstacle, and ρ_o is a positive constant denoting the distance of influence of the obstacle.

The corresponding repulsive force is given by

$$\begin{aligned} \mathbf{F}_{rep}(\mathbf{q}) &= -\nabla_q U_{rep}(\mathbf{q}) \\ &= \begin{cases} \eta\left(\frac{1}{\rho(\mathbf{q},\mathbf{q}_{obs})} - \frac{1}{\rho_o}\right) \times \\ \qquad \frac{1}{\rho^2(\mathbf{q},\mathbf{q}_{obs})}\nabla_q\rho(\mathbf{q}, \mathbf{q}_{obs}), & \text{if } \rho(\mathbf{q}, \mathbf{q}_{obs}) \leq \rho_o \\ 0, & \text{if } \rho(\mathbf{q}, \mathbf{q}_{obs}) > \rho_o \end{cases} \end{aligned} \tag{2.4}$$

The total force applied to the robot is the sum of the attractive force and the repulsive force

$$\mathbf{F}_{total} = \mathbf{F}_{att} + \mathbf{F}_{rep} \tag{2.5}$$

The mobile robot in this traditional potential field approach moves in the direc-

tion of this resultant force as in Eq. 2.6.

$$\dot{\mathbf{q}} = -\nabla_q(U_{att}(\mathbf{q}) + U_{rep}(\mathbf{q})) \tag{2.6}$$

C. The Extrapolated Artificial Potential Field Method

The traditional artificial method works well for static obstacles. This method does not incorporate any mechanism for dynamic obstacles. In a dynamic environment the path generated through the traditional approach will be safe but not logical. In order to find a better path in a dynamic environment, a new method named the *Extrapolated Artificial Potential Field* is proposed.

New potential functions for the obstacle and goal are proposed which take into consideration the extrapolated positions for the obstacles and the goal. The path of minimum potential which is generated thus considers the dynamic nature of the obstacles and goal.

The attractive potential is given by

$$\begin{aligned} U_{att}(\mathbf{q}) &= \sum_{j=0}^{n} \frac{1}{2}\xi_j \rho^m(\mathbf{q}, \mathbf{q}_{goal(j)}), \quad \text{if } \rho(\mathbf{q}, \mathbf{q}_{goal(0)}) \geq \rho_g \\ &= \frac{1}{2}\xi_0 \rho^m(\mathbf{q}, \mathbf{q}_{goal(0)}), \qquad \text{if } \rho(\mathbf{q}, \mathbf{q}_{goal(0)}) < \rho_g \end{aligned} \tag{2.7}$$

where n is the number of extrapolated images of the goal, ξ_0 is the positive scaling factor for the attractive potential of the goal and $\xi_1,\ \xi_2, \ldots,\ \xi_n$ are the positive scaling factor for the attractive potential for each extrapolated image of the goal, which becomes smaller as n increases, $\mathbf{q}_{goal(0)}$ is the present position of the goal and $\mathbf{q}_{goal(1)},\ \mathbf{q}_{goal(2)}, \ldots,\ \mathbf{q}_{goal(n)}$ are extrapolated position of the goal. When the robot is far away from the goal, it considers the future positions of the goal in planning its path. When it is near the goal $(\mathbf{q}_{goal(0)}) < \rho_g)$ it does not consider the future goal

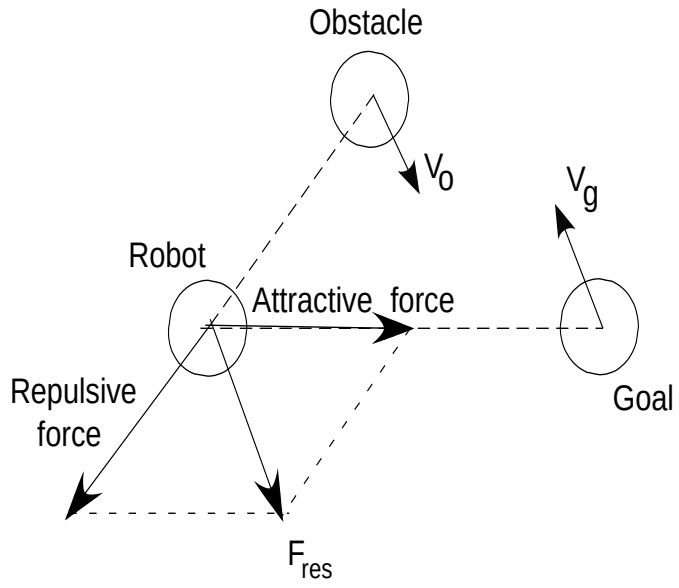


Fig. 1. Resultant direction of motion in traditional potential field

positions. This enables it to reach the goal exactly.

The repulsive potential is given by

$$U_{rep}(\mathbf{q}) = \sum_{j=0}^{n} \frac{1}{2}\text{flag} \times \eta_j \left(\frac{1}{\rho(\mathbf{q}, \mathbf{q}_{obs(j)})} - \frac{1}{\rho_o} \right)^2 \tag{2.8}$$

$$\text{flag} = \begin{cases} 1, & \text{if probability of collision} > P_{threshold} \\ 0, & \text{if probability of collision} < P_{threshold} \end{cases}$$

where n is the number of extrapolated images of the obstacle, η_0 is the positive scaling factor for the repulsive potential of the obstacle and $\eta_1,\ \eta_2, \ldots,\ \eta_n$ are the positive scaling factor for the repulsive potential for each extrapolated image of the obstacle, $\mathbf{q}_{obs(0)}$ is the present position of the obstacle and $\mathbf{q}_{obs(1)},\ \mathbf{q}_{obs(2)}, \ldots,\ \mathbf{q}_{obs(n)}$ are extrapolated position of the obstacles. $P_{threshhold}$ is the thresh-hold value of the probability of collision.

There will be the existence of the local minima. However, as the environment

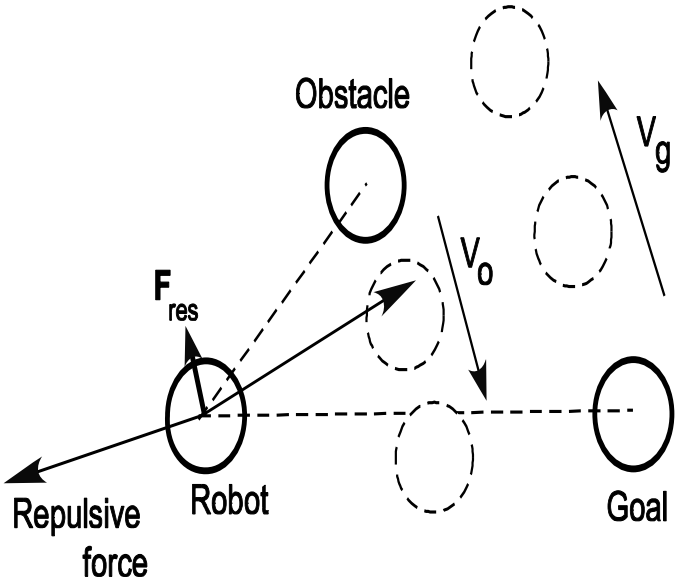


Fig. 2. Resultant direction of motion in extrapolated potential field

is dynamic, i.e. the obstacles and goal are continuously moving, we do not have to worry about the local minima as the mobile robot will not get stuck in the local minima. The mobile robot will stop its motion for that time instance and in the next time step the robot is given the appropriate motion command.

Fig. 1 and Fig. 2 show direction in which the robot moves using the traditional potential field approach and the extrapolated potential field approach respectively. The dotted circles in Fig. 2 depicts the extrapolated position of the obstacles and goal. The traditional potential field approach neglects the dynamic nature of the obstacles and goal.

The future positions of the obstacles and the target are predicted by using the history of past motion of the obstacles and targets respectively. The **extrapolation** is done by using piecewise cubic Hermite polynomial. Given a list of previous locations of the obstacle, future locations can be extrapolated.

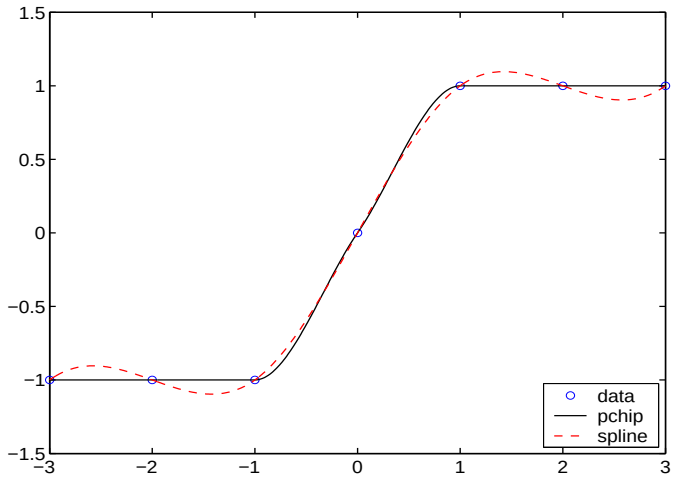


Fig. 3. Comparison between the use of splines and cubic Hermite polynomial

On each subinterval $\mathbf{t_k} \leq \mathbf{t} \leq \mathbf{t_{k+1}}$, $\mathbf{P(t)}$ is the cubic Hermite polynomial to the given values and certain slopes at the two endpoints. $\mathbf{P(t)}$ interpolates $\mathbf{x}$, i.e., $\mathbf{P(t_j)} = \mathbf{x_j}$, and the first derivative $\dot{\mathbf{P}}\mathbf{(t)}$ is continuous. $\ddot{\mathbf{P}}\mathbf{(t)}$ is probably not continuous; there may be jumps at the $\mathbf{t_j}$. The slopes at the $\mathbf{t_j}$ are chosen in such a way that $\mathbf{P(t)}$ preserves the shape of the data and respects monotonicity. This means that, on intervals where the data are monotonic, so is $\mathbf{P(t)}$; at points where the data has a local extremum, so does $\mathbf{P(t)}$.

The extrapolated values of $\mathbf{q}_{obs}$ is thus found.

The method of splines also constructs the function $\mathbf{S(t)}$ in almost the same way as $\mathbf{P(t)}$. However, spline chooses the slopes at the $\mathbf{t_j}$ differently, namely to make even $\ddot{\mathbf{S}}\mathbf{(t)}$ continuous. The effects of the two method can be seen in Fig. 3. Cubic Hermite polynomial has no overshoots and less oscillation if the data are not smooth. Either the cubic Hermite polynomial or the method of splines can be used for extrapolation purposes depending on the nature of motion of the obstacle and direction.

CHAPTER III

PROBABILISTIC COLLISION PREDICTION AND AVOIDANCE METHOD

A. Introduction

Based on the extrapolated artificial potential field we get a predetermined path for the mobile robot, which is a feasible solution at that given time. In a static environment, this solution will hold. In a dynamic environment, this trajectory may lead to collision. This is because the position of the obstacles and/or target keep changing. It is not necessary to calculate the repulsive potential due to the obstacles if the obstacle and the robot are not on a collision course. Smith et al. [17] and Durrant-White [18] formulated statistical estimation techniques using the Gaussian probability distribution. They generated the so called *error ellipse*, which depicts the uncertainty in the estimated position. These ellipses are used in estimating the robot's and the obstacles' positions

B. Collision Alarms

Before calculating the probability of collision, there needs to be an estimate of whether the robot and the obstacle are moving towards a possible collision. This is done by comparing the distance between the robot and the obstacles periodically. Comparison of the velocities of the robot and the obstacles is used to check the possibility of collision.

Collision Alarm #1 If the distance between mobile robot and moving obstacle at a given time t_c is less then predefined safety distance (d_s), which means $|\overrightarrow{RO}| < d_s$ in Fig. 4, first collision alarm is made.

Collision Alarm #2 If the collision alarm #1 is caused, then the predictor

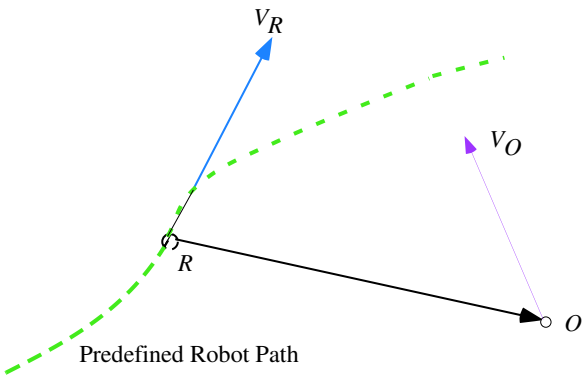


Fig. 4. Condition for alarming

examines the mobile robot's and obstacles' velocity and decides whether second alarm should be made or not. As shown in Fig. 4, O and R are the obstacle position and mobile robot position, respectively in the workspace at a given time t_c. V_o represents the velocity of the obstacle and V_r is the velocity of the robot. $\overrightarrow{RO}$ implies the vector from R to O and define $\hat{e}$ as its unit vector. Then inspection for the second alarming is made by categorizing into the following possible cases (Table I).

Table I. Conditions for collision alarm #2

Situation	**Result**
sign $(V_o \cdot \hat{e}) < 0$ and sign $(V_r \cdot \hat{e}) > 0$ and sign $(V_o \times \hat{e}) =$ sign $(V_r \times \hat{e})$	Alarm
sign $(V_o \cdot \hat{e}) > 0$ and sign $(V_r \cdot \hat{e}) > 0$ and $\lvert V_r \rvert > \lvert V_o \rvert$ and $((\cos^{-1}(V_o \cdot \hat{e}) > (\cos^{-1}(V_r \cdot \hat{e}))$	Alarm
sign $(V_o \cdot \hat{e}) < 0$ and sign $(V_r \cdot \hat{e}) < 0$ and $\lvert V_r \rvert < \lvert V_o \rvert$ and $((\cos^{-1}(V_o \cdot \hat{e}) < (\cos^{-1}(V_r \cdot \hat{e}))$	Alarm
In all other cases	No Alarm

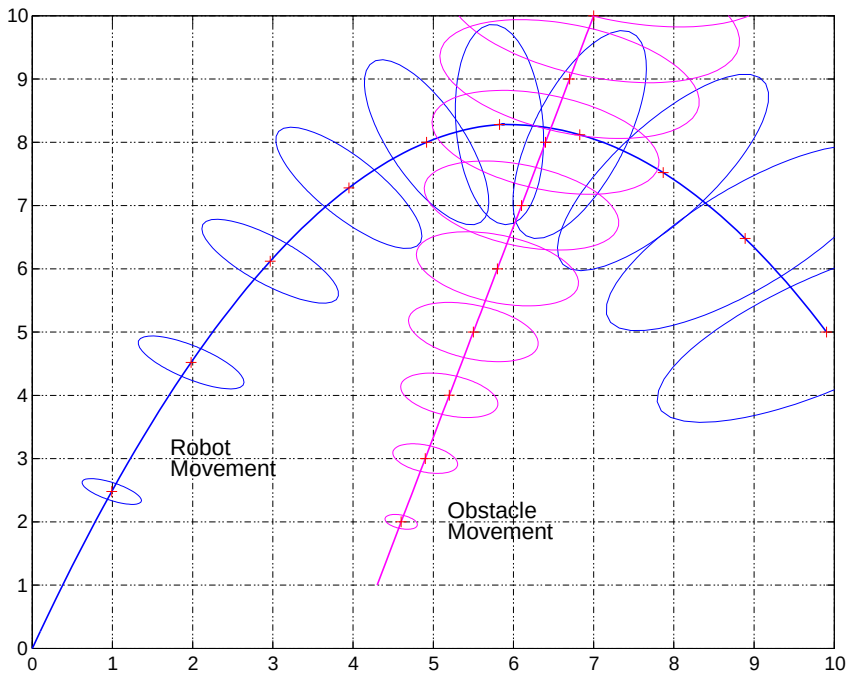


Fig. 5. Increase of error ellipse

C. Collision Probability

When both the alarm #1 and alarm #2 occur, the need arises to find the probability of collision. Based on this probability of collision, decision about necessary maneuver for obstacle avoidance is taken.

Based on the predetermined mobile robot path and obstacle information from sensors, collision probability at a certain time is made. The determination of whether the probability of observing the object is greater than a given threshold assumes that the probability distribution of our knowledge of the object's location is a multi variate (x, y, θ) Gaussian distribution. The contours of equal probability of this distribution

form ellipses that are centered at the mean location at any time instant.

The probability of collision is calculated by comparing predicted probability of the common regions of the ellipses (of a given confidence level) of the mobile robot and obstacles at future time t_e. However, the errors for mobile robot and obstacles increase as the moving objects travel, shown in Fig. 5. Fig. 6 shows the error ellipse (of a given confidence level) formed by the mobile robot and an obstacle. The shaded region shows the common region.

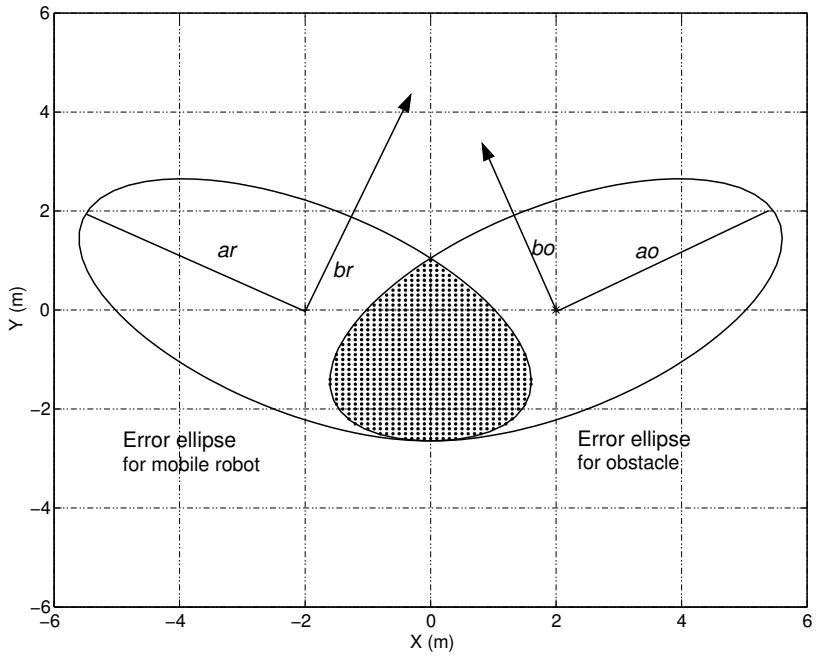


Fig. 6. Geometry of the ellipses

For the error ellipse of mobile robot, major axis length (a_r) and minor axis length (b_r) are both proportional to the velocity command and the angle changes for each future time step . In case of obstacle, past data sets are used to determine its mean

velocity. These can be summarized as follow:

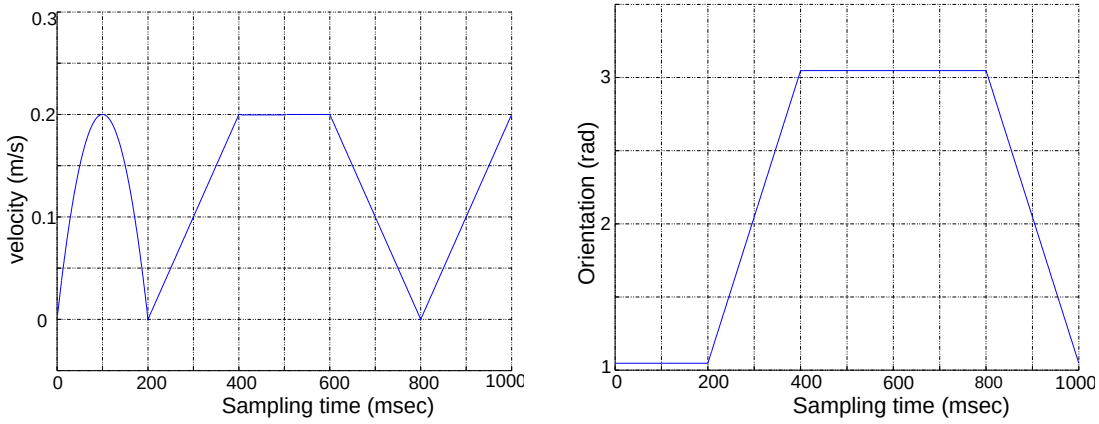
$$a_r = \sum_{i=0}^{m} \{(|v_{ri}| \times \alpha_{11}) + (|\Delta\theta_{ri}| \times \alpha_{12})\} \tag{3.1}$$

$$b_r = \sum_{i=0}^{m} \{(|v_{ri}| \times \alpha_{21}) + (|\Delta\theta_{ri}| \times \alpha_{22})\} \tag{3.2}$$

$$a_o = \sum_{j=1}^{n} \{(|v_{oj}| \times \beta_{11}) + (|\Delta\theta_{oj}| \times \beta_{12})\} \tag{3.3}$$

$$b_o = \sum_{j=1}^{n} \{(|v_{oj}| \times \beta_{21}) + (|\Delta\theta_{oj}| \times \beta_{22})\} \tag{3.4}$$

where m is the number of future steps when the possibility check is made for the mobile robot, and n is the number of past data required to calculate the mean velocity of the obstacle. α_{ij} and β_{ij} depends on the errors which are introduced due to factors, such as, interaction between the surface and the robot's wheel, slip, odometric errors and errors in implementing a given velocity command. Fig. 8 shows the evolution of the major and minor axes of the mobile robot for the inputs shown in Fig. 7.



(a) Velocity command (b) Orientation input

Fig. 7. Arbitrary control input for the mobile robot

Assuming the robot's and obstacles' positional errors are following the gaussian

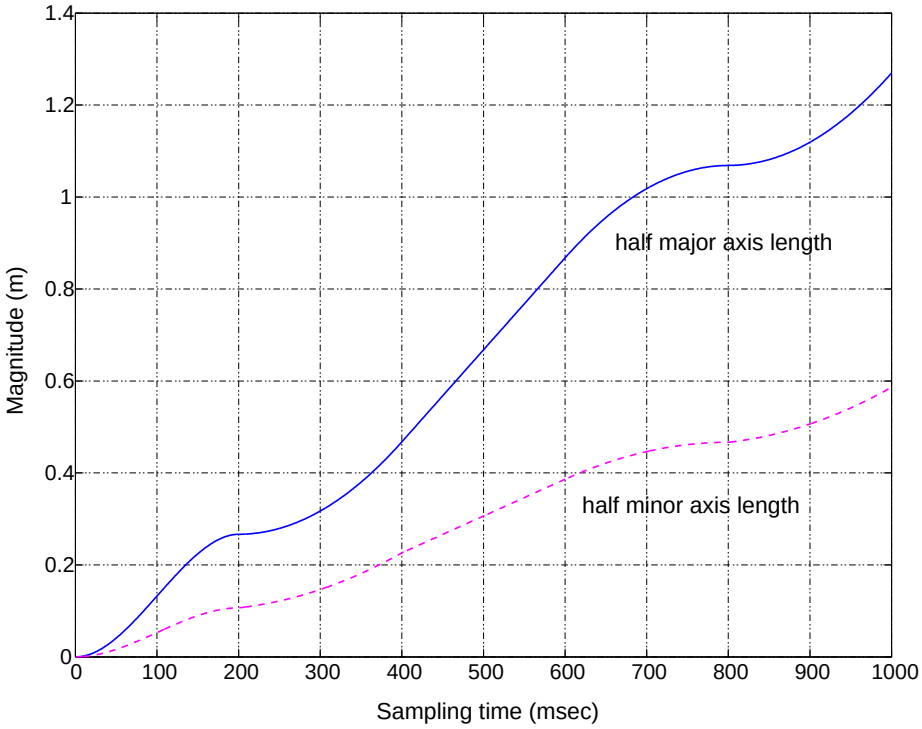


Fig. 8. Increment of major and minor axis lengths

distribution, the position profile of the moving objects can be described by:

$$P_r(\mathbf{x_r}) = \frac{1}{2\pi\sqrt{\sigma_{xr}^2\sigma_{yr}^2(1-\rho_r^2)}} e^{-\frac{1}{2(1-\rho_r^2)}\left[\frac{(x_r-\mu_{xr})^2}{\sigma_{xr}^2}+\frac{2\rho_r(x_r-\mu_{xr})(y_r-\mu_{yr})}{\sigma_{xr}\sigma_{yr}}+\frac{(y_r-\mu_{yr})^2}{\sigma_{yr}^2}\right]} \tag{3.5}$$

for the mobile robot and

$$P_o(\mathbf{x_o}) = \frac{1}{2\pi\sqrt{\sigma_{xo}^2\sigma_{yo}^2(1-\rho_o^2)}} e^{-\frac{1}{2(1-\rho_o^2)}\left[\frac{(x_o-\mu_{xo})^2}{\sigma_{xo}^2}+\frac{2\rho_o(x_o-\mu_{xo})(y_o-\mu_{yo})}{\sigma_{xo}\sigma_{yo}}+\frac{(y_o-\mu_{yo})^2}{\sigma_{yo}^2}\right]} \tag{3.6}$$

for the obstacle. ρ_r is the correlation coefficient for x_r and y_r. μ_{xr} and μ_{yr} are nominal mean values for x_r and y_r, respectively. ρ_o is the correlation coefficient for x_o and y_o. μ_{xo} and μ_{yo} are nominal mean values for x_o and y_o, respectively. $\mathbf{x_r}$ represents the mobile robot coordinates inside the robot's ellipse boundary at t_e and

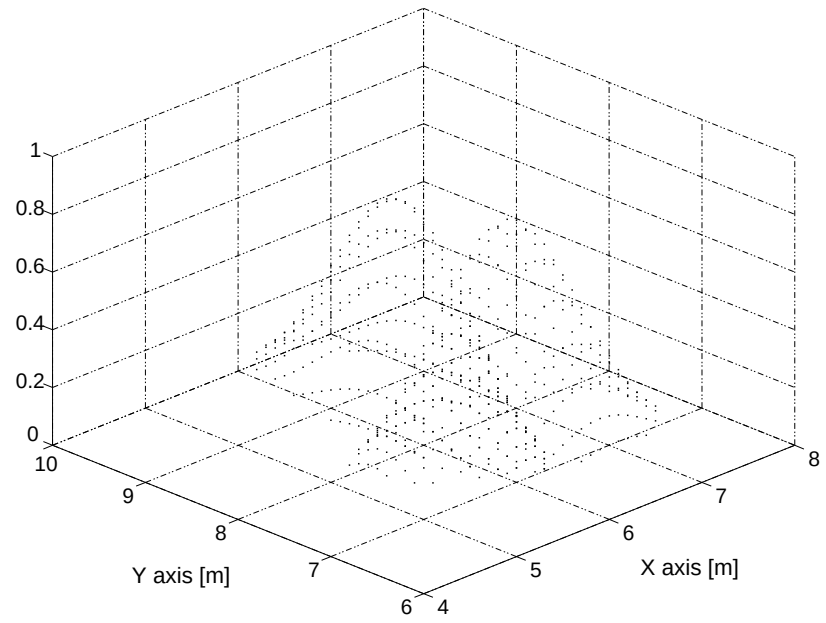


Fig. 9. Probability density function

$\mathbf{x_o}$ represents the coordinates of the obstacle inside the obstacle's ellipse boundary at the corresponding time.

Given the estimates of the ellipse major and minor axes (Eq. 3.1–3.4) and the orientation, it is possible to derive the probability density function parameters as shown in Appendix A.

If we define A as the set of coordinates where the mobile robot and the obstacle are in the common area of the two ellipses, then the probability of location of the robot and the obstacle in the area will be presented as:

$$P_{cr} = \int_A P_r(x_r, y_r) dA, \qquad \{(x_r, y_r) | (x_r, y_r) \in A\} \tag{3.7}$$

$$P_{co} = \int_A P_o(x_o, y_o) dA, \qquad \{(x_o, y_o) | (x_o, y_o) \in A\}, \tag{3.8}$$

where P_{cr} denotes the probability that the mobile robot will be inside the intersection

in Fig. 9 and Fig. 6. P_{co} implies the probability of the obstacle being inside the same area. Then the collision probability P_c can be computed by $P_c = P_{cr} \bigcap P_{co}$ as in Fig. 10.

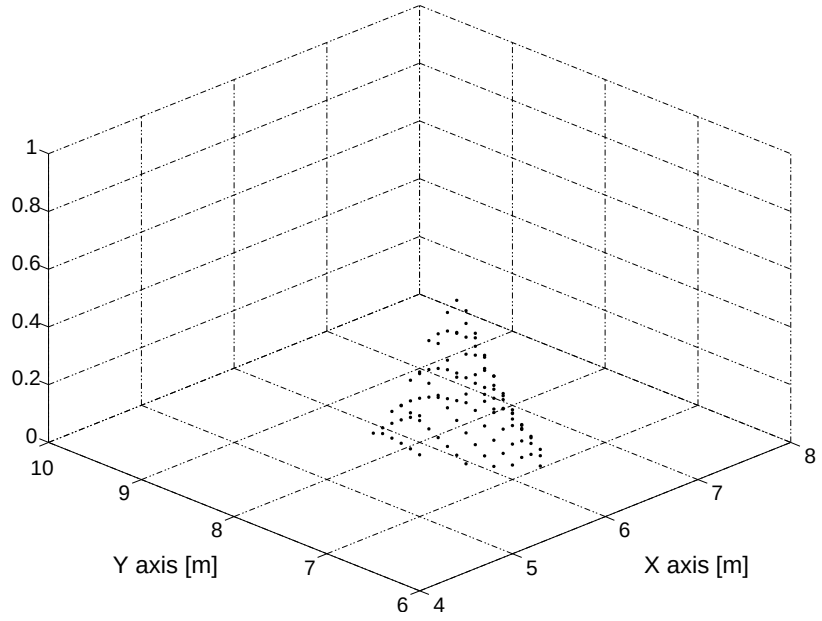
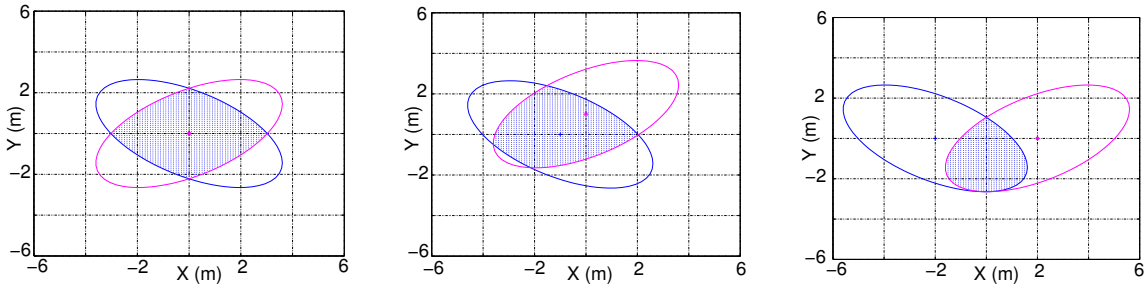


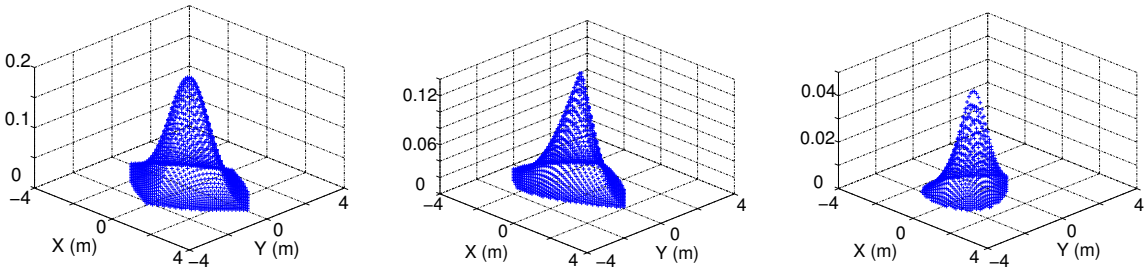
Fig. 10. Collision probability check

Fig. 11 show some example cases of collision check. Fig. 12 is the probability density function (p.d.f.) plotting for each case of the example cases and the numerical values of those probabilities are shown in Table II. Because the probability distribution is not uniform over the ellipse, the size of the common area does not directly represent the probability of collision. P_c increases as the common area get near to the centers of the two ellipses.



(a) example 1 (b) example 2 (c) example 3

Fig. 11. Examples of collision probability check



(a) example 1 (b) example 2 (c) example 3

Fig. 12. P.D.F. plotting of the examples in Fig. 11

Table II. P_c values in Fig. 12

Examples	P_{cr}	P_{co}	P_c
example 1	0.8909	0.8909	0.6274
example 2	0.8503	0.6791	0.3876
example 3	0.2439	0.2439	0.0857

D. Path Update

The obstacle path is estimated from data set obtained by the image processing program. Collision check and robot path regeneration are made based on the collision probability information.

If the probability of collision between the robot and the obstacle is less than some threshold value, the repulsive potential due to that obstacle is not calculated. When the target is static, it is not necessary to replan the path at each time step. The velocity of the robot can be changed along the path to avoid collision with the obstacles.

The original path is updated with collision probability check. If it is highly likely to have collision, i.e. having a probability larger than a certain value, the navigator replans the path to avoid collision. Three ways to avoid the possible collision are

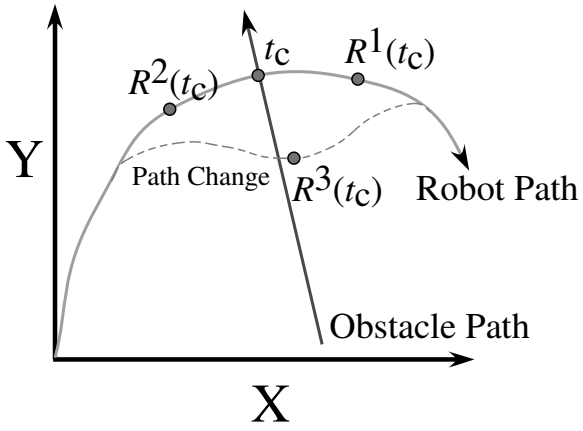


Fig. 13. Path updates

considered (Fig. 13). Let us denote t_c at the possible time of collision. The first way is to increase the robot velocity, so that the robot is located at R^1 at time t_c. The second one is to reduce the mobile robot velocity so that the mobile robot can

be located at R^2 at time t_c. Finally, the path itself can be modified to avoid the collision with obstacle. In this case, the mobile robot is expected to be R^3 at time t_c. There exist two constraints in considering the collision avoidance. The first one is the maximum velocity of the mobile robot, and the second one is the total travel time. The first constraint can be applied to the first case of possible collision avoidance strategy. The second one is for the second case of the strategies.

The decision for the different path updates is taken based on the position of the maximum probability of the collision. Fig. 14 shows the different scenarios. If the maximum probability occurs in the central shaded circle, then the path needs to be modified. The maximum probability cannot occur in regions 2 and 4. If the maximum probability occurs in region 3, then the robot velocity should be reduced i.e. the robot should allow the obstacle to pass. If the maximum probability occurs in region 1, then the robot velocity should be increased.

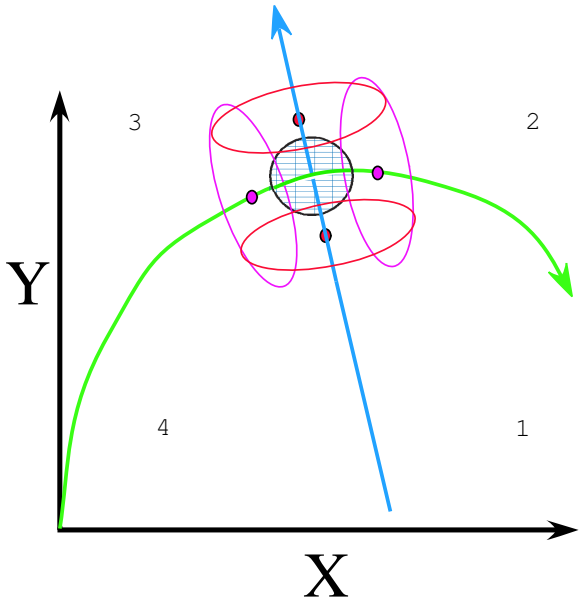


Fig. 14. Condition for applying path updates

CHAPTER IV

TIME OPTIMAL MOTION

A. Introduction

The potential field approach and the roadmap based approach provides a collision free path for the robot. The trajectory generated will be optimal based on some cost function. The generation of these trajectories did not consider the kinematic and dynamic constraints of the mobile robot. This chapter discusses the different kinematic, dynamic and actuator constraints and provides a strategy for time-optimal velocity planning along the given trajectory.

B. Description of Trajectory

The path obtained by the potential field approach and the roadmap based approach describes the robot motion in space. This path is not obtained as an analytical function. It is specified as a finite number of points. An analytical expression for this path is found by using a curve fitting technique which ensures that the curve obtained is atleast C^2 smooth. The path is a function of x, y, θ where x, y is the position of the robot and θ is the orientation. The path is then represented as a parameterized curve $r = r(s)$, s is a scalar "path parameter". It is defined as length along a specified robot's path. The trajectory is obtained from the path by specifying the "path parameter" as a function of time. Differentiating the path function twice we obtain:

$$\dot{r} = r_s \dot{s} \quad \text{and} \quad \ddot{r} = r_{ss} \dot{s}^2 + r_s \ddot{s} \tag{4.1}$$

where r_s is the vector tangent to the path and r_{ss} is the curvature vector obtained by differentiating r_s with respect to s. The vector tangent, r_s is along the orientation of

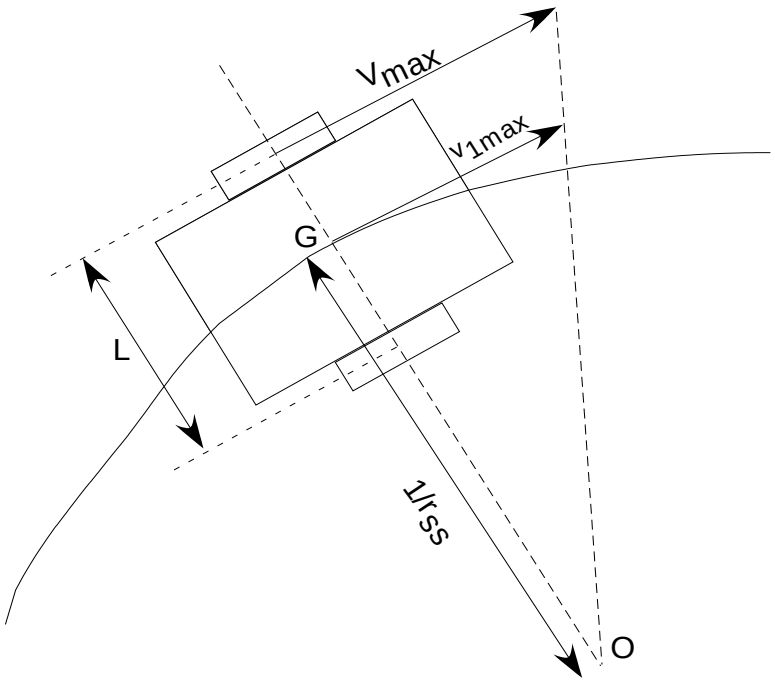


Fig. 15. Kinematic velocity constraints

the robot.

C. Kinematic and Actuators Constraints

The mobile robot is a kinematic mechanism composed of the body and the rolling wheels. Its kinematics can be modelled based on the assumption that the wheels are purely rolling. The robot is assumed to move on a flat surface. The length of the axle is L. The robot has two independently driven wheels. For any segment of the path, we get constraints on the velocity of the robot.

Mechanical speed, acceleration and deceleration are limited due to the actuator constraints. The capabilities of each individual actuator driving the two wheels are limited. These actuator constraints limit the torque applied to each wheel. This limi-

tation will also cause a limitation on the wheels speed and acceleration, the maximum values of which are obtained by making approximations.

- The maximum value of the acceleration (a_{max}) of each individual wheel cannot be higher than the maximum servomotor's acceleration.
- The deceleration must be less than the maximum permissible by the electromagnetic motor brakes.

The center of gravity is assumed to be in the middle of the wheel axis (point G in Fig. 15). When the robot moves along a section of the path with curvature r_{ss}, we get a limit on the maximum velocity and acceleration of the center of gravity of the robot.

$$v_{1max} = \frac{2V_{max}}{2 + Lr_{ss}} \tag{4.2}$$

$$a_{1max} = \frac{2a_{max}}{2 + Lr_{ss}} \tag{4.3}$$

where V_{max} is the maximum speed of each wheel, a_{max} is the maximum acceleration of each wheel, L is the distance between the two wheels, and v_{1max} and a_{1max} is the upper limit on the velocity and acceleration of the center of gravity of the robot due to the kinematic and actuator constraints.

D. Dynamic Constraints

Whenever the robot moves along any curved path, there is a lateral centrifugal force acting on the robot. This lateral force can cause the robot to slip sideways. This lateral force can also cause deformation of the the rubber wheel. This causes a change in the distance between the wheels. This deformation can cause the robot to deflect

from the predefined path. To prevent this we need to restrict the lateral force below some maximum value. This imposes a restriction on the velocity of the robot at each radius of curvature. The constraint on the velocity is given by the equation,

$$v_{2max} = \sqrt{\frac{\frac{\mu_{lat}(F_{z1} + F_{z2})}{M}}{\det r_{ss}}} \tag{4.4}$$

where μ_{lat} is the coefficient of friction between the wheel and the ground in the lateral direction, M is the mass of the robot and F_{z1} and F_{z2} are the normal reaction at the two traction wheel.

E. Generation of Time-Optimal Velocity Profile along the Trajectory

The path is represented as a parameterized curve $r = r(s)$ as explained in Section B. This path is then discretized into equal length segments, r_i. Kinematic and actuator constraints and 'collision avoidance' scheme, discussed in the previous section, assigns an upper bound on the velocity and the acceleration.

The constraints in velocity and acceleration can be converted in terms of the "path parameter" s :

$$\begin{aligned} \dot{s}_{min} &\leq \dot{s} \leq \dot{s}_{max} \\ \ddot{s}_{min} &\leq \ddot{s} \leq \ddot{s}_{max} \end{aligned} \tag{4.5}$$

Bounds on the slope of any trajectory in the phase-plane (s – $\dot{s}$) can be expressed in the form

$$\kappa_{min}(s, \lambda) \leq \kappa(s, \lambda) \leq \kappa_{max}(s, \lambda) \tag{4.6}$$

Minimization of travel time along the given trajectory requires that the mobile robot should try to move with the maximum allowable velocity along the path. The optimum velocity pattern is obtained by projecting the velocity $\dot{s}$ along the phase plane s – $\dot{s}$. This technique was first proposed by Kang et al. [33] for finding the

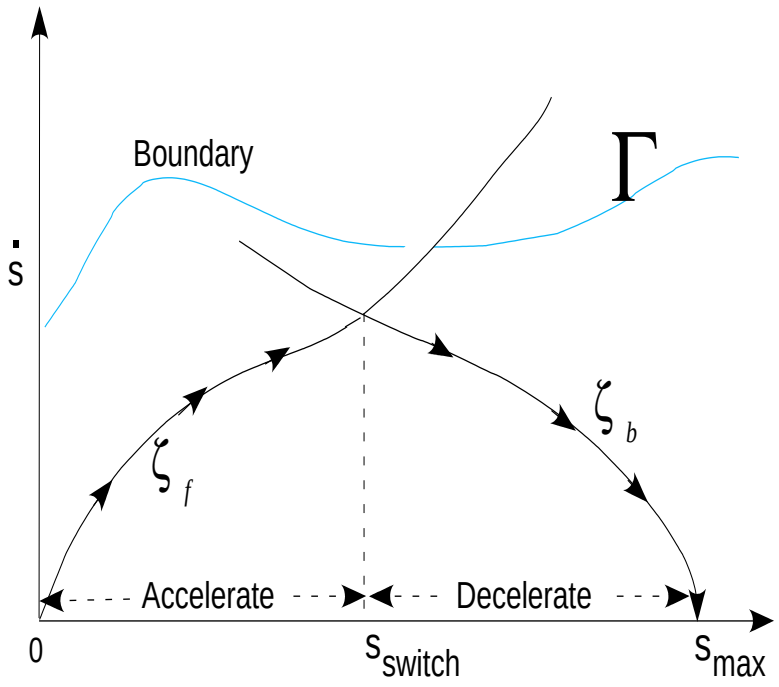


Fig. 16. Generation of optimal velocity

minimum time control of robotics manipulator with geometric path constraints. Yamamoto et al. [28] used this algorithm for planning the motion of mobile robots and gave it the name MTTP (Minimum Time Trajectory Planning). Formulation of the procedure is provided in Appendix B.

Given the bounds on velocity and acceleration, and subsequently on $\dot{s}$, $\ddot{s}$, the algorithm to calculate the optimal velocity (time minimizing) along the given trajectory can be constructed by the following steps.

step 1 Start at $s = 0$ and $\lambda = v_{initial}$ and construct a trajectory that has the maximum slope value (κ_{max}). Continue this curve until it hits the boundary (Γ) of the admissible region or goes past $s = s_{final}$. If the curve hits the boundary of the admissible region, extend the curve along this boundary, if possible. Call

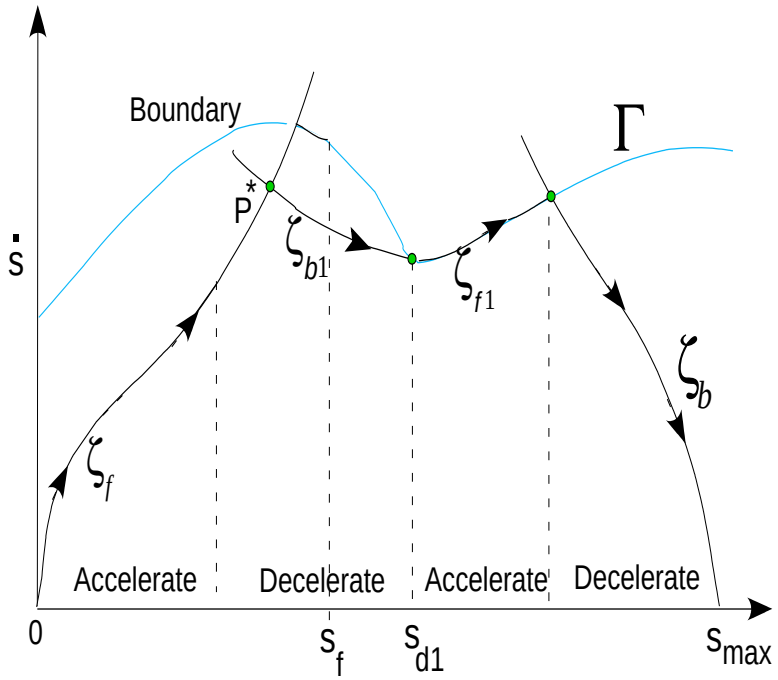


Fig. 17. Complete optimal trajectory

this curve ζ_f.

step 2 Construct a second trajectory that starts at $s = s_{max}$ and $\lambda = v_{final}$ and proceed backwards with maximum deceleration (slope value (κ_{min})). Continue this curve until it hits the boundary (Γ) of the admissible region or goes past $s = 0$. If the curve hits the boundary of the admissible region, extend the curve along this boundary, if possible. Call this curve ζ_b.

step 3 If the two trajectories intersect, then optimal velocity profile is achieved (Fig. 16). The point where the two curves intersect is the switching point. The optimal velocity profile consists of the first (accelerating) curve from $s = 0$ to $s = s_{switch}$ and the second (decelerating) curve from $s = s_{switch}$ to $s = s_{max}$.

step 4 If the two trajectories do not intersect (Fig. 17), it means that the curves

hits the admissible boundary curve and the slope of the admissible boundary (in some region) is outside the limits imposed by Eq. 4.6. Let s_f be the point where ζ_f ends. Starting at s_f move along the boundary (Γ) until the slope is within the limits imposed by the constraints. Call this point s_{d1}.

step 5 Construct a decelerating curve ζ_{b1} backward from s_{d1} until it intersects an accelerating curve. This intersection provides a switching point (Point P^* in Fig. 17.)

step 6 Construct an accelerating curve ζ_{f1} forward from s_{d1} until it intersects the decelerating curve ζ_b, or leaves the admissible region. If it hits the curve ζ_b, the algorithm terminates. If the curve leaves the admissible region, then goto **step 4**.

This algorithm gives a sequence of accelerating and decelerating curves ζ_f, ζ_{b1}, ζ_{f1}, ζ_{b2}, ζ_{f2},...,ζ_b, which gives the optimal velocity profile along a given path.

CHAPTER V

THE ROADMAP BASED PLANNER

A. Introduction

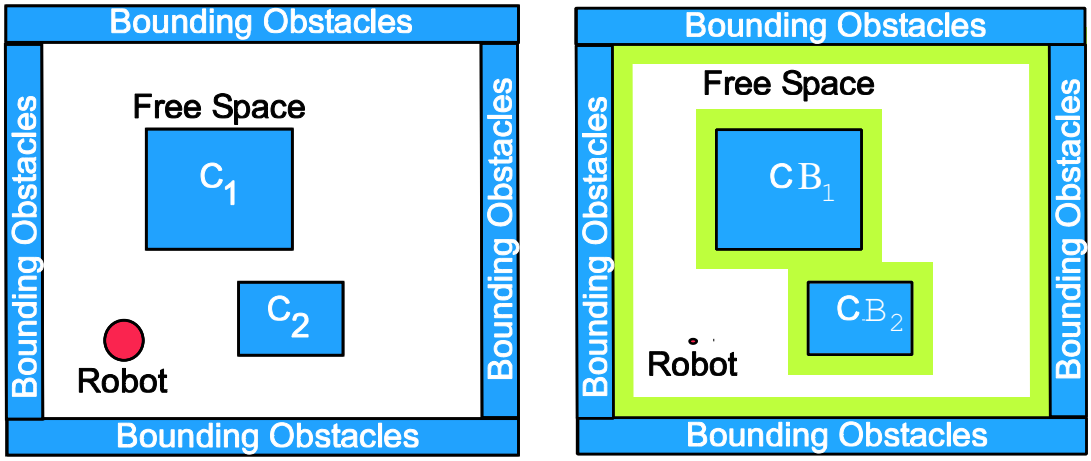
When the environment becomes more complex, such as the presence of narrow paths, the potential field method of robot navigation is not so effective. The problem of local minima may become more common due to many objects and their geometric configurations. The robot will not move between closely spaced obstacles. The robot has an oscillatory motion when it travels in narrow passages. When the environment becomes complex, roadmap based method is better to implement. We chose the roadmap based method over other methods due to its concise representation. Roadmap based methods are concise as they do not require the entire workspace to be discretized into smaller grids. An example of a roadmap based method is the generalized Voronoi diagram, the locus of points equidistant to two or more obstacles. The generalized Voronoi diagram is an extension of the Voronoi diagram, the set of points equidistant to two or more points in the plane. Generalized Voronoi diagrams have long been used as a basis for motion planning algorithms [30], [31]. By following the boundary of a Voronoi cell, the robot will be guaranteed to remain at a maximum clearance from the obstacles enabling the robot to move in narrow pathways.

B. Representation of the Environment

The robot is assumed to be circular and operates in a work space $\mathcal{W}$ which is a subset of an two-dimensional planar Euclidean space. The work space $\mathcal{W}$ is populated by polygonal obstacles $C_1, ..., C_n$, which are closed sets.

For the trajectory generation we need to model the robot as a point. This is

done by extending the obstacle's dimensions by a length equal to the radius of the robot. These expanded obstacles, $CB_1, ..., CB_n$, are called the C-obstacles. The set of points where the robot is free to move is called the free space and is defined as $\mathcal{FS} = \mathcal{W} \setminus \bigcup_{i=1}^{i=n} CB$. (see Fig. 18).



(a) Work Space (b) Configuration Space

Fig. 18. Representation of the environment

It is assumed that the robot operates in a bounded, connected subset of the free space $\mathcal{FS}$. This subset is bounded by obstacles. By working in the configuration space of the robot, the calculation of the trajectories becomes easier.

C. The Generalized Voronoi Diagram

For each obstacle CB_i, define a distance function $d_i(\mathbf{x}) = \text{dist}(CB_i, \mathbf{x})$. The Voronoi region of CB_i is the set

$$VR_i = \{x \mid d_i(\mathbf{x}) \leq d_j(\mathbf{x}) \forall j \neq i\}$$

The Generalized Voronoi Diagram (GVD) (Fig. 19) is a collection of the Voronoi regions. The GVD partitions the space into cells.

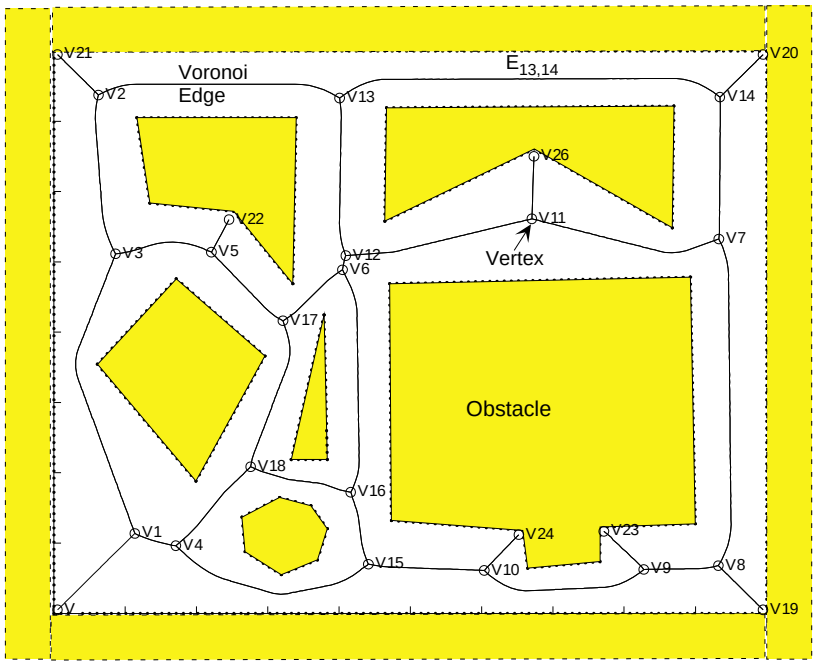


Fig. 19. Generalized Voronoi diagram

In two-dimensional planar case, the intersection of two Voronoi regions is a Voronoi edge. The intersection of at least two Voronoi edges is a Voronoi vertex. The Voronoi edge will a straight line or a parabolic curve segment. A straight line is the set of configurations that are closest to the same pair of obstacles' edges or the same pair of obstacles' vertices. A parabolic curve segment is the set of configurations that are closest to the same pair consisting of an obstacles' edge and a vertex. These pair of the obstacles' edge/edge, edge/vertex or vertex/vertex determines the equation of the algebraic curve forming the Voronoi edge.

The set of Voronoi edges and Voronoi vertex form the Generalized Voronoi Graph (GVG). For a planar case, the GVG will be connected. The GVG is accessible from any point in the $\mathcal{FS}$, therefore it is possible to move the robot from any initial point

to any final point in the $\mathcal{FS}$.

The GVG reduces the motion-planning problem by one dimension. In the planar case the path planning reduces to searching for a solution in a one-dimensional network of curves lying in $\mathcal{FS}$.

D. Prediction of Obstacle's Position

In the presence of dynamic obstacles, the robot has to predict the motion of the obstacles. Humans inherently have an intuitive motion prediction scheme when planning a path in a crowded environment. When crossing a road humans observe any incoming traffic and effectively predict its future position and decide whether it is safe to cross the road or not.

Prediction methods can give satisfactory results for one-step ahead prediction. For the robot navigation task it would be more useful to have many-steps ahead prediction. This would give the robot sufficient time and space to perform the necessary maneuvers to avoid obstacles and change its route from a geometrically shortest path to a path which will be short and with less obstacles.

If the movement of obstacles is considered to be random, it is unlikely that any available prediction method will give satisfactory results for many-steps ahead prediction. If we consider humans as the obstacles it may be assumed that the humans do not move aimlessly but with the intention of reaching a specific location. Hence, a possible approach for long term prediction would be to define 'points of interest' (POI) in the environment. In an office environment desks, doors, telephone, tool-rack and chairs are objects that may be considered as POI. If the POI of an environment are defined, then the long-term prediction could refer to the prediction of which POI a human is targeting. It is assumed that the POI are fixed and the obstacle moves

Table III. Algorithm for prediction of obstacle's position

Input: $\tilde{G} = [V, E], P_{obs}$, POI's

1. Find nearest edge E_{ij} and nodes $i,\ j$ for the obstacle with position P_{obs}
2. Using past position data find node from which the obstacle is moving away ($from_{node}$)
3. Using past position data find node towards which the obstacle is moving (to_{node})
4. Construct set S of all POI which have the shortest path from to_{node}
5. Calculate initial probability and time of entry and exit of each edge (based on S)
6. while $P_{obs} \neq$ POI's
   - Find nearest edge E_{ij} and nodes $i,\ j$ for the obstacle with new position P_{obs}
   - Find new $from_{node}$ and to_{node}
   - If new $from_{node}$ and to_{node} is different from old $from_{node}$ and to_{node}
     - Construct set S' of all POI which have the shortest path from new $from_{node}$ and the path to the POI contains to_{node}
     - Let new $S = S' \bigcap S$
     - Calculate new probability and time of entry and exit of each edge (based on S)
     - *New path for the robot needs to be calculated*

towards only one POI. There can be cases that a human moves randomly with no intention. In such cases, no long term prediction can be made. By considering that the obstacles also move approximately along the shortest Voronoi paths, the nodes of the Voronoi diagram can be assigned a probability of an obstacle trying to reach that node. Considering that the obstacle moves at nearly constant speed we can also predict the time an obstacle will reach a given node. This combined probability and knowledge of time allows us to set the cost function. Hence, we can predict with some probability the position of an obstacle in the long term future. The algorithm for prediction of obstacle's position is given in Table III.

E. Cost Function for Trajectory Generation

There will be many trajectories from the robot initial position ($\mathbf{x_{ri}}$) to the robot final position ($\mathbf{x_{rf}}$). In order to calculate the optimal path, we need to form a cost function (J) and search for a path along which J will be minimized. The cost will consist of two parts, namely the static and the dynamic cost functions i.e $J = J_{static} + J_{dynamic}$. The static cost function will be due to the static obstacles and will remain fixed and not change with time. The dynamic cost function is due to the dynamic obstacles and will change with time.

A cost is associated with each Voronoi edge (E_{ij}). The accumulated cost from 'start' to 'end' has to be minimized.

The first factor that needs to be considered is the total length of the path. A shorter path will be given preference over a longer one. Thus cost associated with a Voronoi edge (E_{ij}) should be proportional to its length.

$$J^{ij}_{static} \propto l_{ij} \tag{5.1}$$

where l_{ij} is the length of the Voronoi edge E_{ij}, $(i,\ j)$ are two adjacent Voronoi vertex (node). If $(i,\ j)$ are not adjacent then $l_{ij} = \infty$

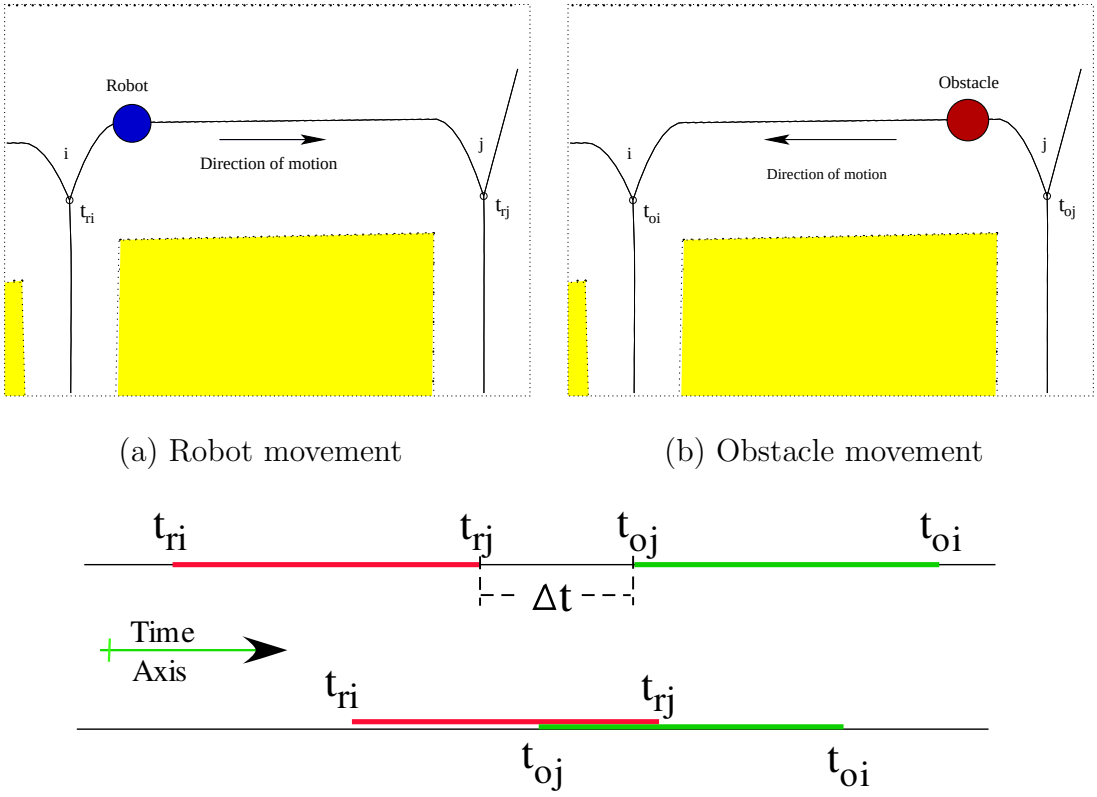
When a robot is moving through a narrow passageway, it has to follow the path precisely. In order to do so, it has to localize after it has moved a short distance. Due to this the robot has to reduce its speed. This is similar to a human trying to walk through a crowded corridor or on a narrow ledge. Therefore,

$$J^{ij}_{static} \propto \frac{1}{d^{min}_{ij}} \tag{5.2}$$

where d^{min}_{ij} is the minimum distance from a Voronoi edge E_{ij} to an obstacle, $(i,\ j)$ are two adjacent Voronoi vertex (node). If $(i,\ j)$ are not adjacent then $d^{min}_{ij} = 0$

If we consider dynamic obstacles, the robot should give preference to a path on which it has less probability of encountering an obstacle. In Section D, a method to predict the long-term future position of an obstacle is presented. The probability of an obstacle of being at node k at time t_{ok} is p_k. Let the robot reach node k at time t_{rk}.

Consider an edge on which the robot is trying to move from node $i \rightarrow j$ and the obstacle is trying to move from node $j \rightarrow i$ (Fig. 20). The probability (p_{ij}) of the obstacle of being on the edge E_{ij} is p_i. The robot reaches node i and j at time t_{ri} and t_{rj} respectively. Similarly, the obstacles reaches nodes j and i at time t_{oj} and t_{oi} respectively. Let $t_o = [t_{oj}, t_{oi}]$ and $t_r = [t_{ri}, t_{rj}]$. If $t_r \cap t_o$ exists, then the robot and obstacle will be on a collision course with probability p_{ij}. The separation between the two sets will be 0. Let Δt be the separation between these two sets $t_o,\ t_r$. If $\Delta t > 0$ then, in the ideal case, collision does not occur on that edge. If both the obstacle and the robot are moving from node $i \rightarrow j$, then $t_o = [t_{oi}]$ and $t_r = [t_{ri}]$ and $p_{ij} = p_j$. The



(a) Robot movement

(b) Obstacle movement

(c) Time axis

Fig. 20. Calculation of dynamic cost

cost function associated with a Voronoi edge (E_{ij}) will be proportional to,

$$J^{ij}_{dynamic} \propto \begin{cases} p_{ij}, & \text{if } \Delta t = 0 \\ 0, & \text{if } \Delta t \neq 0 \end{cases} \tag{5.3}$$

From Eq. 5.1–5.3, the overall cost function becomes,

$$J_{ij} = \begin{cases} \alpha_1 l_{ij} + \alpha_2 \frac{1}{d^{min}_{ij}} + \alpha_3 p_{ij}, & \text{if } \Delta t = 0 \\ \alpha_1 l_{ij} + \alpha_2 \frac{1}{d^{min}_{ij}}, & \text{if } \Delta t \neq 0 \end{cases} \tag{5.4}$$

where $\alpha_1,\ \alpha_2,\ \alpha_3$ are constants which needs to be assigned with regard to the performance required.

F. Trajectory Generation

1. Absence of Dynamic Obstacles

In the absence of dynamic obstacles, the Dijkstra's shortest path algorithm is used to find the required trajectory. Dijkstra's algorithm searches for the shortest path in a weighted directed graph $G = [V, E]$, where all edge weights are nonnegative.

Let the robot's initial position be ($\mathbf{x_{ri}}$) and it's final position be ($\mathbf{x_{rf}}$). Find the nearest point from these positions to the Voronoi graph. Call these ri and rf. Add ri and rf as nodes to the Voronoi graph and find the new cost functions. Using Dijkstra's algorithm, find the shortest path from node ri to node rf. Let these be the set of nodes $\eta = [\ ri,\ v_p, ..., \ v_q,\ rf\ ]$. Thus the minimum path, in the absence of dynamic obstacles, is [$\mathbf{x_{ri}}$, η, $\mathbf{x_{rf}}$] (Fig. 21).

The trajectory which we get in Fig. 21 is along the Voronoi edges and thus has maximum clearance from the obstacles. In open spaces, this trajectory will not be best solution available. To get a shorter path, we draw circles at the nodes (Fig. 22(a)). These circles are the circles with maximum radius which have the center at the nodes

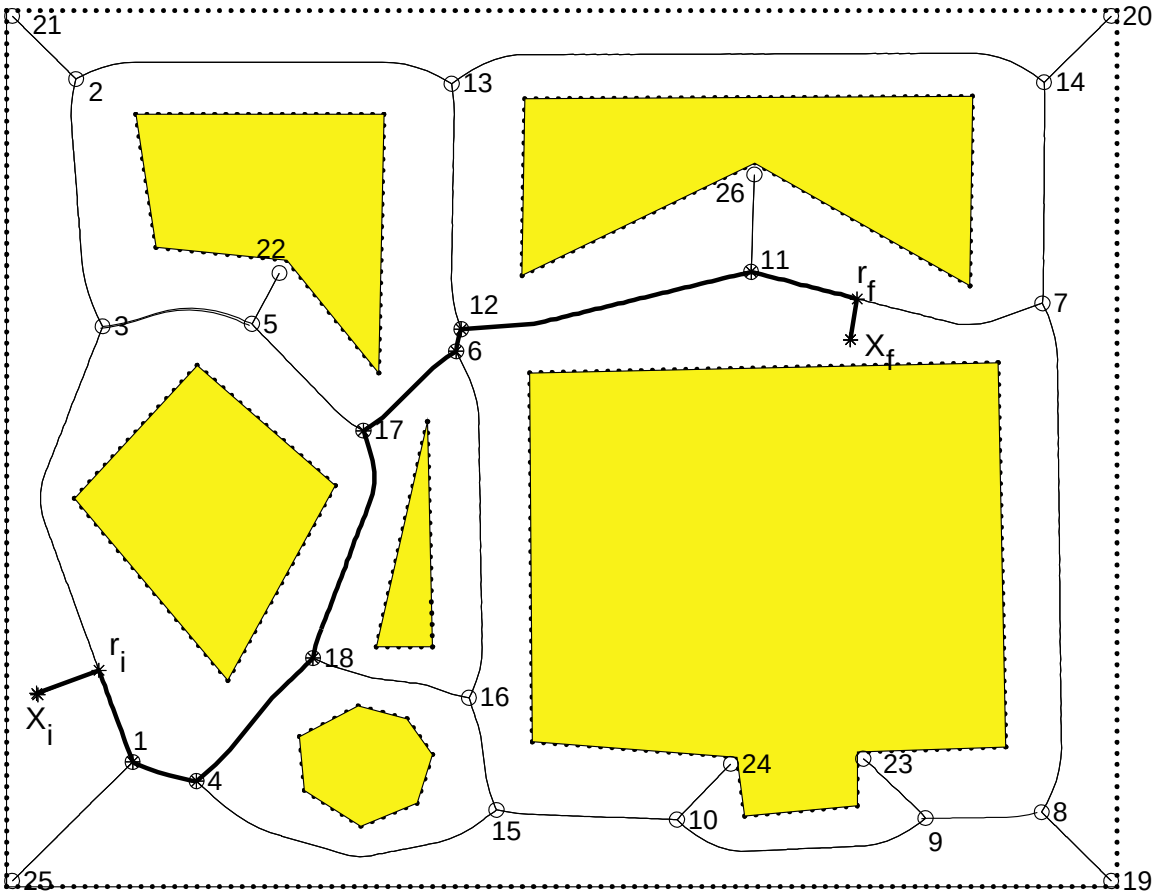
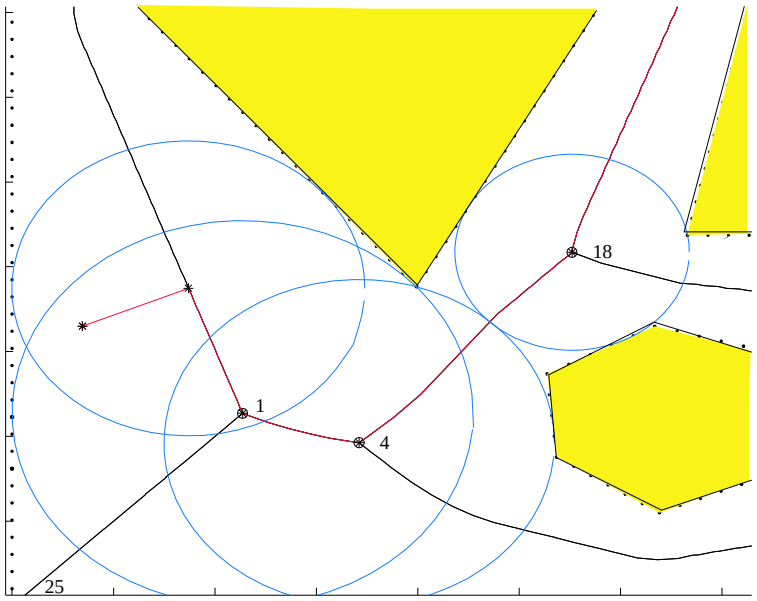
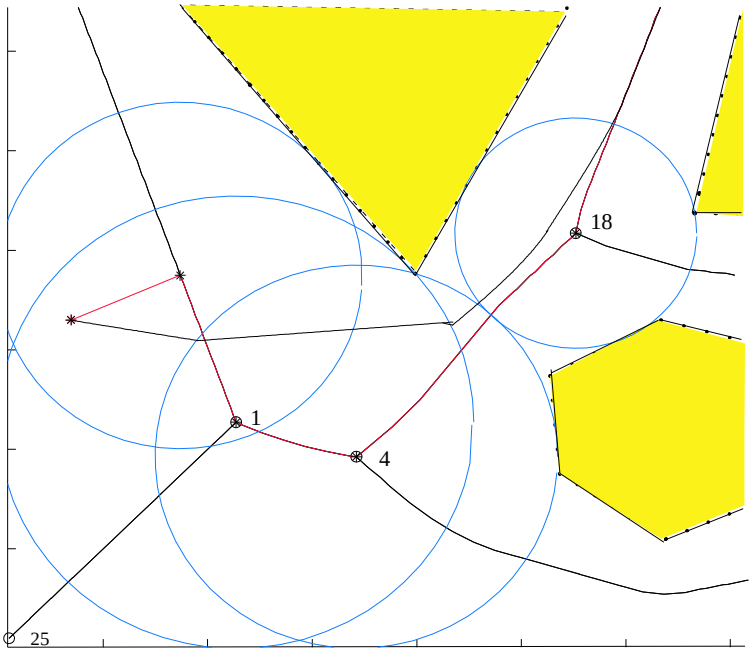


Fig. 21. Shortest path along Voronoi edges

and lie in the free space $\mathcal{FS}$. All points within these circles will also lie in the $\mathcal{FS}$. We find the intersection of the calculated path with this circle, and update the path, by joining the two points where the original cost minimizing path cut these circles. We repeat this procedure along all the nodes in the original path. The resultant path is shown in Fig. 22(b) and Fig. 23. The path is smoothened by fitting a bezier curve through the points in the path.



(a) Circles at nodes



(b) The updated path

Fig. 22. Method for path update

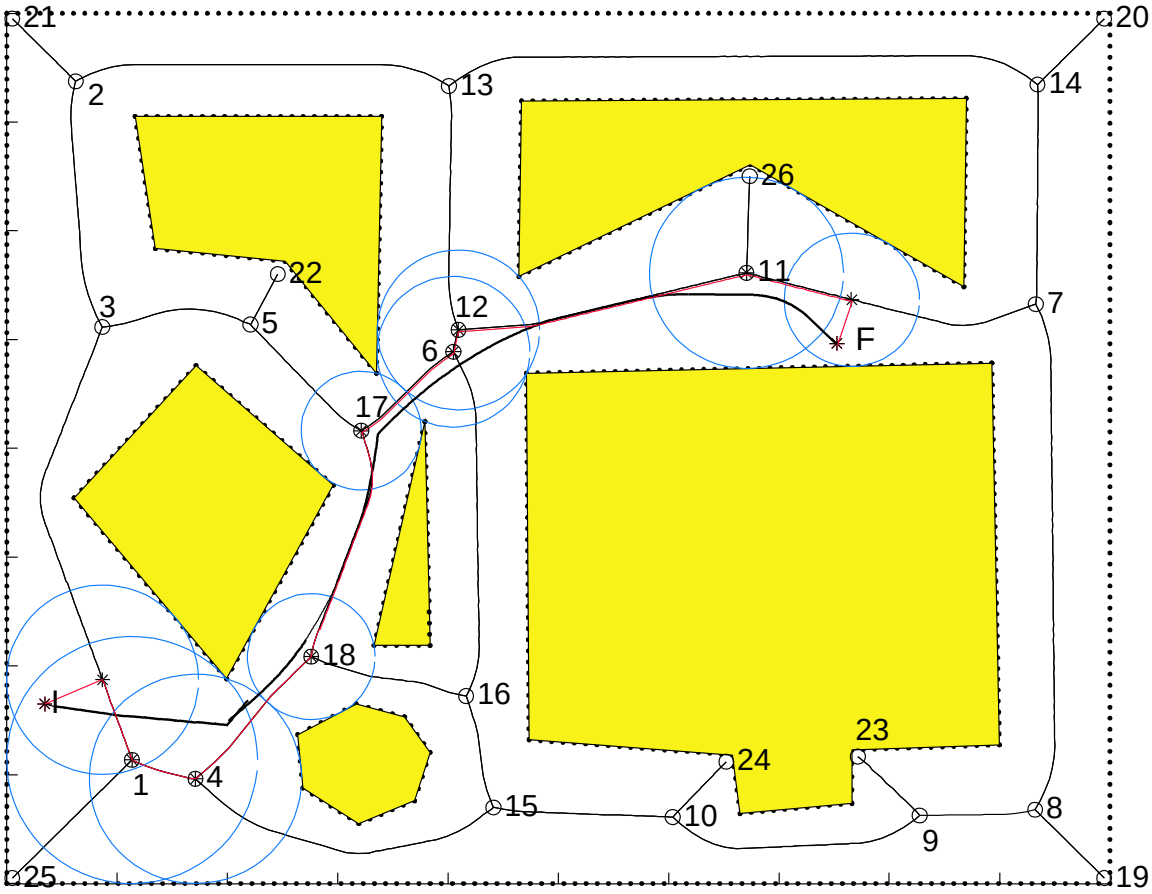


Fig. 23. Shortest path by drawing circles at the nodes

2. Presence of Dynamic Obstacles

The optimization method to minimize the cost function and find the required trajectory is based on the modified Dijkstra's (Table IV) shortest path algorithm. Modified Dijkstra's algorithm searches for the minimum cost path in a weighted directed graph $G = [V, E]$, where all edge weights are nonnegative. Results are provided in the next chapter (Section E).

Table IV. Modified Dijkstra's algorithm

Input: $\tilde{G} = [V, E],\ ri,\ rf$

1. For each $v \in V$ $cost[v] = \infty$, $cost[ri] = 0$
2. Construct set $\mathcal{TL}$ of all $v \in V$
3. While $\mathcal{TL} \neq 0$
   - Find a k in $\mathcal{TL}$ for which $cost[k]$ is minimum, Delete k from $\mathcal{TL}$
   - For each $u \in \mathcal{TL}$ connected to k
     - Calculate $time_{ku},\ cost_{ku}$
     - If $cost[u] > cost[k] + cost_{ku}$
       - $cost[u] = cost[k] + cost_{ku}$
       - $time[u] = time[k] + time_{ku}$
       - $nodebefore[u] = k$
     - If $u = rf$ STOP

CHAPTER VI

EXPERIMENTAL AND SIMULATION RESULTS

A. Experimental Setup

The mobile robot used is the AmigoBot manufactured by ActivMedia Robotics. AmigoBot is a small, two wheel, differential drive mobile robot. It has a maximum linear speed of $750mm/sec$ and a rotational speed of $300°/sec$. It has two drive wheel powered by a 12V DC motor. The drive system includes a passive rear caster wheel for balance. Attached to each drive axle is a high-resolution optical quadrature shaft encoder that provides 9,550 ticks per wheel revolution. It also has eight sonar sensors. The optical encoders and the sonar are deactivated for this experimental setup as the localization and obstacle detection are carried out by the overhead camera. The AmigoBot drive and sensor systems are powered and processed from a single Hitachi H8 microprocessor. The robot communicates with the computer through a $900MHz$ wireless radio modem. High level library functions of the Saphira are used to send simple motion commands to the robot. Saphira is a C/C++ language based software development environment created by SRI International, Inc.

The robot workspace is an area of $3m \times 2m$. White paper is pasted on the ground to reduce the noise in the image. The camera used is the Logitech Quickcam Pro. This is a CCD camera and is one of the many webcam available in the market. The camera provides an image of resolution 640×480 at $15fps$ and 320×240 at $30fps$. The camera is used from a overhead position, $3m$ directly above the center of the workspace. The camera is attached to a computer, equipped with NVidia GeForce2 graphics card. The computer has a single Intel Pentium III processor with speed of 850MHz. The operating system used is Linux.

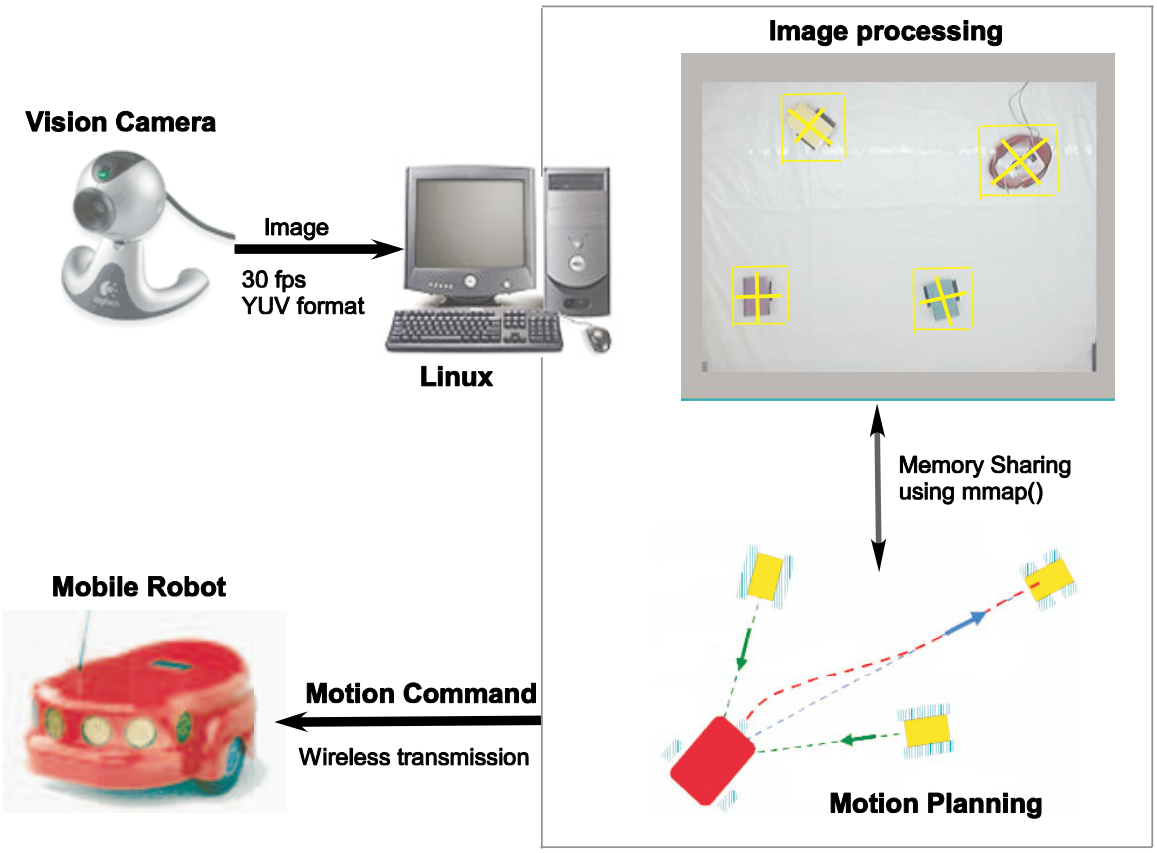


Fig. 24. Overview of the experiment

The path planner and the image processing are two different programs. They share information through memory sharing using memory maps. Intel OpenCV libraries and Intel's Image Processing Library (IPL) are used for image processing. The image processing program provides the positions and orientations of the robot, obstacles and the target. The path planner module uses these information to extract information about the velocities of the objects and to calculate a collision free path of the robot.

The obstacles and the targets are Lego robots. Each individual robot has a flat cardboard of different colors on the top to distinguish them from each other. They

are programmed to move in a random pattern in the experimental space.

The image processing program, writes the data in the memory. The path planner takes the latest data from the memory map.

An overview of the experimental setup is provided in Fig. 24.

B. Visual Sensing

The calculation of a collision free path for a mobile robot is dependent on its ability to correctly and continuously keep track of the poses of the obstacle, target and itself. Additional knowledge of their velocities is helpful in predicting their future motions. Use of on-board sensor system provides unlimited workspace but these suffer from many disadvantages. Advantages of using an overhead camera is noted in [34]. Image processing is done using Intel OpenCV Library functions. The CAMSHIFT algorithm [35] is used. This algorithm was developed for tracking human faces, in an effort towards the creation of a perceptual user interface. This algorithm was adjusted to track multiple objects simultaneously.

1. Setup and Theory

The visual sensor used in this research is a color CCD camera. The camera is placed in the overhead off-board position. The optical axis of the camera is perpendicular to the workspace of the robot. Dissociating the camera and robot and placing it in the overhead position provides valuable information which would otherwise not be possible to achieve. The image provided using overhead camera is not restricted to the field of view. The robot can keep track of its position and those of the obstacle and the targets at all times, regardless of the relative position of these with respect to the robot. From each acquired image frame buffer, the following parameters are

to be identified:

- The position and orientation of the mobile robot.
- The position and orientation of the obstacles.
- The position and orientation of the target.

These positions have to be converted from pixel coordinates to the physical coordinates. The overhead camera position has the advantage of matching pixels directly to the physical coordinates.

Intel OpenCV Library functions are used for the image processing part. The CAMSHIFT [1] algorithm is used. The CAMSHIFT algorithm [35] is a modified mean shift algorithm [36]. The mean shift algorithm is a non-parametric technique that climbs the gradient of a probability distribution to find the nearest dominant mode (peak).

For each video frame, the raw image is converted to a color probability distribution image via a color histogram model of the color being tracked. The Hue Saturation Value (HSV) color system is used. The robot, obstacle and the targets are of different color. The current size and location of the tracked objects are reported and used to set the size and location of the search window in the next video frame. The process is repeated for continuous tracking.

For discrete 2D image probability distributions, the mean location (the centroid) within each search window is found by finding the zeroth and first moment about x and y. $I(x, y)$ is the probability value at (x, y) and the summation is carried over the (x, y) range of each search window.

[1] Library function of Intel OpenCV library

The zero-th moment is defined as

$$M_{00} = \sum_x \sum_y I(x, y) \tag{6.1}$$

The first moment for x and y is defined as

$$M_{10} = \sum_x \sum_y xI(x, y); \quad M_{01} = \sum_x \sum_y yI(x, y) \tag{6.2}$$

The Centroid of the object in each search window is

$$x_c = \frac{M_{10}}{M_{00}}; \quad y_c = \frac{M_{01}}{M_{00}} \tag{6.3}$$

The 2D orientation of the probability distribution is also obtained from the image by using the second moments

$$M_{20} = \sum_x \sum_y x^2 I(x, y); \quad M_{02} = \sum_x \sum_y y^2 I(x, y) \tag{6.4}$$

Then the object orientation (major axis) is

$$\Theta = \frac{\arctan\left(\frac{2\left(\frac{M_{11}}{M_{00}} - x_c y_c\right)}{\left(\frac{M_{20}}{M_{00}} - x_c^2\right) - \left(\frac{M_{02}}{M_{00}} - y_c^2\right)}\right)}{2} \tag{6.5}$$

The length l and width w of the probability distribution (the objects) can be calculated as follows:

$$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}}, \qquad w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}} \tag{6.6}$$

where,

$$a = \frac{M_{20}}{M_{00}} - x_c^2, \quad b = 2\left(\frac{M_{11}}{M_{00}} - x_c y_c\right), \quad \text{and} \quad c = \frac{M_{02}}{M_{00}} - y_c^2$$

2. Image Processing Results

Fig. 25 shows a processed video image that has four search windows corresponding to the robot, target, and the two obstacles. CAMSHIFT calculates the centroid of the 2D color probability within each of the 2D window of calculation, re-centers the window and then calculates the area for the next window size. The centroid are marked with a cross and the search window is displayed with a box. Thus the color probability distribution (Fig. 25) is not calculated over the whole image, it is calculated for the smaller image region surrounding the four windows.

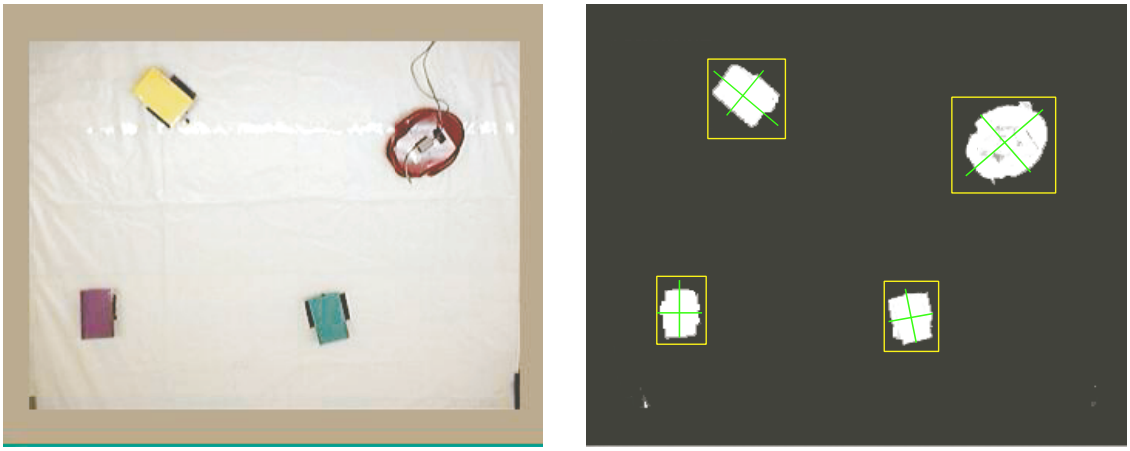


Fig. 25. A video image and its color probability image

C. Planner Scheme

1. Dynamic Obstacles and Dynamic Target

For the case when we have the obstacles and the target as dynamic, it becomes important to update the path at each time interval. Flowchart of the planner is illustrated in Fig. 26.

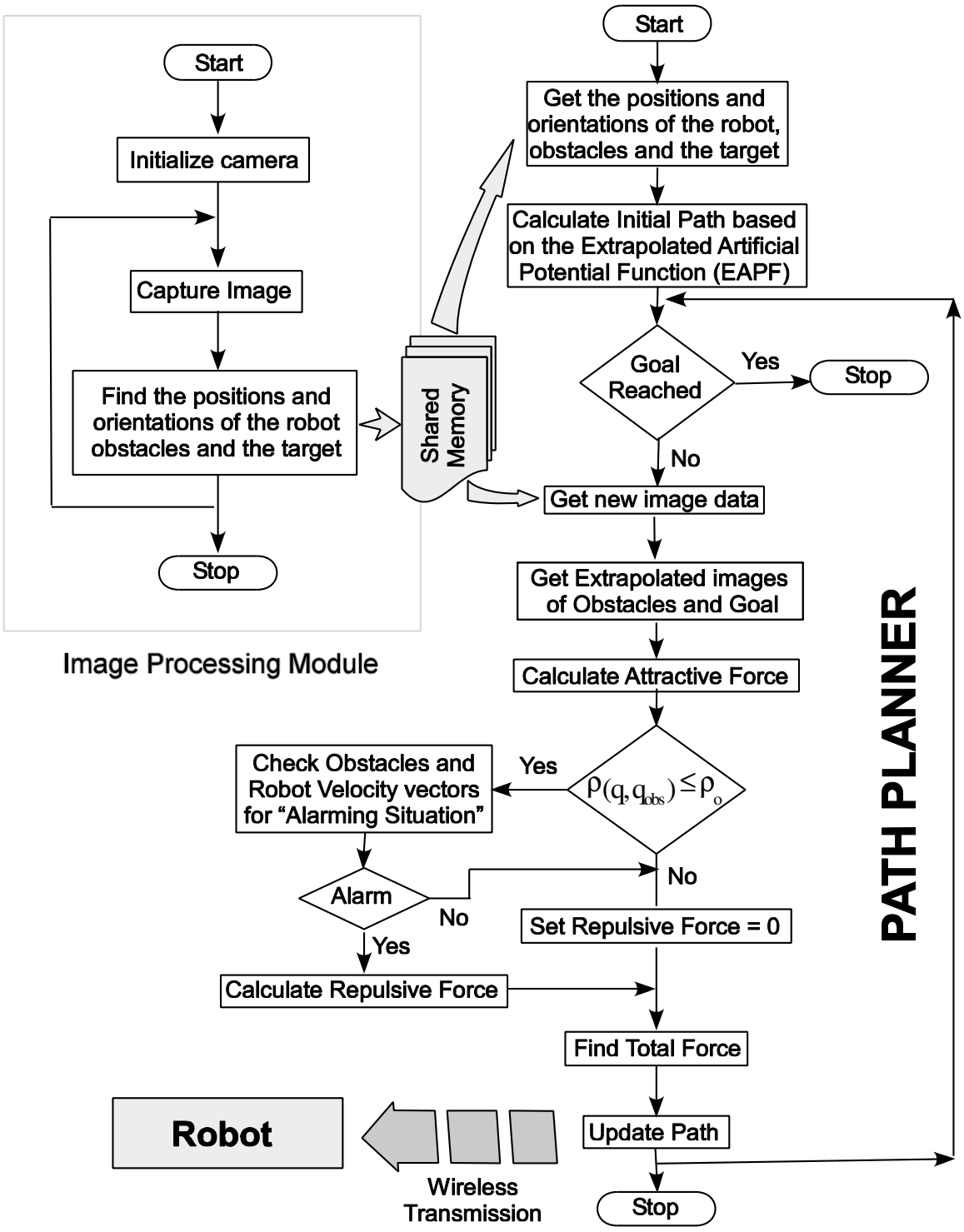


Fig. 26. Flowchart of planner for dynamic obstacles and dynamic target

2. Dynamic Obstacles and Static Target

When we have dynamic obstacles and static target, it may be possible to avoid the obstacles by change in velocity of the robot. In certain scenarios it becomes necessary to change the path too. Probabilistic collision detection is used to find the necessary action that will be required. Fig. 27 gives an outline of the procedure that needs to be followed.

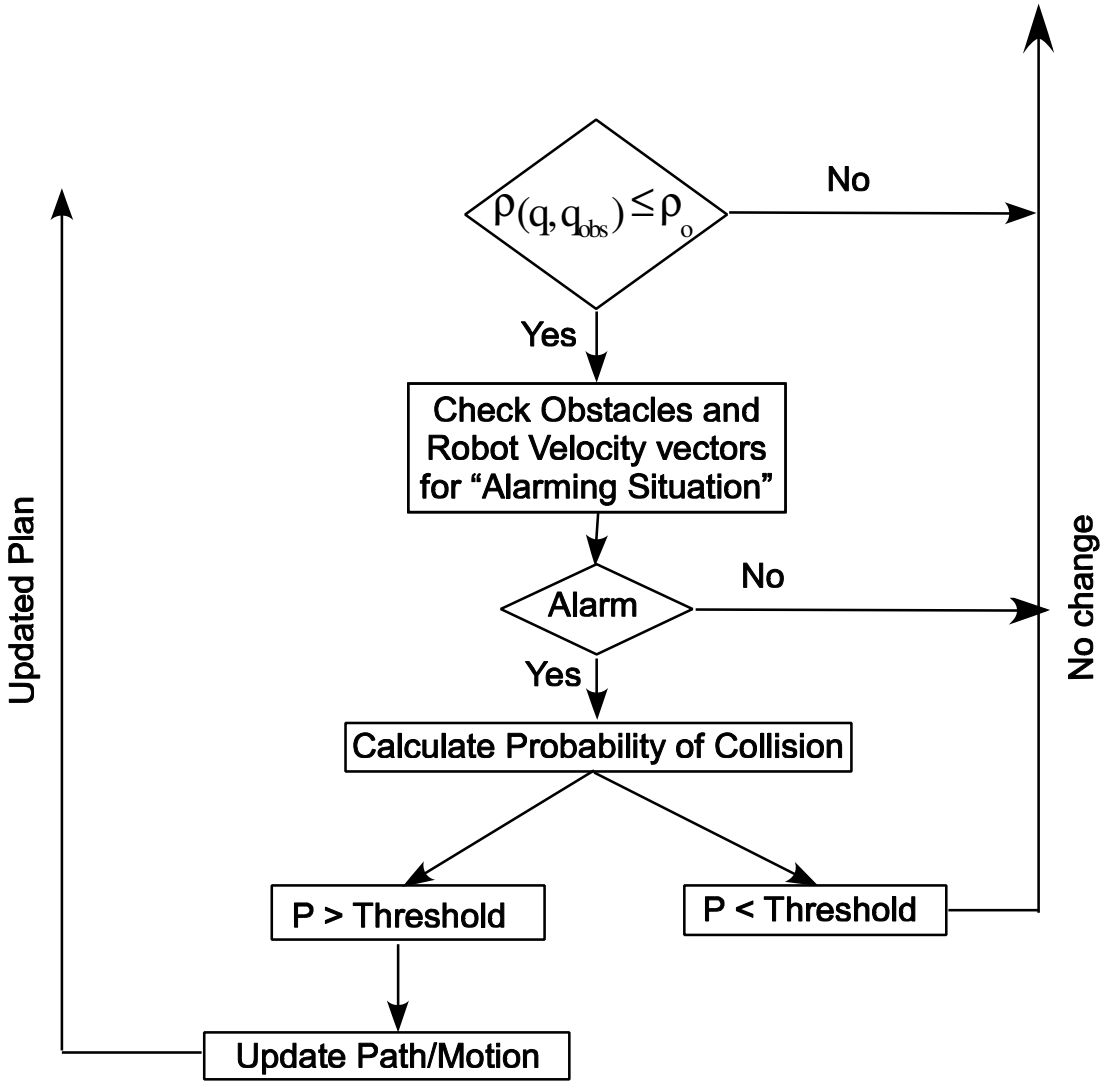


Fig. 27. Flowchart of planner for dynamic obstacles and static target

D. Experimental Results

This experiment (Fig. 28) has two dynamic obstacles and one dynamic target. The farthest robot in the first picture is the target, the two robots in the center are the obstacle. The mobile robot is at the lower right corner of the picture.

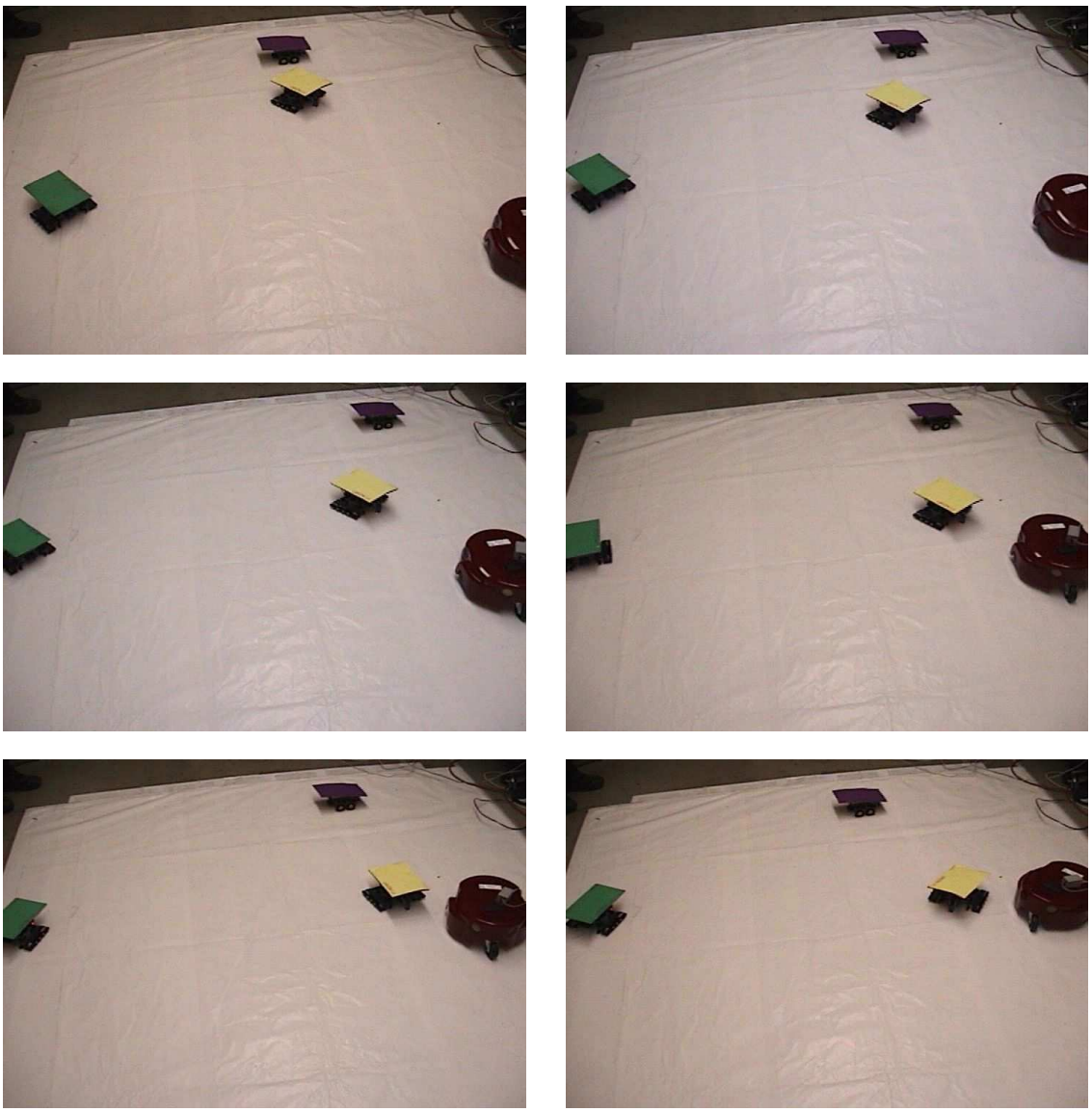


Fig. 28. Snapshots of the experiment (in sequence)

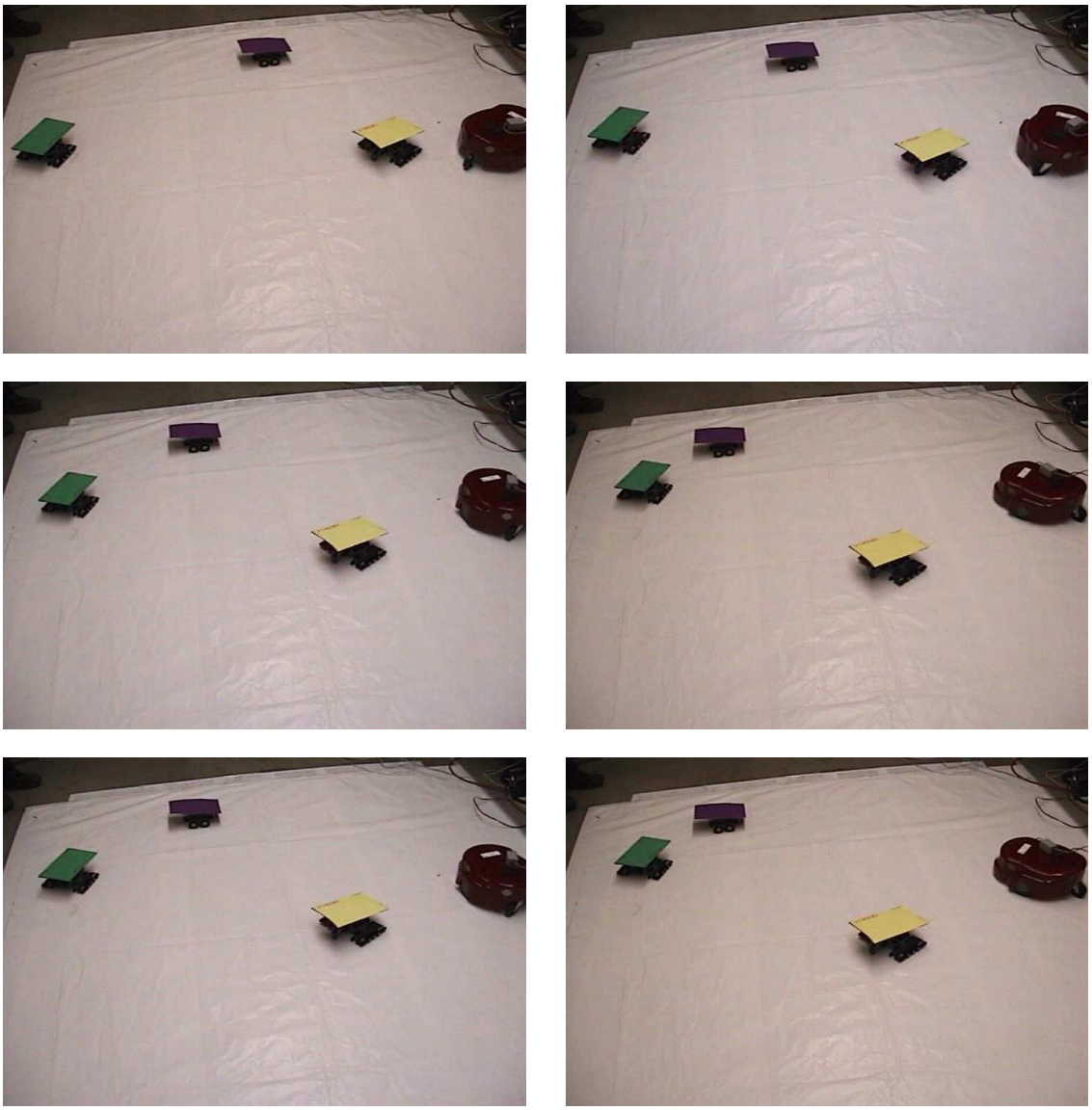


Fig. 28. Continued

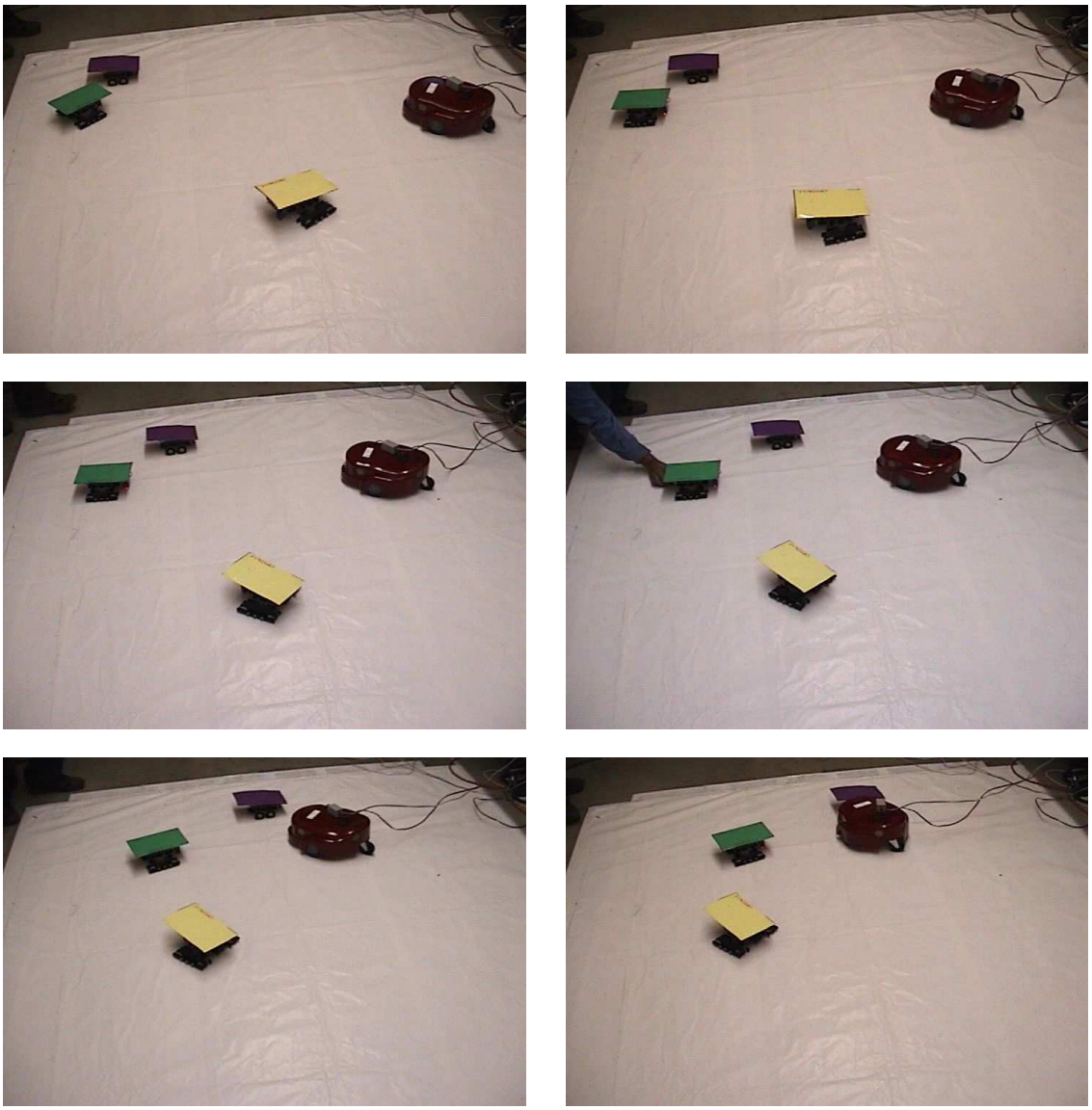


Fig. 28. Continued

E. Simulation Result for Roadmap Based Planner

Simulation was carried out to check the validity of the proposed Roadmap based planner. The environment consists of four static obstacles (Fig. 29). There are four points of interests (POI #n). R_i and R_f are the initial and final position of the robot. *Obs* is the initial position of the dynamic Obstacle.

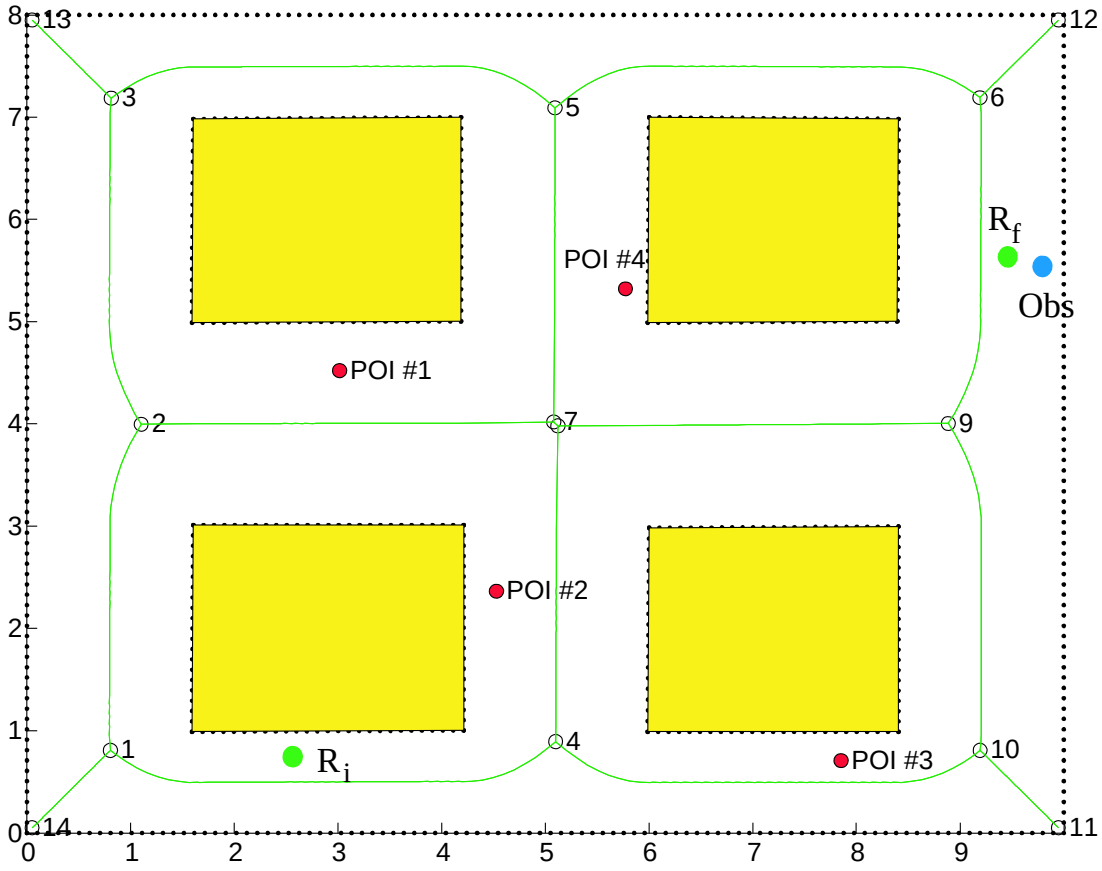


Fig. 29. Environment for the Roadmap based planner

The motion of the robot in the absence of the dynamic obstacle is shown in Fig. 30.

The next two sequences of images (Fig. 31 and Fig. 32) show the motion of the obstacle towards two random POI. The sequence of images shows how the robot

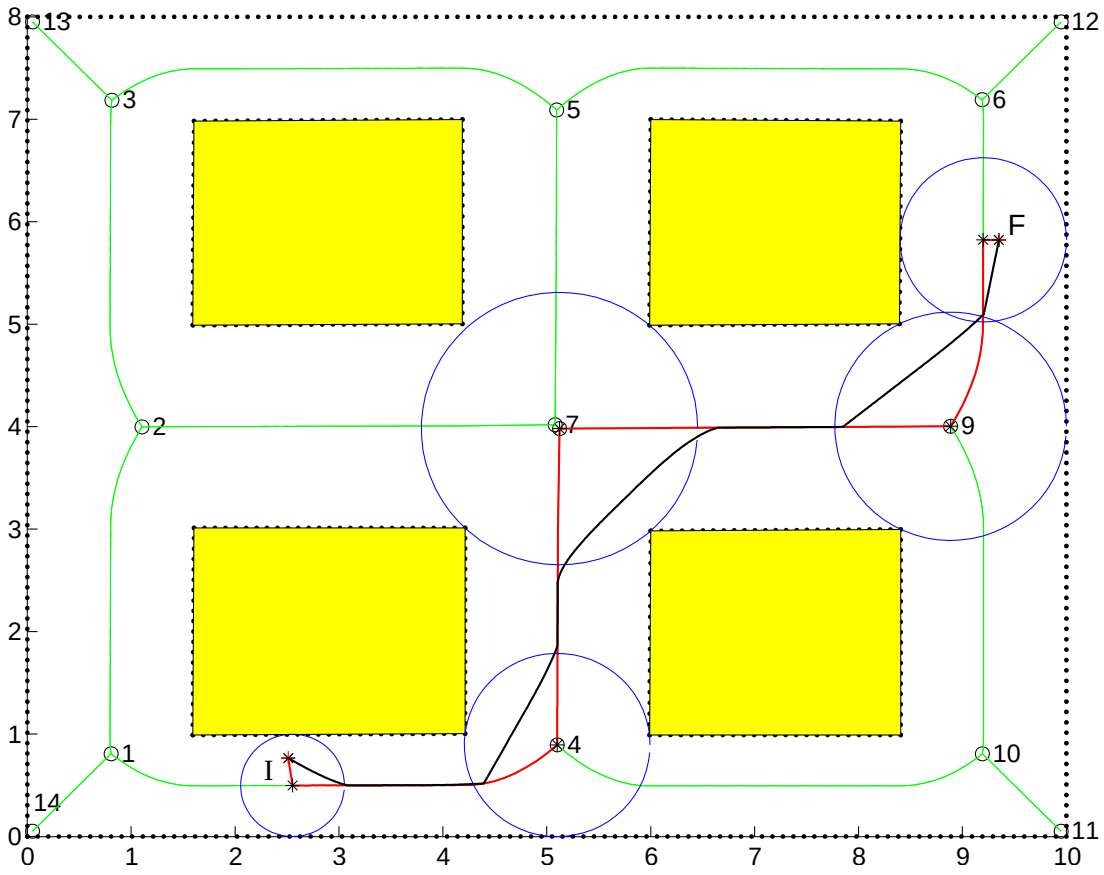


Fig. 30. Robot path in absence of dynamic obstacles

updates its path to avoid the dynamic obstacles. The red lines shows the probable paths along which the obstacle will move. The black line shows the calculated minimum cost path for the robot. The green circle is the robot and the blue circle is the obstacle.

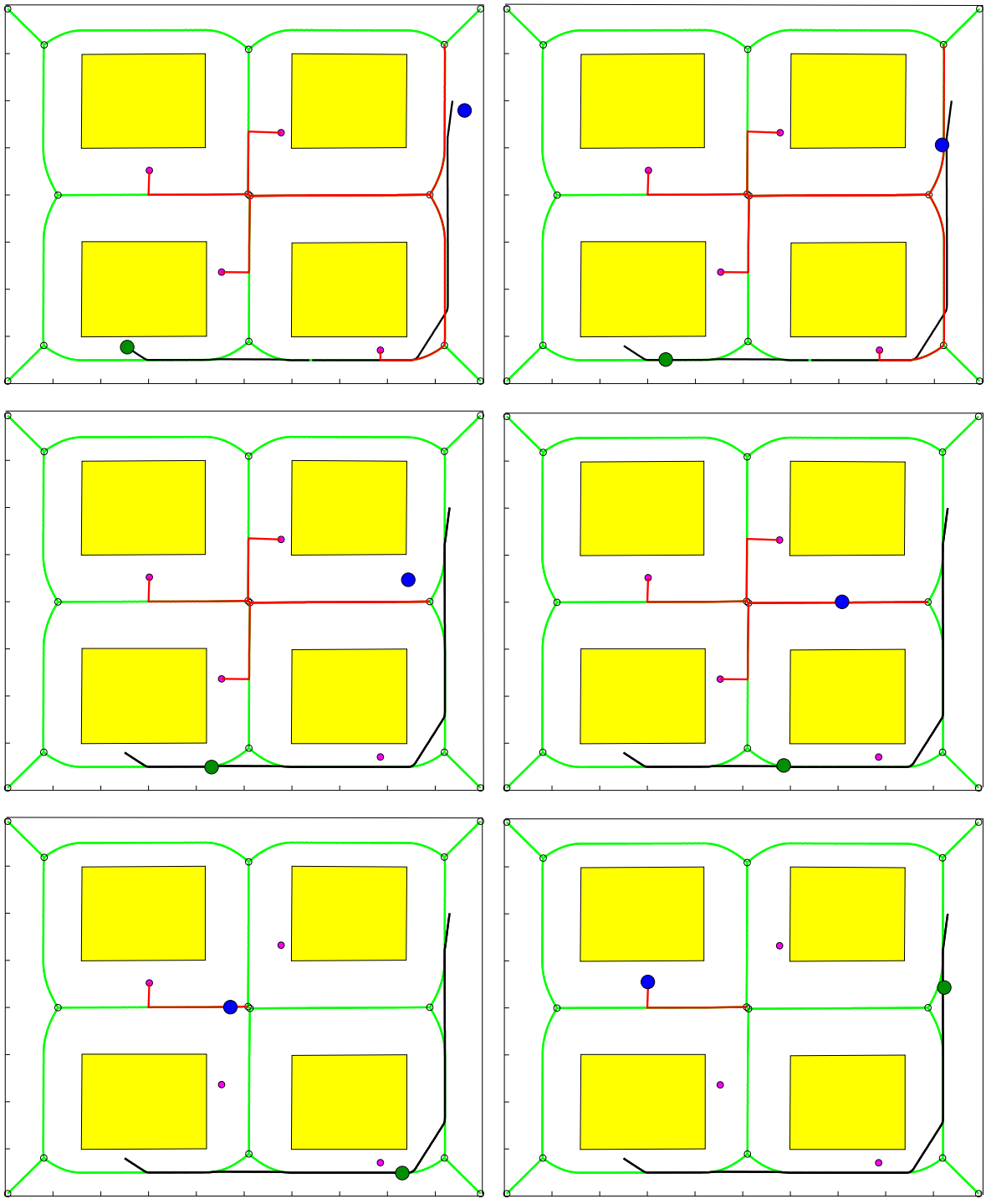


Fig. 31. Simulation result: Case 1 (in sequence)

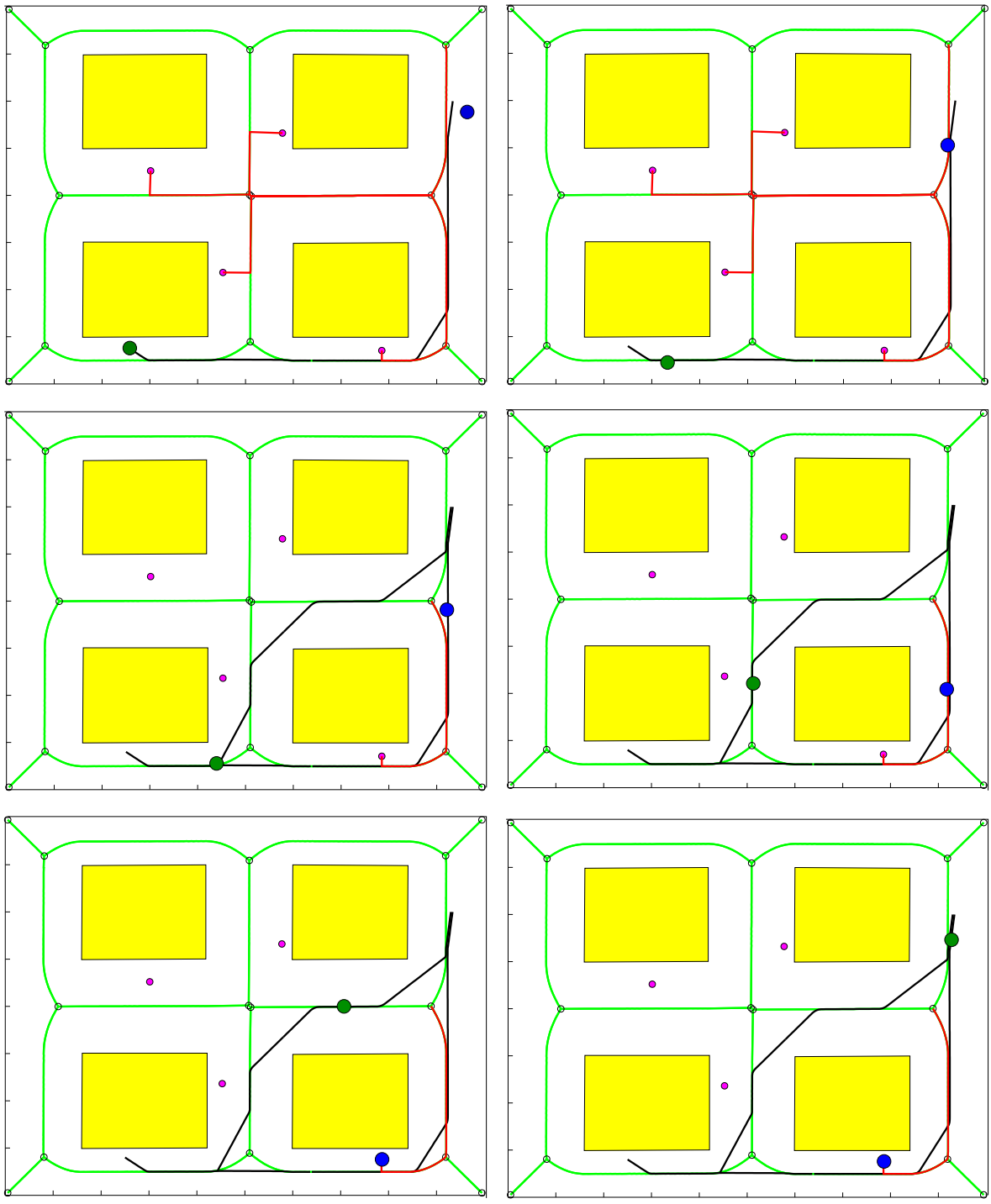


Fig. 32. Simulation result: Case 2 (in sequence)

CHAPTER VII

CONCLUSION

A. Summary

In this thesis, I have developed two trajectory generation plans. The extrapolated artificial potential field method works in a dynamic environment. Through experiments I have shown that this method is able to calculate the trajectory in real-time. The probabilistic collision prediction and avoidance method makes the method mentioned above more robust. The collision alarms coupled with the prediction of collision, provides the robot with means to react to probable collision without the need for trajectory update. The vision system (implemented experimentally) is low cost, and can be used for other experiments in future. The roadmap based planner provides a method for the planning of robot motion in more realistic environments. The Voronoi diagram has been modified to provide a safe, yet shorter path. The trajectory minimizes a cost function which takes into consideration the long-term prediction of obstacle position. The modified Dijkstra's algorithm is used for finding the trajectory which minimizes the cost function.

B. Future Work

The roadmap based planner currently considers only one robot. This method can be extended to form co-operative motion and task planner for multi-robot systems. For better long-term prediction of obstacles, behavioral models can be implemented. The cost function needs to be improved to take into account the curvature of the path and to avoid sharp turns. The generalized Voronoi diagram needs to be updated locally to facilitate the replanning of path around dynamic obstacles.

REFERENCES

[1] H. Bulata and M. Devy, "Incremental construction of a landmark-based and topological model of indoor environments by a mobile robot," in *Proc. IEEE Conference on Robotics and Automation*, Minneapolis, MN, April 1996, pp. 1054-1060.

[2] P. Veelaert and W. Bogaerts, "Ultrasonic potential field sensors for obstacle avoidance," *IEEE Transactions on Robotics and Automation*, vol. 15, pp. 774-779, August 1999.

[3] B. Yamauchi, "Mobile robot localization in dynamic environment using dead reckoning and evidence grids," in *Proc. IEEE Conference on Robotics and Automation*, Minneapolis, MN, April 1996, pp. 1401-1406.

[4] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Transactions on System, Man, and Cybernetics*, vol. 19, pp. 1179-1187, Sept/Oct 1989.

[5] J. Barraquand and J. Latombe, "A Monte-Carlo algorithm for path planning with many degrees of freedom," in *Proc. IEEE Conference on Robotics and Automation*, Cincinnati, Ohio, May 1990, pp. 1712-1717.

[6] E. Rimon and D.E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Transactions on Robotics and Automation*, vol. 8, pp. 501-518, October 1992.

[7] C.W. Warren, "Global path planning using artificial potential fields," in *Proc. IEEE Conference on Robotics and Automation*, Scottsdale, AZ, 1989, pp. 316-321.

[8] J. Latombe, *Robot Motion Planning*, Norwell, MA:Kluwer, 1991.

[9] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, pp. 90-98, 1986.

[10] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proc. IEEE Conference on Robotics and Automation*, Sacramento, CA, April 1991, pp. 1398-1404.

[11] S.S. Ge and Y.J. Cui, "New potential functions for mobile robot path planning," *IEEE Transactions on Robotics and Automation*, vol. 16, pp. 615-621, October 2000.

[12] S. Akshita, T. Hisanobu, S. Kawamura, "Fast path planning available for moving obstacle avoidance by use of Laplace potential," in *Proc. IEEE/RSJ International Conference on Intelligent Robot and Systems*, Yokohama, Japan, July 1993, pp. 673-678.

[13] Y. Wang, G.S. Chirikjian, "A new potential field method for robot path planning," in *Proc. IEEE Conference on Robotics and Automation*, San Francisco, CA, April 2000, pp. 977-982.

[14] J. Miura, H. Uozumi, and Y. Shirai, "Mobile robot motion planning considering the motion uncertainty of moving obstacles," in *Proc. 1999 IEEE International Conference on Systems, Man, and Cybernetics*, Tokyo, Japan, October 1999, pp. IV-692-698.

[15] T. Tsubouchi and S. Arimoto, "Behavior of a mobile robot navigated by an iterated forecast and planning scheme in the presence of multiple moving obstacles,"

in *Proc. IEEE Conference on Robotics and Automation*, San Diego, CA, 1994, pp. 2470-2475.

[16] P. Fiorini and Z. Shiller, "Time optimal trajectory planning in dynamic environments," in *Proc. IEEE Conference on Robotics and Automation*, Minneapolis, MN, April 1996, pp. 1553-1558.

[17] R.C. Smith, P. Cheeseman, "On the representation and estimation of spatial uncertainty," *International Journal of Robotics Research*, vol. 5, no. 4, pp. 56-88, 1986.

[18] H.F. Durrant-Whyte, "Uncertain geometry in robotics," *IEEE Journal of Robotics and Automation*, vol. 4, no. 1, pp. 23-31, 1988.

[19] J. Sanborn and J. Hendler, "A model of reaction for planning in dynamic environments," *International Journal of Artificial Intelligence in Engineering*, vol. 3, no. 2, pp. 95-101, 1988.

[20] J. Reif and M. Sharir, "Motion planning in the presence of moving obstacles," in *25th IEEE Symposium on the Foundation of Computer Science*, Portland, Oregon, October 1985, pp. 144-154.

[21] M. Erdmann and Lozano-Perez, "On multiple moving objects," *Algorithmica*, vol. 2, pp. 477-521, 1987.

[22] K. Fujimura and H. Samet, "A hierarchial strategy for path planning among moving obstacles," *IEEE Transactions on Robotics and Automation*, vol. 5, pp. 61-69, Feb. 1989.

[23] K. Kant and S.W. Zucker, "Towards efficient trajectory planning: the path-velocity decomposition," *International Journal of Robotics Research*, vol. 5,

no. 3, pp. 72-89, 1986.

[24] B.H. Lee and C.S.G. Lee, "Collision free motion planning of two robots," *IEEE Transactions on System, Man, and Cybernetics*, vol. 17, no. 2, pp. 21-32, Jan/Feb 1987.

[25] T. Fraichard, "Dynamic trajectory planning with dynamic constraints: a state-time space approach," *IEEE/RSJ International Conference Intelligent Robots and Systems*, Yokohama, Japan, 1993, pp. 1393-1400.

[26] D.B. Reister and F.G. Pin, "Time-optimal trajectories for mobile robots with two independently driven wheels," *International Journal of Robotics Research*, vol. 13, no. 1, pp. 38-54, 1994.

[27] M. Renaud and J.Y. Fourquet, "Minimum time motion of a mobile robot with two independent, acceleration-driven wheels," in *Proc. IEEE Conference on Robotics and Automation*, Albuquerque (USA), 1997, pp. 2608-2613.

[28] M. Yamamoto, M. Iwamura and A. Mohri, "Time-optimal motion planning of skid-steer mobile robots in the presence of obstacles," in *Proc. of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Victoria, Canada, 1998, pp. 32-37.

[29] P. Pledel and Y. Bestaoui, "Actuator constraints in optimal motion planning of manipulators," in *Proc. IEEE Conference on Robotics and Automation*, Nagoya, Japan, 1995, pp. 2427-2432.

[30] H. Choset and J. Burdick, "Sensor based planning, part ii: Incremental construction of the generalized Voronoi graph," in *Proc. IEEE Conference on Robotics and Automation*, Nagoya, Japan, 1995, pp. 1643-1648.

[31] H. Choset and J. Burdick, "Sensor based planning: The hierarchical generalized Voronoi graph," *The International Journal of Robotic Research*, vol. 19, no. 2, pp. 96-125, February 2000.

[32] K. Hoff, T. Culver, J. Keyser, M. Lin, and D. Manocha, "Interactive motion planning using hardware accelerated computation of generalized Voronoi diagrams," in *Proc. IEEE Conference on Robotics and Automation*, San Francisco, CA, April 2000, pp. 2931-2937.

[33] K.G. Shin, N.D. Mckay, "Minimum-time control of robotics manipulator with geometric path constraints," *IEEE Transactions on Automatic Control*, vol. AC-30 no. 6, pp. 531-541, 1985

[34] E. Kruse, F.M. Wahl, "Camera-based observation of obstacle motions to derive statistical data for mobile robot motion planning," in *Proc. IEEE Conference on Robotics and Automation*, Leuven, Belgium, 1998, pp. 662-667.

[35] G.R. Bradski, "Computer vision face tracking for use in a perceptual user interface," *Intel Technology Journal*, 2nd quarter, 1998, http://developer.intel.com/technology/itj/q21998/pdf/camshift.pdf; accessed July 16, 2003.

[36] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol. 17, pp. 790-799, 1995.

APPENDIX A

ERROR ELLIPSE GENERATION

The determination of whether the probability of observing the object is greater than a given threshold assumes that the probability distribution of our knowledge of the object's location is a multi variate (x, y, θ) Gaussian distribution. The general form is given by:

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n \det \mathbf{C}}} e^{-\frac{1}{2}\left[(\mathbf{x}-\hat{\mathbf{X}})^T C^{-1} (\mathbf{x}-\hat{\mathbf{X}})\right]}, \tag{A.1}$$

where n is the number of dimensions, $\mathbf{C}$ is the covariance matrix, $\hat{\mathbf{X}}$ is the nominal mean vector, and $\mathbf{x}$ is a vector denoting a particular point. The contours of equal probability of this distribution form ellipsoids in n dimensional space that are centered at the mean location $\hat{\mathbf{X}}$, and whose axes are only aligned with the Cartesian frame if the covariance matrix $\mathbf{C}$ is diagonal. The formulas for extracting the principal axes of a two-dimensional ellipsoid are given below. In the case where we are only interested in the positional error ellipse, $\mathbf{C}$ is the reduced (2×2) covariance matrix formed from the (3×3) matrix by extracting only the X, Y terms. In this case the resulting marginal probability distribution is:

$$P(x, y) = \frac{1}{2\pi\sqrt{\sigma_x^2 \sigma_y^2 (1-\rho^2)}} e^{-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_x)^2}{\sigma_x^2} + \frac{2\rho(x-\mu_x)(y-\mu_y)}{\sigma_x \sigma_y} + \frac{(y-\mu_y)^2}{\sigma_y^2}\right]} \tag{A.2}$$

$$C = \begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix} \tag{A.3}$$

where ρ is the correlation coefficient for x and y. μ_x and μ_y are nominal mean values for x and y, respectively.

For decision making purposes, it is necessary to determine the explicit equiprobable

contours (ellipses or ellipsoids) of the multivariate Gaussian distribution specified by given mean $\hat{\mathbf{X}}$ vector and covariance $\mathbf{C_x}$ matrix. These ellipses can be used to determine the probability that a given vector will lie within, say, the 90% confidence ellipse. The ellipsoid formula is:

$$(\mathbf{x} - \hat{\mathbf{X}})^T C^{-1} (\mathbf{x} - \hat{\mathbf{X}}) = k^2 \tag{A.4}$$

where k is a constant chosen for a particular confidence threshold, and $\mathbf{x}$ is a point on the ellipsoid boundary. In case of two-dimensional ellipse, the relationship between k and the probability of a point lying within the ellipsoid specified by k is [17]:

$$P = 1 - e^{\frac{k^2}{2}}, \tag{A.5}$$

and the corresponding family of two-dimensional ellipses is given by Eq. A.4, and reduces to

$$Ax^2 + 2Bxy + Cy^2 - k^2 = 0, \tag{A.6}$$

where A, B and C are found from the two-dimensional covariance matrix and Eq. A.4, and expressed as:

$$A = \frac{1}{(1-\rho^2)\,\sigma_x^2}, \qquad B = -\frac{\rho}{(1-\rho^2)\,\sigma_x \sigma_y}, \qquad C = \frac{1}{(1-\rho^2)\,\sigma_y^2}. \tag{A.7}$$

The angle θ that the major axis of this ellipse makes with the positive x-axis is:

$$\theta = \frac{1}{2} \arctan\left(\frac{2B}{A-C}\right) \qquad \theta = \left[\frac{-\pi}{2}, \frac{\pi}{2}\right]. \tag{A.8}$$

If we define

$$T = \sqrt{A^2 + C^2 - 2AC + 4B^2}, \tag{A.9}$$

then we find the following lengths:

$$\text{half major axis} = \sqrt{\frac{2k^2}{A+C-T}} \tag{A.10}$$

$$\text{half minor axis} = \sqrt{\frac{2k^2}{A+C+T}} \tag{A.11}$$

As given above, the probability of a point being located inside an ellipse defined by a particular value of k is given by:

$$P(x, y \in ellipse) = 1 - e^{-\frac{k^2}{2}} \tag{A.12}$$

$$k^2 = -2\log(1-Pr) \tag{A.13}$$

with the following confidence ellipses for different k:

$$50\% \Rightarrow k^2 = 1.386, \tag{A.14}$$

$$90\% \Rightarrow k^2 = 4.605. \tag{A.15}$$

Conversely, if we define the ellipse parameters first (major axis length, minor axis length, and the angle) based on proper error model, we can now derive probability density function parameters (σ_x, σ_y, and ρ). This is because there exist a unique ellipse for a given probability density function, and vice vera.

Let us define three ellipse parameters a (half major axis length), b (half minor axis length) and θ (the angle that the major axis of ellipse makes with the positive x-axis). If we define $T = \sqrt{A^2 + C^2 - 2AC + 4B^2}$, then these three parameters can be expressed as:

$$a^2 = \frac{2k^2}{A+C-T} \tag{A.16}$$

$$b^2 = \frac{2k^2}{A+C+T} \tag{A.17}$$

$$\theta = \frac{1}{2}\arctan\left(\frac{2B}{A-C}\right) \tag{A.18}$$

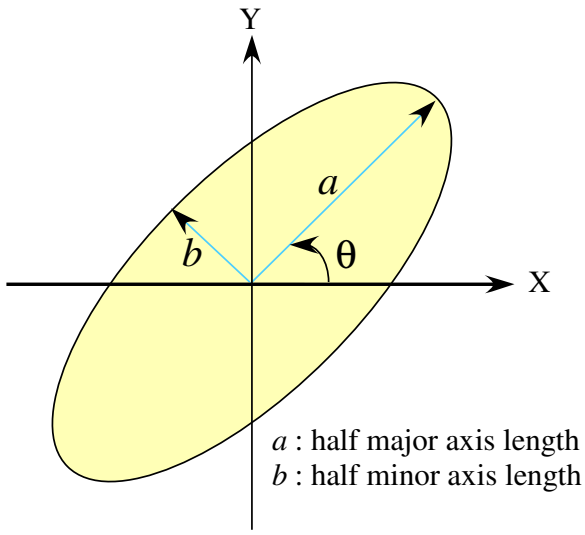


Fig. 33. Ellipse parameters

To derive those parameters from the probability distribution density function, Eq. A.16 to Eq. A.18 can be rearranged as:

$$A + C - T = \frac{2k^2}{a^2} \tag{A.19}$$

$$A + C + T = \frac{2k^2}{b^2} \tag{A.20}$$

$$\frac{2B}{A - C} = \tan 2\theta. \tag{A.21}$$

By extracting Eq. A.19 from Eq. A.20, we can get

$$2T = \frac{2k^2}{b^2} - \frac{2k^2}{a^2} \tag{A.22}$$

$$\Rightarrow T = \frac{k^2}{b^2} - \frac{k^2}{a^2} \qquad (T > 0). \tag{A.23}$$

Adding Eq. A.19 and Eq. A.20 gives us:

$$2\,(A + C) = \frac{2k^2}{b^2} + \frac{2k^2}{a^2} \tag{A.24}$$

$$\Rightarrow A + C = \frac{k^2}{b^2} + \frac{k^2}{a^2} \tag{A.25}$$

From Eq. A.21 and A.25,

$$A = \frac{B}{\tan 2\theta} + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} \tag{A.26}$$

$$C = -\frac{B}{\tan 2\theta} + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} \tag{A.27}$$

$$T^2 = A^2 + C^2 - 2AC + 4B^2 = \left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right)^2 \tag{A.28}$$

$$(A - C)^2 + (2B)^2 = \left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right)^2 \tag{A.29}$$

$$\left(\frac{2B}{\tan 2\theta}\right)^2 + (2B)^2 = \left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right)^2 \tag{A.30}$$

$$(2B)^2 = \frac{\tan^2 2\theta}{(\tan^2 2\theta + 1)} \left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right)^2 \tag{A.31}$$

Finally, B can be expressed with following two cases based on the condition of angle θ.

$$B = \begin{cases} \frac{1}{2}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}} \left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} & (if \quad \theta > 0) \\ -\frac{1}{2}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}} \left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} & (otherwise) \end{cases} \tag{A.32}$$

Then, A, B, and C values from ellipse parameters, a (half major axis length), b (half minor axis length) and θ (angle between major axis and positive x-axis) are rearranged with the angle condition.

Case 1.($\theta \geq 0$)

$$A = -\frac{1}{2\tan 2\theta}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}}\left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} \quad \text{(A.33)}$$

$$B = -\frac{1}{2}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}}\left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) \quad \text{(A.34)}$$

$$C = \frac{1}{2\tan 2\theta}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}}\left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} \quad \text{(A.35)}$$

Case 2.($\theta < 0$)

$$A = \frac{1}{2\tan 2\theta}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}}\left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} \quad \text{(A.36)}$$

$$B = \frac{1}{2}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}}\left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) \quad \text{(A.37)}$$

$$C = -\frac{1}{2\tan 2\theta}\sqrt{\frac{\tan^2 2\theta}{\tan^2 2\theta + 1}}\left(\frac{k^2}{b^2} - \frac{k^2}{a^2}\right) + \frac{k^2}{2b^2} + \frac{k^2}{2a^2} \quad \text{(A.38)}$$

By rewriting Eq. A.7 in terms of $(1-\rho^2)$, we can get:

$$\left(1-\rho^2\right) = \frac{1}{A\sigma_x^2} = \frac{1}{C\sigma_y^2} = \frac{-\rho}{B\sigma_x\sigma_y} \quad \text{(A.39)}$$

$$-A\sigma_x^2\rho = B\sigma_x\sigma_y \qquad -A\sigma_x\rho = B\sigma_y \quad \text{(A.40)}$$

$$-C\sigma_y^2\rho = B\sigma_x\sigma_y \qquad -C\sigma_y\rho = B\sigma_x \quad \text{(A.41)}$$

$$A\sigma_x^2 = C\sigma_y^2 \quad \text{(A.42)}$$

$$A^2\rho^2\sigma_x^2 = B^2\sigma_y^2 = \frac{A}{C}B^2\sigma_x^2 \quad \text{(A.43)}$$

Consequently, ρ, σ_x and σ_y can be expressed in terms of A, B and C,

$$\rho = \begin{cases} \sqrt{\frac{B^2}{AC}} & (if \quad \theta \geq 0) \\ -\sqrt{\frac{B^2}{AC}} & (if \quad \theta < 0) \end{cases} \quad \text{(A.44)}$$

$$\sigma_x^2 = \frac{1}{A\left(1-\rho^2\right)} \quad \text{(A.45)}$$

$$\sigma_y^2 = \frac{1}{C\left(1-\rho^2\right)} \quad \text{(A.46)}$$

APPENDIX B

MINIMUM TIME TRAJECTORY PLANNING

Minimization of travel time along the given trajectory requires that the mobile robot should try to move with the maximum allowable velocity along the path. The optimum velocity pattern is obtained by projecting the velocity $\dot{s}$ along the phase plane s — $\dot{s}$. Yamamoto et al. [28] has dealt with minimum-time control of a mobile robot along a specified trajectory.

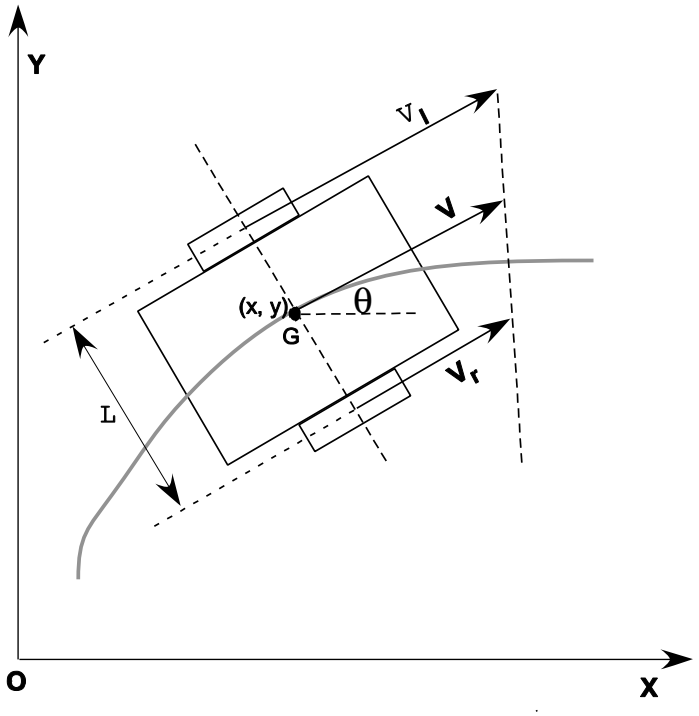


Fig. 34. Kinematic model of a differential drive robot

Fig. 34 shows a kinematic model of a differential drive robot. The state variable is defined as $X = (x, y, \theta, v_r, v_l, v, \dot{\phi})^T$. where (x, y) is the location of the center of

gravity 'G'. It is assumed to be coincident with the center of axle. θ is the orientation of the robot, v is the velocity at 'G', $\dot{\phi}$ is the angular velocity of the robot, L is the distance between the wheels, and $v_r,\ v_l$ are the velocities of the right and left wheels respectively.

The state equation based on the kinematics is given by:

$$\dot{x} = v\cos\theta \tag{B.1}$$

$$\dot{y} = v\sin\theta \tag{B.2}$$

$$\dot{\theta} = \dot{\phi} \tag{B.3}$$

$$\dot{v_r} = a_r \tag{B.4}$$

$$\dot{v_l} = a_l \tag{B.5}$$

$$\dot{v} = \frac{a_r + a_l}{2} \tag{B.6}$$

$$\ddot{\phi} = \frac{a_r - a_l}{L} \tag{B.7}$$

where $a_r,\ a_l$ are the accelerations of the right and left wheels respectively.

The state constraints due to limits in the velocities of the two wheels are:

$$|v_r| \leq v_{max},\ |v_l| \leq v_{max} \tag{B.8}$$

Considering $a_r,\ a_l$ as inputs in the state equations, the input constraints are:

$$|a_r| \leq a_{max},\ |a_l| \leq a_{max} \tag{B.9}$$

The boundary conditions at the initial and final points are:

$$x(0) = (x_0, y_0, \theta_0, 0, 0, 0) \tag{B.10}$$

$$x(t_f) = (x_f, y_f, \theta_f, 0, 0, 0) \tag{B.11}$$

The performance index J is:

$$J = \int_0^{t_f} dt \tag{B.12}$$

Then, the Minimum-time trajectory planning is stated as: Find $X(t)^*$ and $u_r(t)^*,\ u_l(t)^*$ minimizing Eq. B.12, subject to Eq. B.1–B.7 and Eq. B.8–B.11.

Along any given path r_s,

$$\frac{ds}{dt} = v(\equiv \lambda) \tag{B.13}$$

The velocity and acceleration along $r(t)$ is:

$$\dot{r}(t) = \frac{dr(s)}{ds}\lambda \tag{B.14}$$

$$\ddot{r}(t) = \frac{d^2r(s)}{ds^2}\lambda^2 + \frac{dr(s)}{ds}\dot{\lambda} \tag{B.15}$$

Eq. B.1–B.3, B.6, B.7 can be rewritten as:

$$\frac{dx}{ds} = \cos\theta \tag{B.16}$$

$$\frac{dy}{ds} = \sin\theta \tag{B.17}$$

$$\frac{d\theta}{ds} = \frac{d\phi}{ds} \tag{B.18}$$

$$\dot{\lambda} = \frac{a_r + a_l}{2} \tag{B.19}$$

$$\frac{d^2\phi}{ds^2}\lambda^2 + \frac{d\phi}{ds}\dot{\lambda} = \frac{a_r - a_l}{L} \tag{B.20}$$

Solving for $a_r,\ a_l$ using Eq. B.19 and Eq. B.20 yields:

$$a_r = \left(1 + \frac{L}{2}\frac{d\phi}{ds}\right)\dot{\lambda} + \frac{L}{2}\frac{d^2\phi}{ds^2}\lambda^2 \tag{B.21}$$

$$a_l = \left(1 - \frac{L}{2}\frac{d\phi}{ds}\right)\dot{\lambda} - \frac{L}{2}\frac{d^2\phi}{ds^2}\lambda^2 \tag{B.22}$$

The following conditions can be derived,

$$\frac{d\phi}{ds} = \frac{d^2y}{ds^2}\frac{dx}{ds} - \frac{dy}{ds}\frac{d^2x}{ds^2} \tag{B.23}$$

$$\frac{d^2\phi}{ds^2} = \frac{d^3y}{ds^3}\frac{dx}{ds} - \frac{dy}{ds}\frac{d^3x}{ds^3} \tag{B.24}$$

$$\theta = \arctan\left(\frac{\frac{dy}{ds}}{\frac{dx}{ds}}\right) \tag{B.25}$$

Thus, the state equations and the input for the differential drive mobile robot can be expressed in terms of the spatial path $x(s),\ y(s)$ and the velocity λ along the path. The state equations are reduced to:

$$\dot{s} = \lambda \tag{B.26}$$

$$\dot{\lambda} = \frac{a_r + a_l}{2} \equiv a \tag{B.27}$$

The performance index can be rewritten as:

$$J = \int_0^{s_f} \frac{dt}{ds} ds = \int_0^{s_f} \frac{1}{\lambda} ds \tag{B.28}$$

As seen from this equation, λ should be as large as possible to minimize the cost function J. The constraints are rewritten as,

$$\begin{aligned} -v_{max} &\leq M_1\lambda \leq v_{max} \\ -v_{max} &\leq M_2\lambda \leq v_{max} \\ -a_{max} - Q_1\lambda^2 &\leq M_1\dot{\lambda} \leq a_{max} - Q_1\lambda^2 \\ -a_{max} - Q_2\lambda^2 &\leq M_2\dot{\lambda} \leq a_{max} - Q_2\lambda^2 \end{aligned} \tag{B.29}$$

where,

$$M_1 = 1 + \tfrac{L}{2}\tfrac{d\phi}{ds}, \quad M_2 = 1 - \tfrac{L}{2}\tfrac{d\phi}{ds}$$

$$Q_1 = \tfrac{L}{2}\tfrac{d^2\phi}{ds^2}, \quad Q_2 = -\tfrac{L}{2}\tfrac{d^2\phi}{ds^2}$$

These constraints can be rewritten in the form,

$$\begin{aligned} \dot{s}_{min} &\leq \lambda \leq \dot{s}_{max} \\ \ddot{s}_{min} &\leq \dot{\lambda} \leq \ddot{s}_{max} \end{aligned} \tag{B.30}$$

The slope of the trajectories in the phase plane $(s - \dot{s})$ can be written as,

$$\kappa(s, \lambda) = \frac{d\lambda}{ds} = \frac{\frac{d\lambda}{dt}}{\frac{ds}{dt}} = \frac{\dot{\lambda}}{\lambda} \tag{B.31}$$

Bounds on the slope of any trajectory can be expressed in the form

$$\kappa_{min}(s, \lambda) \leq \kappa(s, \lambda) \leq \kappa_{max}(s, \lambda) \tag{B.32}$$

Using the above formulations the optimal velocity profile along a specified path can be found.

VITA

Waqar Ahmad Malik was born in Burnpur, India in September, 1978. He received his baccalaureate degree in mechanical engineering, with honors, from the Indian Institute of Technology, Kharagpur, India in July, 2001. After his master's degree he will continue doctoral studies in mechanical engineering at Texas A&M University.

He can be reached at the Department of Mechanical Engineering, Texas A&M University, College Station, TX 77843-3123.

This document was typeset in LaTeX by Waqar Ahmad Malik.