INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

University Microfilms International
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700  800/521-0600
Adaptive robot path planning with obstacle avoidance in a dynamic environment

Pham, Trung Tat, Ph.D.

Rice University, 1991
RICE UNIVERSITY

ADAPTIVE ROBOT PATH PLANNING
WITH OBSTACLE AVOIDANCE IN A DYNAMIC ENVIRONMENT

by

Trung Tat Pham

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

APPROVED, THESIS COMMITTEE

Dr. Rui J. P. deFigueiredo, Director
Professor of Electrical and Computer
Engineering and of Mathematical Sciences

Dr. J. Boyd Pearson
Professor of Electrical and Computer
Engineering

Dr. John B. Cheatham, Jr.
Professor of Mechanical Engineering
and Materials Science

Houston, Texas
April, 1991
ADAPTIVE ROBOT PATH PLANNING
WITH COLLISION AVOIDANCE IN A DYNAMIC ENVIRONMENT

by

Trung Tat Pham*

Abstract

The problem of path planning for a robotic system is considered, under the conditions in which both the target to be reached and the obstacles with which collision must be avoided are moving in a way not known in advance. To address this situation, an adaptive path planning scheme is proposed: A path is designed for a short time interval $I$ between consecutive data points based on the data gathered at the beginning of $I$. In addition to this adaptive path planning scheme, a key contribution of this research has been the introduction, in the obstacle avoidance problem, of a new artificial potential energy function. This function, induced by an obstacle, depends not only on the robot's position but also on its velocity. Its use permits avoidance of false alarm and trapping caused by obstacles.

Both the cases of a mobile robot system and a manipulator system are considered in detail. For a mobile robot, the system is modeled as a linear time-invariant system. The path planning problem is formulated in the form of two optimization problems: (i) Find the path as if there are no obstacles by minimizing the error between the state of the robot and that of the destination, and (ii) Find the path with obstacle avoidance by minimizing the sum of the deviation from the path computed in (i) and the artificial potential energy function resulting from the obstacles. For a manipulator, the problem is formulated and solved in a similar way. Solutions, analytical examples, and computer simulations are presented for both the mobile robot and manipulator cases.

---

*B.S.E.E. Rice University (1983), M.S.E.E. Rice University (1986)*
ACKNOWLEDGEMENTS

I wish to express my special thanks to my advisor, Dr. Rui J. P. deFigueiredo, for his guidance, support, and encouragement on this work. I also wish to thank Dr. J. Boyd Pearson of the Electrical and Computer Engineering Department, and Dr. John B. Cheatham, Jr. of the Mechanical Engineering and Materials Science Department for being on my research committee.

I also wish to thank Ms. Kathleen Healey, Chief of EF-5 of the Simulation Branch of NASA Johnson Space Center for serving as my NASA technical advisor and sponsor on this project.

Thanks to my former supervisors, Dr. Rafeal E. Pena of Seiscom Delta United, Mr. Jim VanValkenburg of Lockheed Engineering and Sciences, and Mr. Ralph Krog also of Lockheed ESC; Dr. Wolf Kohn of Boeing, and to all my friends at works and as well as at school for the moral support and encouragement during the years I spent at Rice.

A special debt of gratitude to my parents, Mr. and Mrs. Hien Tat Pham for their support, understanding, and love especially during my academic career.

Support of most of this work by the NASA grant NGT 50071, and also the ONR contract N00014-85-K-0152 are gratefully acknowledged.

Trung Pham
# Table of Contents

Cover Page......................................................................................................................... i  
Abstract............................................................................................................................... ii  
Acknowledgment ............................................................................................................... iii  
Table of Contents ............................................................................................................. iv  
List of Figures .................................................................................................................. vii  
Chapter 1. Introduction .................................................................................................... 1  
  1.1. Introduction ............................................................................................................... 1  
  1.2. Previous Work .......................................................................................................... 3  
  1.3. Main Contributions .................................................................................................. 5  
  1.4. System Description ................................................................................................. 5  
  1.5. Problem Formulation ............................................................................................... 6  
    1.5.1. Problem Statement ............................................................................................ 6  
    1.5.2. Assumptions ....................................................................................................... 6  
    1.5.3. Proposed Approach ......................................................................................... 7  
  1.6. Summary of Solution .............................................................................................. 7  
Chapter 2. Motion Planning for a Mobile Robot ............................................................... 9  
  2.1. Introduction ............................................................................................................. 9  
  2.2. Problem Formulation .............................................................................................. 10
<table>
<thead>
<tr>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.1. Notation</td>
</tr>
<tr>
<td>2.2.2. Problem Statement</td>
</tr>
<tr>
<td>2.2.3. Assumptions</td>
</tr>
<tr>
<td>2.2.4. Approach</td>
</tr>
<tr>
<td>2.3. Artificial Potential Energy</td>
</tr>
<tr>
<td>2.4. Computational Procedures</td>
</tr>
<tr>
<td>2.4.1. If There Are No Obstacles</td>
</tr>
<tr>
<td>2.4.2. If There Are Obstacles In The Path</td>
</tr>
<tr>
<td>2.4.3. Summary of Solution</td>
</tr>
<tr>
<td>2.5. Example</td>
</tr>
<tr>
<td>2.6. Simulation Results</td>
</tr>
<tr>
<td>2.7. Convergence</td>
</tr>
<tr>
<td>2.8. Conclusion</td>
</tr>
</tbody>
</table>

Chapter 3. Motion Planning for a Multi-Link Manipulator | 55 |

<table>
<thead>
<tr>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1. Introduction</td>
</tr>
<tr>
<td>3.2. Problem Formulation</td>
</tr>
<tr>
<td>3.2.1. Notation</td>
</tr>
<tr>
<td>3.2.2. Problem Statement</td>
</tr>
<tr>
<td>3.2.3. Assumptions</td>
</tr>
<tr>
<td>3.2.4. Approach</td>
</tr>
<tr>
<td>3.3. Artificial Potential Energy</td>
</tr>
<tr>
<td>3.4. Computational Procedures</td>
</tr>
<tr>
<td>3.5. Example</td>
</tr>
<tr>
<td>3.6. Simulation Results</td>
</tr>
<tr>
<td>3.7. Convergence</td>
</tr>
<tr>
<td>3.8. Conclusion</td>
</tr>
</tbody>
</table>
List of Figures

Figure 2.1a. Head-on Collision ................................................................. 18
Figure 2.1b. Some Chance of Collision ...................................................... 19
Figure 2.1c. No Collision ............................................................................ 19
Figure 2.2. Artist's Concept of a Free Flying Robot ...................................... 31
Figure 2.3. Simulation 1: Stationary Destination and Stationary Obstacles ...... 39
Figure 2.4. Simulation 1: Relative Positions of Obstacles and Destination wrt
Robot .......................................................................................................... 40
Figure 2.5. Simulation 2: Stationary Destination and Moving Obstacles .......... 41
Figure 2.6. Simulation 2: Relative Positions of Obstacles and Destination wrt
Robot .......................................................................................................... 42
Figure 2.7. Simulation 3: Moving Destination and Stationary Obstacles .......... 43
Figure 2.8. Simulation 3: Relative Positions of Obstacles and Destination wrt
Robot .......................................................................................................... 44
Figure 2.9. Simulation 4: Moving Destination and Moving Obstacles ............ 45
Figure 2.10. Simulation 4: Relative Positions of Obstacles and Destination wrt
Robot .......................................................................................................... 46
Figure 2.11. Simulation 5: General Case, Combination of Moving and Stationary
Objects ....................................................................................................... 47
Figure 2.12. Simulation 5: Relative Positions of Obstacles and Destination wrt Robot ................................................................. 48
Figure 2.13. Simulation 6: Impossible Case, Destination Coincides with Obstacle .... 49
Figure 2.14. Simulation 6: Relative Positions of Obstacles and Destination wrt Robot ................................................................. 50
Figure 2.15. Simulation 7: Impossible Case, Destination is Blocked ...................... 51
Figure 2.16. Simulation 7: Relative Positions of Obstacles and Destination wrt Robot ................................................................. 52
Figure 3.1. Kinematics of a 2-Link, 2-DOF Manipulator................................. 66
Figure 3.2. Kinematics of a 3-Link, 3-DOF Manipulator................................. 69
Figure 3.3. Simulation 1: Stationary Destination and Stationary Obstacles .......... 70
Figure 3.4. Simulation 1: Relative Positions of Obstacles and Destination wrt End-Effector.......................................................... 71
Figure 3.5. Simulation 2: Stationary Destination and Moving Obstacles ............ 72
Figure 3.6. Simulation 2: Relative Positions of Obstacles and Destination wrt End-Effector.......................................................... 73
Figure 3.7. Simulation 3: Moving Destination and Stationary Obstacles ............ 74
Figure 3.8. Simulation 3: Relative Positions of Obstacles and Destination wrt End-Effector.......................................................... 75
Figure 3.9. Simulation 4: Moving Destination and Moving Obstacles ............... 76
Figure 3.10. Simulation 4: Relative Positions of Obstacles and Destination wrt End-Effector.......................................................... 77
Figure 3.11. Simulation 5: General Case, Combination of Moving and Stationary Objects ................................................................. 78
Figure 3.12. Simulation 5: Relative Positions of Obstacles and Destination wrt End-Effector.......................................................... 79
Figure 3.13. Simulation 6: Impossible Case, Destination is Blocked .................. 80

Figure 3.14. Simulation 6: Relative Positions of Obstacles and Destination wrt End-Effector ......................................................... 81

Figure 3.15. Simulation 7: Impossible Case, Destination is Out of Reach .......... 82

Figure 3.16. Simulation 7: Relative Positions of Obstacles and Destination wrt End-Effector ......................................................... 83
to Jennifer Nguyễn
Chapter 1
Introduction

1.1. Introduction.

Most robot systems operate on the basis of a detailed plan of motion being available before the motion is actually executed by the controller. For a manipulator robot system, the motion consists of the rotations of the joints required to put the end-effector in a specific position in order to handle some object. The motion is executed by a set of joint servos, each having a local controller that provides an appropriate electrical current to its corresponding servo generating the desired torque applied to the respective joint. This torque is related to the joint angles, joint angular rates, and accelerations by a mapping known as the dynamic relation or equation of motion. For a mobile robot system, the motion consists of the translation of the whole body to a specific position, normally to put the robot within reach of some object so that its manipulator can pick up and handle that object. Depending on the type of the robot, the tool for its motion can be jets (for a space robot), wheels or legs (for a land robot), or a propellant (for an underwater robot). The mobile robot also has a local controller that permits it to acquire the desired position and velocity upon request.
In this dissertation, the computation of the desired motion for an intelligent robot is presented with emphasis on a dynamic environment. In other words, it is assumed that the position and velocity of the desired goal (target) are changing rapidly and that there are other rapidly moving objects (obstacles) with which the robot must not collide. We assume that the positions and velocities of the target and obstacles in the Cartesian space are provided by external sensors at execution time. Under these conditions, there are two problems on adaptive motion planning that one may consider and on which we focus our attention: (i) the one for a manipulator, and (ii) the one for a mobile robot.

The problem of adaptive motion planning for a manipulator may be stated as follows: Given specifications of a manipulator (its kinematic and dynamic equations), and its initial states (initial joint positions and velocities), compute its states as functions of time that converge to a desired goal (target) while avoiding collision with obstacles. The target and obstacles may be nonstationary and their positions and velocities are assumed to be known only at the time of execution.

The problem of adaptive motion planning for a mobile robot may be similarly stated as follows: Given specifications of a mobile robot (its mass and equations of motion), and its initial states (Cartesian position and velocity), compute its states as functions of time that converge to a desired goal (destination) while avoiding collision with obstacles. The destination and obstacles may be nonstationary and their positions and velocities are assumed to be known only at the time of execution.

In this dissertation, the definition of the motion planning problem and our approach to it are presented in chapter 1; the adaptive motion planning problem for a mobile robot in chapter 2; the motion planning problem for a multi-link manipulator in chapter 3; and concluding remarks in chapter 4. In chapter 2, in addition to a detailed formulation and solution of the mobile robot problem stated above, we provide a number of computer simulation results to confirm our theoretical analysis. A similar set of developments are
presented in chapter 3 for the multi-link manipulator problem. Appendices A, B, C, and D provide additional details on the derivation pertaining to chapters 2 and 3.

1.2. Previous Work.

There have been numerous algorithms for finding a collision-free path of a manipulator in a workspace with stationary obstacles, but most are considered too complicated and slow for implementation in a dynamic environment. For practical purposes, there have been two approaches: (i) joint-space configuration, with obstacles represented by a set of forbidden joint angles (Udupa [68], and Widdoes [71]), and (ii) Cartesian-space configuration, with some joints locked so that there is a unique one-to-one transformation from the joint space to a Cartesian space (Lozano-Perez [46], [48]). With the introduction of task-level programming (Lozano-Perez [47]), path planning was elevated to a higher level. Pioneer works (Brooks [12], Brooks and Lozano-Perez [13], Donald [18], Faverjon [23], Gouzenes [29], Laugier and Germain [44], Lozano-Perez [46], Lozano-Perez and Wesley [49]) were many, but again the solutions have not been very practical for dynamic environment applications. The main idea in task-level path planning is to quantize the work space into a set of discrete points, each small path from one point to another being a subtask. Representation of each subtask in a configuration space (joint space) was done with a transformation from a Cartesian space to the joint space (Brooks and Lozano-Perez [13], Lozano-Perez [46], Canny [15]).

To validate that a path is collision-free, techniques have been developed to detect whether a path may result in a collision or interference with other objects (Canny [15], Ozawa, Kumamoto, and Akashi [52], Boyse [9], Cameron [14], Esterling and Rosendale [21], Kawabe, Okano, and Shimada [38]). The term "collision" was referred to the state when an object touched another object, and the term "interference" to the state when more
than one object occupied the same space at the same time interval. There are three categories of collision/interference: (i) multiple interference detection (computing coordinates and orientation of objects), (ii) swept volume (modeling moving objects as swept volume), and (iii) intersection calculation (solving for intersection of trajectories). Using this framework of collision/interference detection, an on-line obstacle avoidance strategy was implemented by continuously evaluating an environment to detect collision in a given path, and if collision was detected, a modification would be made to that path to avoid unforeseen obstacles (Freund [24], Hogan [33], Khatib and LeMaitre [40], Krogh [41]).

Once a path is planned, the remaining task is to find the velocity to set the position of the robot in the time domain. This sub-problem is often called the velocity planning problem (VPP). Works on the VPP have concentrated on simplifying the solution for practical implementation (Krogh [41], Kant and Zucker [37], Angeles, Rojas, and Lopez-Cajun [2], Sahar and Hollerbach [63], Bobrow [6]). In [41], the VPP was transformed into a path planning problem (PPP) with time as another dimension in addition to x, y, and z dimension of the Cartesian space. In [37], a solution as a set of points in the configuration space was derived. In [2] and [63], the solution that optimized time was given for a specific path.

Another approach to the obstacle avoidance problem is to model the work-space as a set of artificial potential fields generated by the obstacles (Krogh [42], Khatib [39], Sahar and Hollerbach [2]). In this approach, each obstacle was modeled as a convex-shaped object (Singh and Wagh [64], Bajaj and Kim [3], [4]) with the distance between two objects computed using the formulation for the distance function between two sets (Gilbert and Johnson [26], Gilbert, Johnson, and Keerthi [27]). Based on the distance between the robot and the obstacles, an artificial potential energy field is generated pushing the robot away from the obstacle.
1.3. Main Contributions.

There are three contributions in this dissertation: (i) a new artificial potential energy function that is in the generalized form, i.e. consisting of both position and velocity, to avoid false alarm and trapping by obstacles, (ii) an adaptive computational scheme that allows the use of external data to address the problem of a rapidly changing environment, and (iii) proof of convergence for the technique in (ii).

1.4. System Description.

In this section, the equations of motion for a manipulator and a mobile robot are given. The equation of motion for a manipulator is a mapping of the joint angles, joint angular velocities, and accelerations into joint torques. It can be derived using the Lagrange relation [54], and has the form:

\[ \tau(t) = f[ \theta(t), \dot{\theta}(t) ] \ddot{\theta}(t) + g[ \theta(t), \dot{\theta}(t) ] \]  

(1.4.1)

where \( \tau(t) \) is a vector of dimension \( N \) consisting of joint torques, \( \theta(t) \) an \( N \)-by-1 vector consisting of joint angles, \( \dot{\theta}(t) \) an \( N \)-by-1 vector of joint angular velocities, \( \ddot{\theta}(t) \) an \( N \)-by-1 vector of joint angular accelerations, \( f[\cdot,\cdot] \) an \( N \)-by-\( N \) matrix often known as the inertia matrix, \( g[\cdot,\cdot] \) an \( N \)-by-1 vector known as the set of Coriolis torques. Let \( x_R(t) \) be the vector consisting of \( \theta(t) \) and \( \dot{\theta}(t) \), and let \( u(t) = \ddot{\theta}(t) \). Then, the manipulator dynamics are described by:

\[ \tau(t) = f[ x_R(t) ] u(t) + g[ x_R(t) ] \]  

(1.4.2a)

\[ . \quad \dot{x}_R(t) = A x_R(t) + B u(t) \]  

(1.4.2b)
where $A$ is a $2N$-by-$2N$ constant matrix, $B$ a $2N$-by-$N$ constant matrix defined below:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad (1.4.3a)$$

$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (1.4.3b)$$

For a mobile robot, let $r_R(t)$ be the Cartesian coordinates of the robot, $\dot{r}_R(t)$ its velocity, and $u(t)$ its acceleration. Let $x_R(t) = [r_R(t) \quad \dot{r}_R(t)]^T$. Then the dynamics of the mobile robot are defined by:

$$\dot{x}_R(t) = A x_R(t) + B u(t) \quad (1.4.4)$$

where the matrices $A$ and $B$ are the same as in (1.3.3a) and (1.3.3b).

1.5. Problem Formulation.

1.5.1. Problem Statement. Given a robotic system description, sensor data on the external environment and on the robot, destination, and obstacles, compute a path for the robot that leads to the destination while avoiding collision with obstacles. The sensor data consists of positions and velocities, and is provided in real-time.

1.5.2. Assumptions. At the beginning of each time interval $[t_{\text{present}}, t_{\text{present}} + \Delta t]$, the sensors provide the positions and velocities of the destination and obstacles. For the case of a mobile robot, the position and velocity of the robot are also provided, and for the case of a multi-link manipulator, joint angles and joint angular velocities of the manipulator are provided.
1.5.3. Proposed Approach. Let the acceleration of a mobile robot, or the joint angular acceleration for a manipulator be constant during each interval $[t_{\text{present}}, t_{\text{present}}+\Delta t]$.

The problem is divided into two steps: (i) compute the acceleration of the robot for the time interval $[t_{\text{present}}, t_{\text{present}}+\Delta t]$ assuming there are no obstacles, and (ii) compute the additional acceleration that pushes the robot from the path computed in (i) to avoid collision with obstacles.

The computation of the acceleration when there are no obstacles can be formulated as an optimization problem minimizing the sum of the accumulated error between the robot and the destination, and the power of the acceleration signal (for a mobile robot) or the power of the torque signal (for a manipulator). The error is minimized to provide convergence toward the destination. The acceleration/torque signal is minimized to provide minimal effort.

After the acceleration/torque is computed assuming there are no obstacles, the additional acceleration/torque is computed to push the robot away from that path to avoid collision. Here, each obstacle is assumed a source of artificial potential energy that increases monotonically with the chance of collision. The second part of the problem can be formulated into an optimization problem minimizing the accumulated error between the actual path of the robot and the path computed in (i) assuming no obstacles, the power of the acceleration/torque signal that pushes the robot in this actual path, and the sum of the artificial potential energies generated by the velocities and positions of the obstacles and the robot.

1.6. Summary of Solution.

The motion planning problem can be solved in the following steps:
(i) Obtain data from sensors: $x_{O(n)}(t_{\text{present}})$, $x_R(t_{\text{present}})$, and $x_D(t_{\text{present}})$.

(ii) Plan the path assuming there are no obstacles.

(iii) Plan the path that deviates the least from the path in (ii) if there are obstacles.

In (i), for the case of a mobile robot, sensors are assumed to provide the positions and velocities of the obstacles, destination, and robot. For the case of a multi-link manipulator, sensors are assumed to provide positions and velocities of the obstacles and destination, joint angles and joint angular velocities of the manipulator. A set of inverse kinematics is provided to transform the data on the obstacles and destination into joint space representation consisting of joint angles and joint angular velocities.

In (ii) and (iii), the solution is provided for the interval $[t_{\text{present}}, t_{\text{present}}+\Delta t]$ in the form of acceleration, velocity, and position of a mobile robot, or joint torques, joint angular accelerations, velocities, and joint angles of a manipulator.

This computation procedure is repeated every time fresh data arrives.
Chapter 2
Motion Planning for a Mobile Robot

In this chapter, our approach to the adaptive motion planning of a mobile robot is presented. In the mobile robot problem, the system under consideration is a linear time-invariant system. The motion of the robot is given in terms of its acceleration. Based on sensor data on outside environment, the motion is computed so that the robot will try to reach the destination while avoiding the obstacles. Motion is computed at the beginning of every short time interval when new data arrives.

2.1. Introduction.

We assume a mobile robot to be a body mass floating in space. Its motion can be described in a linear time invariant state-variable equation, where the state is a vector consisting of the position and velocity, the input vector the acceleration.

The following information is obtained by the sensors: coordinates and velocities of the robot, destination, and all obstacles in the working environment, assuming that data is provided at the beginning of each time interval of length \( \Delta t \), where \( \Delta t \) is in the order of milliseconds.
In this chapter, the task of planning a path for the robot so that it follows and approaches a certain target (or destination) is studied. The problem can be stated as follows: Given the dynamic representation (in form of the state variable equation) of the robot, data on the outside environment (positions and velocities of the target and obstacles) which is provided at execution time, compute the path for the robot to track the target while avoiding collision with obstacles. The path for the robot can be given either in terms of position and velocity or acceleration. These two representations of the path are equivalent because there is a unique mapping that transforms one to another and vice versa. It has been shown in chapter 1 that, under approximate conditions, the state-variable representation of the robot is that of a linear time-invariant system.

In section 2.2, the problem described above is formulated as an optimization problem. In section 2.3, formulation of a proposed artificial potential energy is presented. In section 2.4, a solution is derived using the principle of dynamic programming to address the situation that data on the outside environment is provided only at execution time. In section 2.5, an analytical example for the robot on a 2-dimensional test platform is given. In section 2.6, numerical simulation for the example in section 2.5 is presented. In section 2.7, convergence issues are discussed.

2.2. Problem Formulation.

2.2.1. Notation. Let \( x_R(t) = [ r_R(t) \quad \dot{r}_R(t) ]^T \) be the state vector for the robot, where \( r_R(t) \) is its position vector and \( \dot{r}_R(t) \) its velocity vector in some reference frame. Similarly, the state vector for the destination is \( x_D(t) = [ r_D(t) \quad \dot{r}_D(t) ]^T \), where \( r_D(t) \) is the destination's position vector, \( \dot{r}_D(t) \) its velocity vector; and \( x_{O(n)}(t) = [ r_{O(n)}(t) \quad \dot{r}_{O(n)}(t) ]^T \) where \( r_{O(n)}(t) \) is the \( n \)th obstacle's position vector, \( \dot{r}_{O(n)}(t) \) its velocity vector. Throughout
this chapter, all coordinates and velocities of the robot, target, and obstacles are given in the same reference frame unless otherwise stated.

The robot's motion is described by the state variable equation:

\[ \dot{x}_R(t) = A \ x_R(t) + B \ u(t) \]  \hfill (2.2.1.1)

where \( x_R(t) \) is the state of the robot, \( u(t) \) the acceleration of the robot, and the matrices \( A \) and \( B \) constant matrices defined as:

\[
A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \]  \hfill (2.2.1.2)

\[
B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \]  \hfill (2.2.1.3)

**2.2.2. Problem Statement.** Given a mobile robot capable of moving in all directions of a three-dimensional space, a destination, and several obstacles; assuming that the destination and obstacles can either be moving or stationary; calculate the motion of the robot so that it will reach the destination while avoiding collision with the obstacles.

**2.2.3. Assumptions.** The destination and obstacles are assumed to travel with constant velocities at any short time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), where \( \Delta t \) is in the order of milliseconds. The motion of the destination in a state-variable form is described by:

\[ \dot{x}_D(t) = A \ x_D(t) \]  \hfill (2.2.3.1)

where the matrix \( A \) is given in (2.2.1.2). Similarly, the motion of the \( n^{\text{th}} \) obstacle is described by:
\[ \dot{x}_{O(n)}(t) = A \cdot x_{O(n)}(t) \]  

### 2.2.4. Approach

The problem stated in section 2.2.2 will be solved in the following manner: At the beginning of each time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), data on the robot, destination, and obstacles is collected. Based on this data, a motion is planned only for this time interval. This computation is repeated until the destination is reached.

The problem of computing the motion in a short time interval can be stated in mathematical terms as follows: Compute the acceleration \(u(t)\) for \(t \in [t_{\text{present}}, t_{\text{present}} + \Delta t]\) given the following: \(x_R(t_{\text{present}}), x_D(t_{\text{present}}),\) and \(x_{O(n)}(t_{\text{present}})\) so that \(x_R(t)\) gets closer to \(x_D(t)\).

The solution \(u(t)\) is computed using the following strategies:

(i) Let \(u(t)\) be constant during the short time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), i.e. \(\Delta t\) is in the order of milliseconds.

(ii) Split \(u(t)\) into two parts: \(v(t)\) and \(w(t)\):

\[ u(t) = v(t) + w(t) \]  

where \(v(t)\) is the input that drives the system toward the destination if there are no obstacles, \(w(t)\) the input that drives the system away from the path created by \(v(t)\) if there are obstacles in that path.

(iii) Let \(v(t)\) and \(w(t)\) be constant during the time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\).

(iv) Compute \(u(t)\) at the beginning of the interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\) for \(t \in [t_{\text{present}}, t_{\text{present}} + \Delta t]\) when \(x_R(t_{\text{present}}), x_D(t_{\text{present}}),\) and
\( x_{O(t)}(t_{\text{present}}) \) are available from external sensors. Repeat this step every time sensor data is available.

The solution \( u^*(t) \) is calculated at the beginning of every short time interval \([t_{\text{present}}; t_{\text{present}}+\Delta t]\) when fresh data on the robot, destination, and obstacles at time \( t=t_{\text{present}} \) arrives. The signal \( u^*(t) \) is computed in two steps:

(i) Compute the optimal path when there are no obstacles. The path, denoted by \( x^*_{R(\text{no})}(t) \), and its corresponding input signal \( v^*(t) \) are computed using standard optimal control theory: Minimizing the linear combination of the accumulated difference between the robot's path \( x_{R(\text{no})}(t) \) and the destination's path \( x_D(t) \), and the power of the input signal \( v(t) \). The destination is assumed having no acceleration, its dynamic equation being described in (2.2.3.1). The robot's path when there are no obstacles, \( x_{R(\text{no})}(t) \), satisfies (2.2.4.2b).

(ii) Compute the optimal deviation from the optimal path computed in (i) when there are obstacles using the solution \( v^*(t) \) and \( x^*_{R(\text{no})}(t) \) computed in (i), and data on obstacles provided by sensors. The sum of the optimal deviation and the optimal path \( x^*_{R(\text{no})}(t) \) when there are no obstacles is the actual path that the robot will follow. Let \( w(t) \) be the input that causes the deviation. Then, the total input \( v^*(t) + w^*(t) \) will be fed into the system to achieve the path \( x^*_R(t) \) that avoids collision with obstacles. The path \( x^*_R(t) \) and its corresponding input signal \( u^*(t) = v^*(t) + w^*(t) \) is computed by solving the standard optimal control problem: Minimizing the linear combination of the accumulated deviation from the solution in (i), and the power of the input signal \( u(t) \), and the total artificial potential energy caused by data from the obstacles.
First, the input \( v^*(t) \) is computed based on initial conditions of the robot and destination, \( x_R(t_{\text{present}}) \) and \( x_D(t_{\text{present}}) \). The corresponding path \( x^*_{R(\text{no})}(t) \), the optimal path if there are no obstacles is also computed. The following motion planning problem is solved:

\[
\min_{v(t)} \int_{t_{\text{present}}}^{\infty} \left[ \left( x_{R(\text{no})}(t) - x_D(t) \right)^T R_1 \left( x_{R(\text{no})}(t) - x_D(t) \right) + v^T(t) R_2 v(t) \right] \, dt
\]

\[(2.2.4.2a)\]

s.t.

\[
\dot{x}_{R(\text{no})}(t) = A \, x_{R(\text{no})}(t) + B \, v(t) \quad (2.2.4.2b)
\]

\[
\dot{x}_D(t) = A \, x_D(t) \quad (2.2.4.2c)
\]

where the first term of (2.2.4.2a), \( [x_{R(\text{no})}(t) - x_D(t)]^T R_1 [x_{R(\text{no})}(t) - x_D(t)] \), forces the convergence of the robot toward the destination if there are no obstacles, and the second term, \( v^T(t) R v(t) \), represents the energy of the corresponding input signal. The constraints in (2.2.4.2b) and (2.2.4.2c) are the motion of the robot and destination respectively. The weight matrices \( R_1 \) and \( R_2 \) are decided by the designer, and have the forms:

\[
R_1 = \text{diag}(a_1, a_2, ..., a_N, a_{N+1}, a_{N+2}, ..., a_{2N}) \quad (2.2.4.3)
\]

\[
R_2 = \text{diag}(b_1, b_2, ..., b_N) \quad (2.2.4.4)
\]

where the constants \( a_i \) and \( b_i \) satisfy:

\[
0 \leq a_i \leq 1; \quad i = 1, 2, ..., 2N \quad (2.2.4.5a)
\]

\[
0 < b_i \leq 1; \quad i = 1, 2, ..., N \quad (2.2.4.5b)
\]
Let $v^*(t)$ and $x_{R_{\text{no}}}(t)$ be the solution to (2.2.4.2). The second part of the input $u(t)$, i.e. $w(t)$, is computed based on the initial condition of the robot, $x_R(t_{\text{present}})$, of the obstacles, $x_{O_{\text{n}}}(t_{\text{present}})$, and the solution $v^*(t)$ and $x_{R_{\text{no}}}(t)$. The path $x_R(t)$ that avoids obstacles and its corresponding input $u(t) = v(t) + w(t)$ is computed by solving the following problem:

$$
\min_{w(t)} \int_{t_{\text{present}}}^{t_{\text{present}} + \Delta t} \{ [x_R(t) - x_{R_{\text{no}}}(t)]^T R_3 [x_R(t) - x_{R_{\text{no}}}(t)] + [w(t) + v^*(t)]^T R_4 [w(t) + v^*(t)] + \sum_{n=1}^{N} E(x_R(t), x_{O_{\text{n}}}(t)) \} \, dt
$$

\hspace{1cm} (2.2.4.6a)

s.t.

$$
\dot{x}_R(t) = A \, x_R(t) + B \, [w(t) + v^*(t)]
$$

\hspace{1cm} (2.2.4.6b)

$$
\dot{x}_{O_{\text{n}}}(t) = A \, x_{O_{\text{n}}}(t)
$$

\hspace{1cm} (2.2.4.6c)

where the first term of (2.2.4.6a), $[x_R(t) - x_{R_{\text{no}}}(t)]^T R_3 [x_R(t) - x_{R_{\text{no}}}(t)]$, is the deviation from the optimal path if there are no obstacles, computed earlier, the second term, $[w(t) + v^*(t)]^T R_4 [w(t) + v^*(t)]$, the energy of the input signal $u(t) = w(t) + v^*(t)$, and the third term, the sum of $E(x_R(t), x_{O_{\text{n}}}(t))$, the artificial potential energy caused by the obstacles' and the robot's path. The artificial potential energy is formulated so that the higher the energy, the greater the chance of collision. The specific formulation of this potential energy is presented in the next section. The constraints in (2.2.4.6b) and (2.2.4.6c) correspond to the motions of the robot in the presence of the obstacles, and the obstacles respectively. The weight matrices $R_3$ and $R_4$ are decided by the designer, and have the forms:
\[ R_3 = \text{diag}(c_1, c_2, ..., c_N, c_{N+1}, c_{N+2}, ..., c_{2N}) \]  
\[ R_4 = \text{diag}(d_1, d_2, ..., d_N) \]

where the constants \( c_i \) and \( d_i \) satisfy:

\[ 0 \leq c_i \leq 1 \]  
\[ 0 < d_i \leq 1 \]

\[ 2.2.4.7 \]
\[ 2.2.4.8 \]

\[ 2.2.4.9a \]
\[ 2.2.4.9b \]

### 2.3. Artificial Potential Energy.

The potential energy is created to have the following properties: (i) The energy increases if the component of relative velocity vector along the direction of the relative position vector is negative, and (ii) the energy increases if the magnitude of the relative position decreases. This means that the larger the energy gets, the greater the chance of collision with the obstacle.

Let \( \mathbf{x}_R(t) \) be the state vector of the robot consisting of its position vector \( \mathbf{r}_R(t) \) and velocity vector \( \dot{\mathbf{r}}_R(t) \),

\[
\mathbf{x}_R(t) = \begin{bmatrix} \mathbf{r}_R(t) \\ \dot{\mathbf{r}}_R(t) \end{bmatrix}
\]

(2.3.1)

and similarly, \( \mathbf{x}_{O(n)}(t) \) the state vector of the \( n \)th obstacle consisting of its position \( \mathbf{r}_{O(n)}(t) \) and its velocity \( \dot{\mathbf{r}}_{O(n)}(t) \):

\[
\mathbf{x}_{O(n)}(t) = \begin{bmatrix} \mathbf{r}_{O(n)}(t) \\ \dot{\mathbf{r}}_{O(n)}(t) \end{bmatrix}
\]

(2.3.2)

The following artificial energy function is proposed:
\[ E( x_R(t) , x_{O(n)}(t) ) = - \frac{[x_R(t) - x_{O(n)}(t)]^T A [x_R(t) - x_{O(n)}(t)]}{[x_R(t) - x_{O(n)}(t)]^T A B B^T A^T [x_R(t) - x_{O(n)}(t)]} \] (2.3.3)

which has the properties mentioned above. Here, the matrices A and B are:

\[ A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \] (2.3.4)

\[ B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \] (2.3.5)

The function E is at its maximum when there is a head-on collision, i.e. the obstacle and the robot are traveling in opposite directions. Furthermore, the smaller the distance between the robot and the obstacle, the greater the energy function becomes.

Let us use the notation:

\[ \xi_n(t) = r_R(t) - r_{O(n)}(t) \] (2.3.6)

i.e. \( \xi_n(t) \) is the relative position of the robot with respect to the \( n^{th} \) obstacle. Similarly,

\[ \dot{\xi}_n(t) = \dot{r}_R(t) - \dot{r}_{O(n)}(t) \] (2.3.7)

or \( \dot{\xi}_n(t) \) is the velocity of the robot in the \( n^{th} \) obstacle reference frame. The potential energy function E can be rewritten as:

\[ E( \xi_n(t) , \dot{\xi}_n(t) ) = -\frac{\xi_n^T(t) \dot{\xi}_n(t)}{\xi_n^T(t) \dot{\xi}_n(t)} \] (2.3.8)
Let $\theta(t)$ be the angle between the two vectors $\xi_n(t)$ and $\dot{\xi}_n(t)$. Then,

\[ E(\xi_n(t), \dot{\xi}_n(t)) = -\frac{\|\dot{\xi}(t)\|}{\|\xi(t)\|} \cos(\theta(t)) \]  

(2.3.9)

Equation (2.3.9) shows the following: (i) if the distance is smaller, the energy tends to increase, (ii) if the velocity gets larger, the energy tends to increase, and (iii) if the direction of the velocity is toward the collision, the energy also increases. The following figures show three different cases: (i) head-on collision, (ii) chance of collision, and (iii) no collision.

Figure 2.1a. Head-on Collision
In Figures 2.1a, 2.1b, and 2.1c, the longer vector represents the position of the robot in the obstacle's reference frame, and the shorter vector, the velocity of the robot, also in the obstacle's reference frame. In Figure 2.1a, the two vectors are in opposite direction, i.e. there is a head-on collision. In Figure 2.1b, the two vectors are at an angle greater than 90 degrees and less than 180 degrees, there is a chance of collision. This chance increases as...
the angle increases. In Figure 2.1c, the two vectors are perpendicular, which means that there is no chance of collision.

2.4. Computational Procedures.

The computational procedures consist of two steps: (i) compute the path and its corresponding input signal if there are no obstacles, and (ii) compute the path that avoids obstacles and deviates the least from the path computed in (i).

2.4.1. If There Are No Obstacles. In this problem, the standard path planning problem in the form of the optimal control problem is formulated minimizing a linear combination of the accumulated error and the power of the input signal. Let \( x_D(t) \) be the projected path of the destination whose motion is modeled dynamically in (2.2.4.2c), \( x_{R(\text{no})}(t) \) the path of the robot if there are no obstacles, whose motion is modeled in (2.2.4.2b). Let \( v(t) \) be the input signal that causes this path \( x_{R(\text{no})}(t) \). Let \( R_1 \) and \( R_2 \) be weight matrices with the form given in (2.2.4.3) and (2.2.4.4). The problem of finding \( v(t) \) has been stated in section 2.2.4 earlier and will be recited here:

\[
\min_{v(t)} \int_{t_{\text{present}}}^{\infty} \left\{ [x_{R(\text{no})}(t) - x_D(t)]^T R_1 [x_{R(\text{no})}(t) - x_D(t)] + v^T(t) R_2 v(t) \right\} dt
\]

(2.4.1.1a)

s.t.

\[
\begin{align*}
\dot{x}_{R(\text{no})}(t) &= A x_{R(\text{no})}(t) + B v(t) \\
\dot{x}_D(t) &= A x_D(t)
\end{align*}
\]

(2.4.1.1b) (2.4.1.1c)
where \( x_{R(\text{no})}(t_{\text{present}}) = x_R(t_{\text{present}}) \), and the initial conditions \( x_R(t_{\text{present}}) \) and \( x_D(t_{\text{present}}) \) are provided by sensors. The solution \( v^*(t) \) to the problem above can be obtained as:

\[
v^*(t) = -R_2^{-1} B^T P [x_{R(\text{no})}(t) - x_D(t)] \tag{2.4.1.2}
\]

where \( P \) is the solution to the following equation:

\[
0 = R_1 - P B R_2^{-1} B^T P + A^T P + P A \tag{2.4.1.3}
\]

The uniqueness and existence of \( v^*(t) \) can be guaranteed because of the definitions of the matrices \( A \) and \( B \) in (2.4.1.1b) given in (2.2.1.2) and (2.2.1.3). The derivation of this solution is given in Appendix A.

The solution of (2.4.1.3) is:

\[
P = \begin{bmatrix}
\text{diag}(\sqrt{a_{N+1}a_1 + 2a_1\sqrt{a_1b_1}}) & \text{diag}(\sqrt{a_1b_1}) \\
\text{diag}(\sqrt{a_1b_1}) & \text{diag}(\sqrt{a_{N+1}b_1 + 2b_1\sqrt{a_1b_1}})
\end{bmatrix} \tag{2.4.1.4}
\]

For a short time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), the signal \( v^*(t) \) is held constant as:

\[
v^*(t) = -R_2^{-1} B^T P [x_R(t_{\text{present}}) - x_D(t_{\text{present}})] \tag{2.4.1.5}
\]

where the corresponding path \( x_{R(\text{no})}'(t) \) is:

\[
x_{R(\text{no})}'(t) = x_R(t_{\text{present}}) + (t-t_{\text{present}}) A x_R(t_{\text{present}}) - (t-t_{\text{present}}) B R_2^{-1} B^T P [x_R(t_{\text{present}}) - x_D(t_{\text{present}})]
\]
\[
\frac{1}{2} (t-t_{\text{present}})^2 A B R_2^{-1} B^T P [x_R(t_{\text{present}}) - x_D(t_{\text{present}})]
\]

(2.4.1.6)

At the end of the time interval \([t_{\text{present}}, t_{\text{present}}+\Delta t]\), the difference between the state of the robot if there are no obstacles, \(x_R^{\ast}(t_{\text{present}}+\Delta t)\), and that of the destination \(x_D(t_{\text{present}}+\Delta t)\),

\[
\epsilon(t) = x_R^{\ast}(t_{\text{present}}) - x_D(t)
\]

(2.4.1.7)

can be computed as:

\[
\epsilon(t_{\text{present}}+\Delta t) = M \epsilon(t_{\text{present}})
\]

(2.4.1.8)

where the matrix \(M\) is:

\[
M = \begin{bmatrix}
\text{diag}(1 - \Delta t^2 \sqrt{\frac{a_i}{b_i}}) & \text{diag}(\Delta t - \Delta t^2 \sqrt{\frac{a_i}{b_i}} + 2 \Delta t \sqrt{\frac{a_i}{b_i}}) \\
\text{diag}(-\Delta t \sqrt{\frac{a_i}{b_i}}) & \text{diag}(1 - \Delta t \sqrt{\frac{a_i}{b_i}} + 2 \Delta t \sqrt{\frac{a_i}{b_i}})
\end{bmatrix}
\]

(2.4.1.9)

and its \(l_2\) norm, the square root of the maximum eigenvalue of \(M^T M\), is approximated for \(\Delta t \ll 1\) by:

\[
\| M \|_2 = \max \left\{ 1 - \frac{\Delta t^2}{2} \sqrt{\frac{a_i}{b_i}} + 2 \sqrt{\frac{a_i}{b_i}} \right\}
\]

(2.4.1.10)
which is less than 1 for all $0 \leq a_i \leq 1$ and $0 < b_i \leq 1$. The norm of $M$ is equal to 1 if and only if $a_i = 0$ for all $i = 1, 2, ..., N$. Therefore,

$$\| \epsilon(t_{\text{present}} + \Delta t) \|_2 < \| \epsilon(t_{\text{present}}) \|_2$$  \hspace{1cm} (2.4.1.11)

2.4.2. If There Are Obstacles In the Path. The problem of minimizing the deviation from the path $x_{R(NO)}^*(t)$, the power of the input signal $u(t)$, and the potential energy function $E$ is solved. Let $x_R(t)$ be the state of the robot if there are obstacles, $u(t) = w(t) + v^*(t)$ the acceleration input to the robot, $x_{O(n)}(t)$ the state of the $n^{\text{th}}$ obstacle. Assume that $x_{R(NO)}^*(t)$ and $v^*(t)$ are known from the previous step, and $x_{O(n)}(t_{\text{present}})$ and $x_R(t_{\text{present}})$ are provided by external sensors, then the signal $w(t)$ can be computed by solving the problem in (2.2.4.6), which is recited below:

$$\min_{w(t)} \int_{t_{\text{present}}}^{t_{\text{present}} + \Delta t} \left\{ [x_R(t) - x_{R(NO)}^*(t)]^T R_3 [x_R(t) - x_{R(NO)}^*(t)] + [w(t) + v^*(t)]^T R_4 [w(t) + v^*(t)] + \sum_{n=1}^{N} E(x_R(t), x_{O(n)}(t)) \right\} dt$$  \hspace{1cm} (2.4.2.1a)

s.t.

$$\dot{x}_R(t) = A x_R(t) + B [w(t) + v^*(t)]$$  \hspace{1cm} (2.4.2.1b)

$$\dot{x}_{O(n)}(t) = A x_{O(n)}(t)$$  \hspace{1cm} (2.4.2.1c)

where the weight matrices $R_3$ and $R_4$ were defined earlier in (2.2.4.7) and (2.2.4.8) respectively. With the initial conditions $x_{O(n)}(t_{\text{present}})$ and $x_R(t_{\text{present}})$ known, and the input
signals \( w(t) \) and \( v^*(t) \) being constants during the time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), equations (2.4.2.1b) and (2.4.2.1c) can be solved directly as:

\[
x_{O(n)}(t) = x_{O(n)}(t_{\text{present}}) + (t - t_{\text{present}}) A x_{O(n)}(t_{\text{present}})
\]

\[
x_{R}(t) = x_{R}(t_{\text{present}}) + (t - t_{\text{present}}) A x_{R}(t_{\text{present}}) + (t - t_{\text{present}}) B [w + v^*] + \frac{1}{2} (t - t_{\text{present}})^2 A B [w + v^*]
\]

(2.4.2.3)

which, when substituted into (2.4.2.1a), lead to an unconstrained optimization problem:

\[
\begin{align*}
\min_{w} \quad & w^T \left[ \frac{\Delta t^2}{20} \Gamma_{31} + \frac{\Delta t^3}{3} \Gamma_{32} + \Delta t R_4 \right] w + w^T \left[ 2 \Delta t R_4 \right] v^* \\
& + v^*^T \left[ \Delta t R_4 \right] v^* \\
& + \sum_{n=1}^{N} \frac{1}{2} \ln \{ \Delta t^4 (v^* + w)^T (v^* + w) + \Delta t^2 \eta_n (v^* + w) + \\
& \Delta t^2 \eta_n^T (v^* + w) + \Delta t^2 \eta_n^T \dot{\eta}_n + 2 \Delta t \eta_n^T \eta_n + \eta_n^T \eta_n \} \\
& - \sum_{n=1}^{N} \frac{1}{2} \ln \{ \eta_n^T \eta_n \}
\end{align*}
\]

(2.4.2.4)

where the derivation of the last two terms is detailed in Appendix C, and:

\[
\eta_n = \begin{bmatrix} I & 0 \end{bmatrix} \{ x_{R}(t_{\text{present}}) - x_{O(n)}(t_{\text{present}}) \}
\]

(2.4.2.5a)

\[
\dot{\eta}_n = \begin{bmatrix} 0 & I \end{bmatrix} \{ x_{R}(t_{\text{present}}) - x_{O(n)}(t_{\text{present}}) \}
\]

(2.4.2.5b)

and:
\[ \Gamma_{31} = \text{diag}(c_1, c_2, \ldots, c_N) \]  
(2.4.2.5c)

\[ \Gamma_{32} = \text{diag}(c_{N+1}, c_{N+2}, \ldots, c_{2N}) \]  
(2.4.2.5d)

and \( R_4 \) was defined in (2.2.4.8). The weight constants \( c_i \) and \( d_i \) satisfy the inequalities in (2.2.4.9a) and (2.2.4.9b). The solution of (2.4.2.4) can be easily obtained via approximation for \( \Delta t \ll 1 \):

\[ w^* = -\left[ \frac{\Delta t^5}{20} \Gamma_{31} + \frac{\Delta t^3}{3} \Gamma_{32} + \Delta t R_4 + \sum_{n=1}^{N} \frac{\Delta t^4 I}{\eta_n \eta_n} \right]^{-1} \]

\[ \left[ (\Delta t R_4 + \sum_{n=1}^{N} \frac{\Delta t^4 I}{4 \eta_n^T \eta_n}) v^* + \sum_{n=1}^{N} \frac{\Delta t^3 \eta_n}{2 \eta_n^T \eta_n} + \Delta t^2 \eta_n \right] \]  
(2.4.2.6)

The difference between the robot state \( x^*_R(t) \) and the destination's state \( x_D(t) \),

\[ \epsilon(t) = x^*_R(t) - x_D(t) \]  
(2.4.2.7)

which, at time \( t = t_{\text{present}} + \Delta t \) is:

\[ \tilde{\epsilon}(t_{\text{present}} + \Delta t) = \tilde{M} \tilde{\epsilon}(t_{\text{present}}) - \mu \]  
(2.4.2.8)

where:

\[ \tilde{M} = \begin{bmatrix} \text{diag}(1 - \frac{\Delta t^2}{2} v_i \alpha_i) & \text{diag}(\Delta t - \frac{\Delta t^2}{2} v_i \beta_i) \\ \text{diag}(\Delta t v_i \alpha_i) & \text{diag}(1 - \Delta t v_i \beta_i) \end{bmatrix} \]  
(2.4.2.9)
\[
\begin{align*}
v_i &= \frac{\Delta t^5 c_i + \Delta t^3 c_{N+i}}{20 c_i + \Delta t^3 c_{N+i} + \Delta t d_i + \sum_{n=1}^{N} \frac{\Delta t^4}{\eta_n^T \eta_n}} \\
\alpha_i &= \sqrt{\frac{a_i}{b_i}} \\
\beta_i &= \sqrt{\frac{a_i}{b_i} + 2 \sqrt{\frac{a_i}{b_i}}} \\
\mu &= \left[ \text{diag}(\Delta t^2 \frac{\Delta t^5 c_i + \Delta t^3 c_{N+i} + \Delta t d_i + \sum_{n=1}^{N} \frac{\Delta t^4}{\eta_n^T \eta_n})^{-1}) \right] \sum_{n=1}^{N} \frac{\Delta t^3 \eta_n + \Delta t^2 \eta_n}{2 \eta_n^T \eta_n} \\
&= \left[ \text{diag}(\Delta t^2 \frac{\Delta t^5 c_i + \Delta t^3 c_{N+i} + \Delta t d_i + \sum_{n=1}^{N} \frac{\Delta t^4}{\eta_n^T \eta_n})^{-1}) \right] \sum_{n=1}^{N} \frac{\Delta t^3 \eta_n + \Delta t^2 \eta_n}{2 \eta_n^T \eta_n}
\end{align*}
\]

with the constants \(a_i, b_i, c_i,\) and \(d_i\) satisfy (2.2.4.5a), (2.2.4.5b), (2.2.4.9a), and (2.2.4.9b).

It can be shown from (2.4.2.9) that the error at time \(t = t_{\text{present}} + \Delta t\) is bounded by:

\[
\| \tilde{e}(t) \|_2 \leq \| \tilde{M} \|_2 \| \tilde{e}(t_{\text{present}}) \|_2 + \| \mu \|_2
\]

where:

\[
\| \tilde{M} \|_2 \leq 1 - \frac{\Delta t}{2} v_i \beta_i
\]
The term \( \| \mu \|_2 \) is in the order of \( \Delta t^3 \). For a small \( \Delta t \ll 1 \), it is clear that:

\[
\| \tilde{e} (t_{\text{present}} + \Delta t) \|_2 < \| \tilde{e} (t_{\text{present}}) \|_2
\]  

(2.4.2.13)

2.4.3. Summary of Solution. At the beginning of each time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), the following is available: \( x_R(t_{\text{present}}) \), \( x_D(t_{\text{present}}) \), and \( x_O(t_{\text{present}}) \).

(i) Compute \( v^*(t) \) for \( t \in [t_{\text{present}}, t_{\text{present}} + \Delta t] \), the input that pushes the robot toward the destination assuming there are no obstacles:

\[
v^*(t) = - \left[ \begin{array}{cc}
\text{diag}(\sqrt{\frac{a_i}{b_i}}) & \text{diag}(\sqrt{\frac{a_i}{b_i}} + 2\sqrt{\frac{a_i}{b_i}}) \end{array} \right] \left\{ x_R(t_{\text{present}}) - x_D(t_{\text{present}}) \right\}
\]

(2.4.3.1)

(ii) Compute the corresponding state \( x_{R(0)}^*(t) \) resulting from the input \( v^*(t) \) during the interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\):

\[
x_{R(0)}^*(t) = x_R(t_{\text{present}}) + (t - t_{\text{present}}) A x_R(t_{\text{present}}) - \\
- (t - t_{\text{present}}) \left[ \begin{array}{cc}
0 & 0 \\
\text{diag}(\sqrt{\frac{a_i}{b_i}}) & \text{diag}(\sqrt{\frac{a_i}{b_i}} + 2\sqrt{\frac{a_i}{b_i}}) \end{array} \right] \left\{ x_R(t_{\text{present}}) - x_D(t_{\text{present}}) \right\}
\]
\[
-\frac{1}{2}(t-t_{\text{present}})^2 \begin{bmatrix}
\text{diag}(\sqrt{\frac{\partial^2}{b_i}}) & \text{diag}(\sqrt{\frac{\partial^2}{b_i} + 2\sqrt{\frac{\partial^2}{b_i}}}) \\
0 & 0
\end{bmatrix} \left\{ x_R(t_{\text{present}}) - x_D(t_{\text{present}}) \right\}
\]

(2.4.3.2)

(iii) Compute \( w^*(t) \) for \( t \in [t_{\text{present}}, t_{\text{present}} + \Delta t] \), the additional input that pushes the robot away from the the state \( x_{R(no)}(t) \) so collision can be avoided:

\[
w^*(t) = -\left[ \frac{\Delta t^3}{20} \Gamma_{31} + \frac{\Delta t^3}{3} \Gamma_{32} + \Delta t \, R_4 + \sum_{n=1}^{N} \Delta t^4 \frac{I}{\eta_{in}^T(t_{\text{present}}) \eta_{in}(t_{\text{present}})} \right]^{-1} \\
\left[ (\Delta t \, R_4 + \sum_{n=1}^{N} \Delta t^3 \frac{I}{4 \eta_{in}^T(t_{\text{present}}) \eta_{in}(t_{\text{present}})}) v^* + \sum_{n=1}^{N} \frac{\Delta t^3 \eta_{in}(t_{\text{present}}) + \Delta t^2 \eta_{in}(t_{\text{present}})}{2 \eta_{in}^T(t_{\text{present}}) \eta_{in}(t_{\text{present}})} \right]
\]

(2.4.3.3)

(iv) Compute the actual state \( x_R^*(t) \) of the robot that resulted from the input \( u^*(t) = v^*(t) + w^*(t) \):

\[
x_R^*(t) = x_R(t_{\text{present}}) + (t-t_{\text{present}}) A \, x_R(t_{\text{present}}) + (t-t_{\text{present}}) B \left[ v^* + w^* \right] \\
+ \frac{1}{2} (t-t_{\text{present}})^2 B \left[ v^* + w^* \right]
\]

(2.4.3.4)
(v) For analysis only, the difference of the state of the robot and that of the destination is computed for $t = t_{\text{present}} + \Delta t$. There are two cases. If there are obstacles, then $u^*(t) = v^*(t) + w^*(t)$, and the difference is:

$$
\varepsilon(t_{\text{present}} + \Delta t) = \tilde{M} \varepsilon(t_{\text{present}}) - \mu
$$

(2.4.3.5)

where $\tilde{M}$ and $\mu$ were given in (2.4.2.9) and (2.4.2.10d). If there are no obstacles, then $u^*(t) = v^*(t)$ and $x^*_R(t) = x^*_R(\text{no})(t)$, the difference is:

$$
\varepsilon(t_{\text{present}} + \Delta t) = M \varepsilon(t_{\text{present}})
$$

(2.4.3.6)

where $M$ is defined in (2.4.1.9).

Steps (i) through (v) are repeated at the beginning of every time interval $[t_{\text{present}}, t_{\text{present}} + \Delta t]$ when new data arrives from external sensors.

2.5. Example.

In this section, an analytical example is given for a mobile robot moving on a flat 2-dimensional platform. Let $x$ and $y$ be the components of the Cartesian coordinate system. Then, the dynamical model for a mobile robot is:

$$
\begin{bmatrix}
\dot{x}_{R1} \\
\dot{x}_{R2} \\
\dot{x}_{R3} \\
\dot{x}_{R4}
\end{bmatrix} =
\begin{bmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
x_{R1} \\
x_{R2} \\
x_{R3} \\
x_{R4}
\end{bmatrix} +
\begin{bmatrix}
0 & 0 \\
0 & 0 \\
1 & 0 \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2
\end{bmatrix}
$$

(2.5.1)
where the states are \( x_{R_1}(t) = x_R(t), x_{R_2}(t) = y_R(t), x_{R_3}(t) = \dot{x}_R(t), \) and \( x_{R_4}(t) = \dot{y}_R(t). \) The inputs \( u_1(t) \) and \( u_2(t) \) are the desired accelerations for reaching destination applied in the \( x- \) and \( y- \) direction.

The robot used in this example is the EVA-Retriever, currently in the development phase at the NASA Johnson Space Center. It is a free-flying robot whose motion is enabled by a set of jets in 6 directions of its local Cartesian space. It is equipped with several sensors to collect data from the outside environment. Its mission is to chase and retrieve a loose tool or an astronaut who cannot fly back to the habitant module in the Space Station Freedom. During the development phase, it is tested on an air-bearing floor, a frictionless surface. Thus, its motion is reduced to being two-dimensional. The robot is shown in figure 2.2 below.
Figure 2.2. Artist's Concept of a Free-Flying Robot

Let the weight matrices $R_1$ and $R_2$ be identity matrices. The motion planning problem is solved as follows. First, a solution for the path if there are no obstacles is calculated by solving the problem:
\[
\min_{v(t)} \int_{t_{\text{present}}}^{\infty} \left[ (x_{R(no)}(t) - x_D(t))^T [x_{R(no)}(t) - x_D(t)] + v^T(t) v(t) \right] dt
\]

\[\text{s.t.}\]
\[
\dot{x}_{R(no)}(t) = A x_{R(no)}(t) + B v(t)
\]
\[
\dot{x}_D(t) = A x_D(t)
\]

which yields the solution:

\[
v^*(t) = - [I \quad v^T] (x_{R(no)}(t) - x_D(t))
\]

(2.5.3)

For a short time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), \(v^*(t)\) is held constant:

\[
\begin{bmatrix}
v_1^*(t) \\
v_2^*(t)
\end{bmatrix} = -\begin{bmatrix}
x_{R}(t_{\text{present}}) \\
y_{R}(t_{\text{present}})
\end{bmatrix} + \begin{bmatrix}
x_{D}(t_{\text{present}}) \\
y_{D}(t_{\text{present}})
\end{bmatrix} - v^T\begin{bmatrix}
\dot{x}_{R}(t_{\text{present}}) \\
\dot{y}_{R}(t_{\text{present}})
\end{bmatrix} + v^T\begin{bmatrix}
\dot{x}_{D}(t_{\text{present}}) \\
\dot{y}_{D}(t_{\text{present}})
\end{bmatrix}
\]

(2.5.4)

The corresponding path is:

\[
\begin{bmatrix}
x_{R(no)}^*(t) \\
y_{R(no)}^*(t)
\end{bmatrix} = \begin{bmatrix}
x_{R}(t_{\text{present}}) \\
y_{R}(t_{\text{present}})
\end{bmatrix} + (t - t_{\text{present}})\begin{bmatrix}
\dot{x}_{R}(t_{\text{present}}) \\
\dot{y}_{R}(t_{\text{present}})
\end{bmatrix} + \frac{1}{2} (t - t_{\text{present}})^2 \begin{bmatrix}
v_1^*(t) \\
v_2^*(t)
\end{bmatrix}
\]

(2.5.5)

and
\[
\begin{bmatrix}
\dot{x}_{R(\text{no})}(t) \\
\dot{y}_{R(\text{no})}(t)
\end{bmatrix}
= \begin{bmatrix}
\dot{x}_R(t_{\text{present}}) \\
\dot{y}_R(t_{\text{present}})
\end{bmatrix}
+ (t - t_{\text{present}}) \begin{bmatrix}
v_1^*(t) \\
v_2^*(t)
\end{bmatrix}
\]  

(2.5.6)

for \(t \in [t_{\text{present}}, t_{\text{present}} + \Delta t]\). Substitute (2.5.4) into (2.5.5) and (2.5.6), the path of the robot with respect to that of the destination at the time \(t = t_{\text{present}} + \Delta t\) is:

\[
\begin{bmatrix}
x_{R(\text{no})}(t_{\text{present}} + \Delta t) - x_D(t_{\text{present}} + \Delta t) \\
y_{R(\text{no})}(t_{\text{present}} + \Delta t) - y_D(t_{\text{present}} + \Delta t) \\
x_{R(\text{no})}(t_{\text{present}} + \Delta t) - x_D(t_{\text{present}} + \Delta t) \\
y_{R(\text{no})}(t_{\text{present}} + \Delta t) - y_D(t_{\text{present}} + \Delta t)
\end{bmatrix}
= \begin{bmatrix}
(1-0.5\Delta t^2) & 0 & (\Delta t-0.5\sqrt{3}\Delta t^2) & 0 \\
0 & (1-0.5\Delta t^2) & 0 & (\Delta t-0.5\sqrt{3}\Delta t^2) \\
(-\Delta t) & 0 & (1-\sqrt{3}\Delta t) & 0 \\
0 & (-\Delta t) & 0 & (1-\sqrt{3}\Delta t)
\end{bmatrix}
\begin{bmatrix}
x_{R_{\text{present}}} - x_{D_{\text{present}}} \\
y_{R_{\text{present}}} - y_{D_{\text{present}}} \\
x_{R_{\text{present}}} - x_{D_{\text{present}}} \\
y_{R_{\text{present}}} - y_{D_{\text{present}}} \\
\end{bmatrix}
\]

(2.5.7)

Let us define the matrix \(M\) as:

\[
M = \begin{bmatrix}
(1-0.5\Delta t^2) & 0 & (\Delta t-0.5\sqrt{3}\Delta t^2) & 0 \\
0 & (1-0.5\Delta t^2) & 0 & (\Delta t-0.5\sqrt{3}\Delta t^2) \\
(-\Delta t) & 0 & (1-\sqrt{3}\Delta t) & 0 \\
0 & (-\Delta t) & 0 & (1-\sqrt{3}\Delta t)
\end{bmatrix}
\]

(2.5.8)

then:
$$\| M \|_2 = 1 - \sqrt{3} \Delta t + 2 \Delta t^2 - 0.5 \sqrt{3} \Delta t^3 + 0.5 \Delta t^4 +$$

$$\sqrt{3\Delta t^2 - 4\sqrt{3}\Delta t^3 + 7.75\Delta t^4 - 3\sqrt{3}\Delta t^5 + 2.75\Delta t^6 - 0.5\sqrt{3}\Delta t^7 + 0.25\Delta t^8}$$

(2.5.9)

The derivation of (2.5.9) is shown in Appendix D. From (2.5.9), it is clear that for $\Delta t \ll 1$,

$$\| M \|_2 \leq 1$$

(2.5.10)

Let the weight matrices $R_3$ and $R_4$ be identity matrices. The remaining motion planning problem can be solved in terms of $w(t)$ which can be obtained by solving the following problem:

$$\min_{w} \left[ \frac{\Delta t^5}{20} + \frac{\Delta t^3}{3} + \frac{\Delta t}{3} \right] w^T w + 2\Delta t w^T v^* + \Delta t v^T v^* +$$

$$\sum_{n=1}^{N} \frac{1}{2} \ln \left( \frac{\Delta t^4}{4} (v^* + w)^T (v^* + w) + \Delta t^3 \tilde{\eta}^T_n (v^* + w) + \Delta t^2 \tilde{\eta}^T_n \tilde{\eta}_n + 2\Delta t \tilde{\eta}^T_n \eta_n + \eta^T_n \eta_n \right) +$$

$$\sum_{n=1}^{N} \frac{1}{2} \ln \left( \eta^T_n \eta_n \right)$$

(2.5.11)

and the solution $w^*(t)$ is:

$$w^*(t) = - \left( 1 + \sum_{n=1}^{N} \frac{\Delta t^3}{4\eta^T_n \eta_n} \right) v^* - \sum_{n=1}^{N} \left( \frac{\Delta t^2}{2\eta^T_n \eta_n} \eta_n + \frac{\Delta t}{2\eta^T_n \eta_n} \right)$$

(2.5.12)

where:
\[ \eta_n = \begin{bmatrix} x_R(t_{\text{present}}) \\ y_R(t_{\text{present}}) \end{bmatrix} - \begin{bmatrix} x_O(n)(t_{\text{present}}) \\ y_O(n)(t_{\text{present}}) \end{bmatrix} \]

(2.5.13)

The input \( u^*(t) \) is the sum of \( v^*(t) \) and \( w^*(t) \) computed above. Applying this input into the system for the time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t] \), at the end of the interval, the difference between the state of the robot and that of the destination is:

\[
\begin{bmatrix}
\varepsilon_1(t_{\text{present}} + \Delta t) \\
\varepsilon_2(t_{\text{present}} + \Delta t) \\
\varepsilon_3(t_{\text{present}} + \Delta t) \\
\varepsilon_4(t_{\text{present}} + \Delta t)
\end{bmatrix} =
\begin{bmatrix}
1 - 0.5\alpha \Delta t^2 & 0 & (1 - \sqrt{3}\alpha)\Delta t & 0 \\
0 & 1 - 0.5\alpha \Delta t^2 & 0 & (1 - \sqrt{3}\alpha)\Delta t \\
-\alpha \Delta t & 0 & 1 - \sqrt{3}\alpha \Delta t & 0 \\
0 & -\alpha \Delta t & 0 & 1 - \sqrt{3}\alpha \Delta t
\end{bmatrix}
\begin{bmatrix}
\varepsilon_1(t_{\text{present}}) \\
\varepsilon_2(t_{\text{present}}) \\
\varepsilon_3(t_{\text{present}}) \\
\varepsilon_4(t_{\text{present}})
\end{bmatrix}
\]

(2.5.14)

\[
\begin{bmatrix}
0.5 \beta_1 \Delta t^2 \\
0.5 \beta_2 \Delta t^2 \\
\beta_1 \Delta t \\
\beta_2 \Delta t
\end{bmatrix}
\]

where:

\[
\alpha = \sum_{n=1}^{N} \frac{\Delta t^3}{4[x_R(t_{\text{present}}) - x_O(n)(t_{\text{present}})]^2 + 4[y_R(t_{\text{present}}) - y_O(n)(t_{\text{present}})]^2}
\]

(2.5.15)
and:

$$
\beta_1 = \sum_{n=1}^{N} \frac{\Delta t^2 [x_R(t_{\text{present}}) - x_O(n)(t_{\text{present}})] + \Delta t [x_R(t_{\text{present}}) - x_O(n)(t_{\text{present}})]}{2[x_R(t_{\text{present}}) - x_O(n)(t_{\text{present}})]^2 + 2[y_R(t_{\text{present}}) - y_O(n)(t_{\text{present}})]^2}
$$

(2.5.16)

$$
\beta_2 = \sum_{n=1}^{N} \frac{\Delta t^2 [y_R(t_{\text{present}}) - y_O(n)(t_{\text{present}})] + \Delta t [y_R(t_{\text{present}}) - y_O(n)(t_{\text{present}})]}{2[x_R(t_{\text{present}}) - x_O(n)(t_{\text{present}})]^2 + 2[y_R(t_{\text{present}}) - y_O(n)(t_{\text{present}})]^2}
$$

(2.5.17)

Let us denote the square matrix on the right hand side of equation (2.5.14) by \( \hat{M} \). Then, the \( l_2 \) norm of \( \hat{M} \) is bounded by:

$$
\| \hat{M} \|_2 \leq 1 - \frac{\sqrt{\lambda}}{2} \alpha \Delta t
$$

(2.5.18)

for \( \Delta t \ll 1 \). Then, the error at time \( t = t_{\text{present}} + \Delta t \) is bounded by:

$$
\| e(t_{\text{present}} + \Delta t) \|_2 \leq [1 - \frac{\sqrt{\lambda}}{2} \alpha \Delta t] \| e(t_{\text{present}}) \|_2 + [\frac{\Delta t^2}{2} + \Delta t] \sqrt{\beta_1^2 + \beta_2^2}
$$

(2.5.19)

Since \( \beta_1 \) and \( \beta_2 \) are in the order of \( \Delta t \) as shown in (2.5.16) and (2.5.17), the last term on the right hand side of (2.5.19) is in the order of \( \Delta t^2 \), which does not have any significant impact on the right hand side of (2.5.19) for \( \Delta t \ll 1 \). Therefore, the magnitude of the error at the time \( t = t_{\text{present}} + \Delta t \) is smaller than that at time \( t = t_{\text{present}} \). This shows the convergence of the state of the robot toward that of the destination.
2.6. Simulation Results.

In this section, we present the effectiveness of the approach developed above by computer simulation. The simulated robot is the EVA-Retriever (figure 2.2), which, as mentioned earlier, is a robot system equipped with propulsion jets that enable its movements in space, currently under development at the NASA Johnson Space Center.

The main objective for building this robot is to send it out to rescue an astronaut during the construction of the Space Station, and to retrieve loose tools floating away from an astronaut while he/she is busy constructing the Space Station.

In this section, the simulations are shown in the seven examples below. Figure 2.3 shows the top view of an air bearing floor; which is a horizontal, flat, and frictionless surface that enables the experiments for the robot in a 2-D environment. The large circle is the robot, the small circles the obstacles, and the square the destination.

Example 1. Consider figure 2.3. In this experiment, the destination and obstacles are stationary. The obstacles are located to form a trap. The robot is shown to avoid getting into the trap while reaching the destination. Figure 2.4 shows the relative positions of the destination and obstacles with respect to the robot body.

Example 2. Consider figure 2.5. In this experiment, the destination is stationary and the obstacles are moving. The obstacles form a wall with one opening. The obstacles are moving toward the right end of the air bearing floor. The robot is shown to avoid collision with the obstacles by moving through the opening and reaching the destination. Figure 2.6 shows the relative positions of the destination and obstacles with respect to the robot body.

Example 3. Consider figure 2.7. In this experiment, the destination is moving and the obstacles are stationary. The robot is shown to navigate itself in this environment, reacting to the location of the destination, changing its direction to reach the destination.
Figure 2.8 shows the relative positions of the destination and obstacles with respect to the robot body.

Example 4. Consider figure 2.9. In this experiment, the destination and obstacles are moving. The robot is shown to avoid all the obstacles which are moving toward it. After that, it reaches the destination. Figure 2.10 shows the relative positions of the destination and obstacles with respect to the robot body.

Example 5. Consider figure 2.11. In this experiment, the destination is stationary, four obstacles are moving and two obstacles are stationary. The robot is shown to avoid collision with the four moving obstacles in the beginning. After that, it reaches the destination by going through the opening created by the two stationary obstacles. Figure 2.12 shows the relative positions of the destination and obstacles with respect to the robot body.

Example 6. Consider figure 2.13. In this experiment, the destination and the obstacle are both stationary and at the same place. Even though this is not physically possible, the simulation is shown here as a case of frustration. The robot moves around the destination in circular motion. Figure 2.14 shows the relative positions of the destination and obstacles with respect to the robot body.

Example 7. Consider figure 2.15. In this experiment, another frustration case is set up. The destination and obstacles are stationary. The obstacles surround the destination. The robot moves around this trap and never reaches the destination. Figure 2.16 shows the relative positions of the destination and obstacles with respect to the robot body.
Figure 2.3. Simulation 1: Stationary Destination and Stationary Obstacles
Figure 2.4. Simulation 1: Relative Positions of Obstacles and Destination wrt Robot
Figure 2.5. Simulation 2: Stationary Destination and Moving Obstacles
Figure 2.6. Simulation 2: Relative Positions of Obstacles and Destination wrt Robot
Figure 2.7. Simulation 3: Moving Destination and Stationary Obstacles
Figure 2.8. Simulation 3: Relative Positions of Obstacles and Destination w.r.t Robot
Figure 2.9. Simulation 4: Moving Destination and Moving Obstacles
Figure 2.10. Simulation 4: Relative Positions of Obstacles and Destination wrt Robot
Figure 2.11. Simulation 5: General Case, Combination of Moving and Stationary Objects
Figure 2.12. Simulation 5: Relative Positions of Obstacles and Destination wrt Robot
Figure 2.13. Simulation 6: Impossible Case
Figure 2.14. Simulation 6: Relative Positions of Obstacles and Destination wrt Robot
Figure 2.15. Simulation 7: Impossible Case
Figure 2.16. Simulation 7: Relative Positions of Obstacles and Destination wrt Robot
2.7. Convergence.

The convergence of the robot toward the destination can be established by showing that the error between the state of the robot and that of the destination at the end of every time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\) is strictly less than the error at the beginning of that interval. Recall the solution in section 2.4. There are two cases: (i) there are no obstacles interfering with the robot, and (ii) there are obstacles interfering with the robot. For the first case, the solution yields the error being bounded by:

\[
\| e(t_{\text{present}} + \Delta t) \|_2 \leq \max \{ 1 - \frac{\Delta t}{2} \sqrt{\frac{a_i}{b_i}} + 2 \sqrt{\frac{a_i}{b_i}} \} \| e(t_{\text{present}}) \|_2
\]

(2.7.1)

for all the weight constants \(0 \leq a_i \leq 1\) and \(0 < b_i \leq 1\). For \(\Delta t \ll 1\), we have:

\[
0 \leq \max \{ 1 - \frac{\Delta t}{2} \sqrt{\frac{a_i}{b_i}} + 2 \sqrt{\frac{a_i}{b_i}} \} < 1
\]

(2.7.2)

which means that the error at the end of the time \(t = t_{\text{present}} + \Delta t\) is smaller than that at the time \(t = t_{\text{present}}\). In this case, convergence is guaranteed.

Consider the case when there are obstacles interfering. The solution in section 2.4 yields the error being bounded by:

\[
\| e(t_{\text{present}} + \Delta t) \|_2 \leq \| \tilde{M} \|_2 \| e(t_{\text{present}}) \|_2 + \| \mu \|_2
\]

(2.7.3)

where \(\| \mu \|_2\) is in the order of \(\Delta t^3\) and:

\[
\| \tilde{M} \|_2 \leq \max \{ 1 - \frac{\Delta t}{2} v_i \beta_i \}
\]

(2.7.4)
and the constants $v_i$ and $\beta_i$ were defined in (2.4.2.10b) and (2.4.2.10d). For a small value of $\Delta t$, i.e. $\Delta t \ll 1$,

$$\| \tilde{e}(t_{\text{present}} + \Delta t) \|_2 \leq \| \tilde{e}(t_{\text{present}}) \|_2$$  \hspace{1cm} (2.7.5)

which means: (i) the state of the robot is closer to that of the destination (for the inequality) at the end of the interval, or the solution converges, or (ii) the state of the robot is unable to get closer than a finite distance from the destination (equality), which represents the case of frustration alluded to before.

2.8. Conclusion.

In this chapter, a real-time path planning scheme for the navigation problem of a mobile robot has been investigated and obtained. To address the incompleteness of the data at the planning stage, an adaptive technique was employed to solve the problem at execution time. The obstacles in the workspace were modeled as sources of artificial potential energies that produce repulsion forces acting on the robot body. These energy functions consist of both position and velocity components. The motion planning problem was set up and solved as follows: At the beginning of each time interval, the duration between two consecutive sensor data, information is gathered and processed. Then an optimization problem with parameters based on this information is set up and solved, this process being repeated until the destination is reached. Simulations have been performed to show the application of the results to the problem of the navigation of a robot in space flying to rescue a floating astronaut while avoiding collision with other astronauts and floating construction materials.
Chapter 3
Motion Planning for
a Multi-Link Manipulator

In this chapter, the motion planning problem for the end-effector of a multi-link manipulator mounted on a fixed platform is considered for the case in which information about the target to be handled by the end-effector and obstacles is not available in advance and can only be obtained at the execution time via external sensors. The following issues will be addressed: (i) formulation of the problem of tracking and reaching the moving destination (target) while avoiding collision with moving obstacles, (ii) its solution in real time, and (iii) simulation results. It is assumed that the moving target is within the reach of the manipulator.

3.1. Introduction.

Consider a general revolute N-link, chained configuration manipulator. Let \( \theta \) be an array of joint angles, \( r \) the coordinates of the end-effector in its base coordinate system, \( r = k(\theta) \) the direct kinematic relation (i.e. the mapping of joint angles into the coordinates of the end-effector in its base coordinate system), \( \tau \) the array of joint torques, \( \tau = f(\theta, \dot{\theta}) \) \( \ddot{\theta} + g(\theta, \dot{\theta}) \) the direct dynamics relation (the mapping of joint angles, joint angle rates, and
joint angular acceleration into joint torques) where $\dot{\theta}$ is an array of joint angle rates (first derivative of joint angle array $\theta$ with respect to time) and $\ddot{\theta}$ an array of joint angular accelerations (second derivative of joint angle array $\theta$ with respect to time).

The desired task is for the manipulator to rotate its joints so that its end-effector can track a moving object (target) while avoiding collision with other objects (obstacles). Most modern manipulators can be put into motion by one of the following methods: (i) joint mode: the operator (or computer) inputs the desired joint angles and joint angle rates and the manipulator moves its joints as requested, (ii) torque mode: the operator (or computer) inputs the desired joint torques and the manipulator joint servos apply those torques to the joints, and (iii) end-effector mode: the operator (or computer) inputs the desired coordinates of the end-effector (in either base or world reference frame) and the manipulator moves its joints appropriately to put its end-effector in the desired position. In this chapter, the torque mode is considered for the motion planning of the manipulator to track a moving target because the other two modes are special cases of the torque mode problem.

The following information is obtained from external sensors: Cartesian coordinates and sizes of the target and other objects, and joint angles and joint angle rates of the manipulator. The target and other objects are assumed to have spherical shapes and their sizes are given in terms of their radius. It is assumed that the sensors will provide data at the beginning of each interval of $[t_{\text{present}}, t_{\text{present}} + \Delta t]$ (where $\Delta t$ is in the order of milliseconds). The issues of complex-shaped objects and inaccuracy of sensor data are not within the scope of this work.

3.2. Problem Formulation.

3.2.1. Notation. Let $\theta_R(t)$ be the joint angle vector, $\dot{\theta}_R(t)$ the joint angular velocity vector, $\ddot{\theta}_R(t)$ the joint angular acceleration vector, $\tau_R(t)$ the joint torque vector,
\( r_{ee}(t) \) the coordinate vector of the end-effector, \( \dot{r}_{ee}(t) \) the Cartesian velocity vector of the end-effector, then the kinematics are described by:

\[
\begin{align*}
    r_{ee}(t) &= k[ \theta_R(t) ] \\
    \dot{r}_{ee}(t) &= \nabla k[ \theta_R(t) ] \dot{\theta}_R(t)
\end{align*}
\]  

(3.2.1.1a) (3.2.1.1b)

Equation (3.2.1.1a) is known as the direct kinematic relation, i.e. the mappings of the joint angles into the end-effector's Cartesian coordinates. Equation (3.2.1.1b) is the application of the direct kinematic relation, sometimes known as the velocity direct kinematic relation.

The dynamics are expressed by:

\[
\tau_R(t) = f[ \theta_R(t), \dot{\theta}_R(t) ] \ddot{\theta}_R(t) + g[ \theta_R(t), \dot{\theta}_R(t) ]
\]  

(3.2.1.2)

where \( f[ \theta_R(t), \dot{\theta}_R(t) ] \) is the inertia matrix, \( g[ \theta_R(t), \dot{\theta}_R(t) ] \) the Coriolis vector. Equation (3.2.1.2) is known as the direct kinematics, i.e. the mappings from joint angles, joint angular velocities, and joint angular accelerations into joint torques.

In this chapter, the state of the manipulator is a vector consisting of the joint angle and joint angular velocity vectors,

\[
x_R(t) = \begin{bmatrix}
    \theta_R(t) \\
    \dot{\theta}_R(t)
\end{bmatrix}
\]  

(3.2.1.3)

Let \( u_R(t) \) be the joint angular acceleration vector, then the motion of a manipulator's joint angles can be expressed as:
\[ \dot{x}_R(t) = A x_R(t) + B u_R(t) \]  

(3.2.1.4)

where:

\[
A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad (3.2.1.5a)
\]

\[
B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (3.2.1.5b)
\]

The computation of the joint torque vector \( \tau_R(t) \) in (3.2.1.2) can be rewritten as:

\[
\tau_R(t) = f[ x_R(t) ] u_R(t) + g[ x_R(t) ] \quad (3.2.1.6)
\]

Let \( \theta_D(t) \), \( \dot{\theta}_D(t) \) be the desired joint angle and joint angular velocity vectors, and \( x_D \) the desired state of the manipulator,

\[
x_D(t) = \begin{bmatrix} \theta_D(t) \\ \dot{\theta}_D(t) \end{bmatrix} \quad (3.2.1.7)
\]

Similarly, let \( \theta_{O(n)}(t) \), \( \dot{\theta}_{O(n)}(t) \) be the joint space joint angle and joint angular velocity representation of the \( n^{th} \) obstacle, its corresponding state is:

\[
x_{O(n)}(t) = \begin{bmatrix} \theta_{O(n)}(t) \\ \dot{\theta}_{O(n)}(t) \end{bmatrix} \quad (3.2.1.8)
\]
3.2.2. **Problem Statement.** Given a manipulator with \( N \) joints and \( N \) degrees of freedom, assuming the destination and obstacles can either be moving or stationary in its operational range, calculate the motion of the manipulator so that it will reach the destination while avoiding collision between its end-effector and the obstacles.

3.2.3. **Assumptions.** The destination and obstacles are assumed to travel with constant velocities within any short time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), where \( \Delta t \) is in the order of milliseconds. The motion of the destination in a state variable equation is:

\[
\dot{x}_D(t) = A x_D(t)
\]  

where \( A \) was defined previously in (3.2.1.5a). Similarly, the motion of the \( n^{th} \) obstacle is:

\[
\dot{x}_{O(n)}(t) = A x_{O(n)}(t)
\]  

Assuming that the external sensors will provide data at the beginning of every time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\): \( x_R(t_{\text{present}}), x_D(t_{\text{present}}), \) and \( x_{O(n)}(t_{\text{present}}) \).

For the manipulator, the joint torque and joint acceleration vectors are assumed constant during this short interval, i.e. \( u_R(t) = u_R(t_{\text{present}}) \), and the joint torque vector is approximated as:

\[
\tau_R(t) = f( x_R(t_{\text{present}}) ) u_R(t_{\text{present}}) + g( x_R(t_{\text{present}}) )
\]

Note that the sensors provide the Cartesian coordinates and velocities of the destination and obstacles. This data is converted into joint space state variable by means of inverse kinematics:
\begin{align}
\theta_D(t) &= k^{-1} [ r_D(t) ] \\
\dot{\theta}_D(t) &= \nabla_{\theta} k^{-1} [ r_D(t) ] \dot{r}_D(t) \\
\theta_{O(n)}(t) &= k^{-1} [ r_{O(n)}(t) ] \\
\dot{\theta}_{O(n)}(t) &= \nabla_{\theta} k^{-1} [ r_{O(n)}(t) ] \dot{r}_{O(n)}(t)
\end{align}

3.2.4. **Approach.** The same approach used for the motion planning of a mobile robot presented in the previous chapter is used here. The problem is formulated in two stages:

(i) Solve for the torque that pushes the end-effector toward the destination:

\[
\min_{v(t)} \int_{t_{\text{present}}}^{t_{\text{present}}+\Delta t} \left\{ [x_{R(\text{no})}(t)-x_D(t)]^T R_1 [x_{R(\text{no})}(t)-x_D(t)] + \tau_v^T(t) R_2 \tau_v(t) \right\} dt
\]

\[\text{s.t.}\]

\[
\begin{align}
\dot{x}_{R(\text{no})}(t) &= A x_{R(\text{no})}(t) + B v(t) \\
\dot{x}_D(t) &= A x_D(t) \\
\tau_v(t) &= f[ x_{R(\text{no})}(t_{\text{present}}) ] v(t) + g[ x_{R(\text{no})}(t_{\text{present}}) ] \\
R_1 &= \text{diag}( a_1, a_2, ..., a_N, a_{N+1}, a_{N+2}, ..., a_{2N} ) \\
R_2 &= \text{diag}( b_1, b_2, ..., b_N )
\end{align}
\]

for \(0 \leq a_n \leq 1, n = 1, 2, ..., 2N,\) and \(0 < b_n \leq 1, n = 1, 2, ..., N.\)

(ii) Solve for the torque \(\tau_R(t)\) that pushes the end-effector in a new path that deviates from the path \(x_{R(\text{no})}^*(t)\) computed in (i) to avoid obstacles:
\[
\min \int_{t_{\text{present}}}^{t_{\text{present}}+\Delta t} \left\{ [x_R(t) - x_{R(\text{no})}(t)]^T R_3 [x_R(t) - x_{R(\text{no})}(t)] + \tau_R^T(t) R_4 \tau_R(t) \right\} dt
+ \sum_{n=1}^{N} E[ x_R(t), x_{O(n)}(t) ] dt \quad (3.2.4.3)
\]

s.t.

\[
\begin{align*}
\dot{x}_R(t) &= A x_R(t) + B u_R(t) \quad (3.2.4.4a) \\
\dot{x}_{O(n)}(t) &= A x_{O(n)}(t) \quad (3.2.4.4b) \\
\tau_R(t) &= f( x_R(t_{\text{present}}) ) u_R(t) + g( x_R(t_{\text{present}}) ) \quad (3.2.4.4c) \\
R_3 &= \text{diag}( c_1, c_2, \ldots, c_N, c_{N+1}, c_{N+2}, \ldots, c_{2N} ) \quad (3.2.4.4d) \\
R_4 &= \text{diag}( d_1, d_2, \ldots, d_N ) \quad (3.2.4.4e)
\end{align*}
\]

for \(0 \leq c_n \leq 1, n = 1, 2, \ldots, 2N\), and \(0 < d_n \leq 1, n = 1, 2, \ldots, N\). The artificial potential energy function \(E[ x_R(t), x_{O(n)}(t) ]\) is given in section 3.3.

### 3.3. Artificial Potential Energy

The same formulation for the artificial potential energy \(E[ x_R(t), x_{O(n)}(t) ]\) used in chapter 2 will be used here:

\[
E[ x_R(t), x_{O(n)}(t) ] = -\frac{[x_R(t) - x_{O(n)}(t)]^T A [x_R(t) - x_{O(n)}(t)]}{[x_R(t) - x_{O(n)}(t)]^T A B B^T A^T [x_R(t) - x_{O(n)}(t)]} \quad (3.3.1)
\]

The matrices \(A\) and \(B\) are:

\[
A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad (3.3.2)
\]
\[
B = \begin{bmatrix}
0 \\
I
\end{bmatrix}
\]

(3.3.3)

For additional description of the properties of this energy function, see section 2.3 in chapter 2.

3.4. Computational Procedures.

The computational procedure is summarized here. Using the same technique in chapter 2, the solution is presented here as follow.

(i) Solve the problem stated in (3.2.4.1) and (3.2.4.2) for no obstacles:

\[
\tau^*_v(t) = f [ x_R(t_{\text{present}}) ] v^*(t) + g [ x_R(t_{\text{present}}) ]
\]

(3.4.1)

\[
v^*(t) = -\frac{1}{2} \left( \frac{\Delta t^5}{20} \Gamma_{11} + \frac{\Delta t^3}{3} \Gamma_{12} + \Delta t f^T [ x_R(t_{\text{present}}) ] R_2 f [ x_R(t_{\text{present}}) ] \right)^{-1}
\]

\[
2 \Delta t f^T [ x_R(t_{\text{present}}) ] R_2 g [ x_R(t_{\text{present}}) ] +
\]

\[
\Delta t^2 \Gamma_{12} B^T ( x_R(t_{\text{present}}) - x_D(t_{\text{present}}) ) +
\]

\[
\frac{\Delta t^3}{3} \Gamma_{11} B^T A^T ( x_R(t_{\text{present}}) - x_D(t_{\text{present}}) ) +
\]

\[
\frac{\Delta t^4}{4} \Gamma_{11} B^T ( x_R(t_{\text{present}}) - x_D(t_{\text{present}}) )
\]

(3.4.2)

\[
x_{R_{\text{no}}}(t) = x_R(t_{\text{present}}) + A x_R(t_{\text{present}}) (t - t_{\text{present}}) + B v^*(t) (t - t_{\text{present}}) +
\]

\[
\frac{1}{2} A B v^*(t) (t - t_{\text{present}})^2
\]

(3.4.3)

\[
\Gamma_{11} = \text{diag}( a_1, a_2, ..., a_N )
\]

(3.4.4)

\[
\Gamma_{12} = \text{diag}( a_{N+1}, a_{N+2}, ..., a_{2N} )
\]

(3.4.5)

(ii) Solve the problem stated in (3.2.4.3) and (3.2.4.4) for obstacles:

\[
\tau^*_R(t) = f [ x_R(t_{\text{present}}) ] u_R^*(t) + g [ x_R(t_{\text{present}}) ]
\]

(3.4.6)
\[ u^*_R(t) = v^*(t) - \]
\[ \left[ \frac{\Delta t^5}{20} \Gamma_{31} + \frac{\Delta t^3}{3} \Gamma_{32} + \Delta t f^T[x_R(t_{\text{present}})] R_4 f[x_R(t_{\text{present}})] + \right. \]
\[ \sum_{n=1}^{N} \frac{\Delta t^d I}{8 \Delta t \eta_n^T \eta_n + 4 \eta_n^T \eta_n} \left]^{-1} \right. \]
\[ \left. \left[ \Delta t f^T[x_R(t_{\text{present}})] R_4 f[x_R(t_{\text{present}})] v^*(t) + \right. \right. \]
\[ \Delta t f^T[x_R(t_{\text{present}})] R_4 g[x_R(t_{\text{present}})] + \]
\[ \sum_{n=1}^{N} \frac{\Delta t^2 \eta_n + \Delta t^2 \eta_n}{4 \Delta t \eta_n^T \eta_n + 2 \eta_n^T \eta_n} \right] \]
(3.4.7)

\[ x^*_R(t) = x_R(t_{\text{present}}) + A x_R(t_{\text{present}}) (t - t_{\text{present}}) + B u^*_R(t) (t - t_{\text{present}}) + \]
\[ \frac{1}{2} A B u^*_R(t) (t - t_{\text{present}})^2 \]
(3.4.8)

\[ \eta_n = [I 0] \{ x_R(t_{\text{present}}) - x_O(n)(t_{\text{present}}) \} \]
(3.4.9)

\[ \dot{\eta}_n = [0 I] \{ x_R(t_{\text{present}}) - x_O(n)(t_{\text{present}}) \} \]
(3.4.10)

\[ \Gamma_{31} = \text{diag}(c_1, c_2, \ldots, c_N) \]
(3.4.11)

\[ \Gamma_{32} = \text{diag}(c_{N+1}, c_{N+2}, \ldots, c_{2N}) \]
(3.4.12)

For convergence analysis, the state of the robot with respect to that of the destination at the end of the time interval, i.e. at \( t = t_{\text{present}} + \Delta t \) is computed. For the case of having no obstacles, the solution in (i) is used, and the error is:

\[ \varepsilon (t_{\text{present}} + \Delta t) = M \varepsilon (t_{\text{present}}) + \xi \]
(3.4.13)

where:
\[
M = \begin{bmatrix}
I - \Delta t^4 R_1^{-1} \Gamma_{11} & \Delta t I - \Delta t^3 R_1^{-1} \Gamma_{12} - \frac{\Delta t^5}{16} R_1^{-1} \Gamma_{11} \\
-\Delta t^3 R_1^{-1} \Gamma_{11} & I - \Delta t^2 R_1^{-1} \Gamma_{12} - \frac{\Delta t^4}{8} R_1^{-1} \Gamma_{11}
\end{bmatrix}
\] (3.4.13a)

\[
\xi = \begin{bmatrix}
\frac{\Delta t^3}{2} f^{-1} \left[ x_R(t_{\text{present}}) \right] g \left[ x_R(t_{\text{present}}) \right] \\
\Delta t^2 f^{-1} \left[ x_R(t_{\text{present}}) \right] g \left[ x_R(t_{\text{present}}) \right]
\end{bmatrix}
\] (3.4.13b)

\[
R_1 = f^T \left[ x_R(t_{\text{present}}) \right] R_2 f \left[ x_R(t_{\text{present}}) \right]
\] (3.4.13c)

For the case of obstacles blocking the path, the solution in (ii) is used, and the error is:

\[
\tilde{e}(t_{\text{present}} + \Delta t) = \tilde{M} \tilde{e}(t_{\text{present}}) + \xi
\] (3.4.14)

where:

\[
\tilde{M} = \begin{bmatrix}
I & \Delta t I \\
0 & I
\end{bmatrix}
\] (3.4.14a)

\[
\xi = \begin{bmatrix}
\frac{\Delta t^3}{2} f^{-1} \left[ x_R(t_{\text{present}}) \right] g \left[ x_R(t_{\text{present}}) \right] \\
\Delta t^2 f^{-1} \left[ x_R(t_{\text{present}}) \right] g \left[ x_R(t_{\text{present}}) \right]
\end{bmatrix}
\]

\[
\xi = \begin{bmatrix}
\frac{\Delta t^3}{t} \left[ x_R(t_{\text{present}}) \right] R_4^{-1} f^T \left[ x_R(t_{\text{present}}) \right] \kappa \\
\left[ x_R(t_{\text{present}}) \right] R_4^{-1} f^T \left[ x_R(t_{\text{present}}) \right] \kappa
\end{bmatrix}
\] (3.4.14b)

\[
\kappa = \sum_{n=1}^{N} \frac{\Delta t^3 \dot{\eta}_n + \Delta t^2 \eta_n}{4 \Delta t \dot{\eta}_n^T \eta_n + 2 \eta_n^T \eta_n}
\] (3.4.14c)
3.5. Example.

In this section, an analytical model is developed for a two-link, two-degree-of-freedom planar manipulator shown in figure 3.1 below. Joint 0 is the base, joint 1 the elbow, joint 2 the tip of the end-effector. The manipulator can rotate (pitch) at joint 0 and 1. Let the positive pitch angle be above the horizontal line and negative otherwise. Let $\theta_0$ and $\theta_1$ be the pitch angles at joints 0 and 1 respectively, and $l_1$ and $l_2$ the lengths of link 1 (between joints 0 and 1) and link 2 (between joints 1 and 2). The direct kinematic relation $k(\theta)$ can be given as:

$$
k(\theta) = l_1 \begin{bmatrix} \cos(\theta_0) \\ \sin(\theta_0) \end{bmatrix} + l_2 \begin{bmatrix} \cos(\theta_0+\theta_1) \\ \sin(\theta_0+\theta_1) \end{bmatrix}
$$

(3.5.1)

which gives a closed-form inverse kinematic relation:

$$
k^{-1}(r) = \begin{bmatrix} \tan^{-1}(\frac{y}{x}) \pm \cos^{-1}(\frac{l_1}{2 ||r||} + \frac{||x||}{2 l_1} - \frac{l_2^2}{2 ||x|| l_1}) \\ -\pi \pm \cos^{-1}(\frac{l_1}{2 l_2} + \frac{l_2}{2 l_1} - \frac{||x||^2}{2 l_1 l_2}) \end{bmatrix}
$$

(3.5.2)

where $x$ and $y$ are the components of $r$. Equation (3.5.2) is used to compute the desired state $x_D(t_{\text{present}})$ of the manipulator and the state $x_D(t_{\text{present}})$ of the obstacles.

Assuming each link is a cylinder having uniformly distributed mass, $m_1$ and $m_2$ being the masses of links 1 and 2 respectively, then the direct dynamic relation is:

$$
\begin{bmatrix} \tau_0(t) \\ \tau_1(t) \end{bmatrix} = \begin{bmatrix} \frac{1}{4} l_1^2 + m_2 l_2^2 + \frac{1}{4} m_2 l_2^2 + m_2 l_2 \cos(\theta_1(t)) & \frac{1}{4} m_2 l_2^2 + m_2 l_2 \cos(\theta_1(t)) \\ \frac{1}{4} m_2 l_2^2 + m_2 l_2 \cos(\theta_1(t)) & \frac{1}{2} l_2^2 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_0(t) \\ \ddot{\theta}_1(t) \end{bmatrix}
$$
\[
\begin{bmatrix}
-m_2l_2 \dot{\theta}_1(t) \sin \theta_1(t) + \left( \frac{m_1}{2} + m_2 \right) l_1 \dot{\theta}_1(t) \cos \theta_1(t) + \left( \frac{m_2}{2} \right) l_2 \cos \theta_1(t) \dot{\theta}_1(t) \\
\frac{m_2}{2} l_2 \cos \theta_1(t) \dot{\theta}_1(t)
\end{bmatrix}
\]

(3.5.3)

Figure 3.1. Kinematics of a 2 Link, 2 DOF Manipulator
3.6. Simulation Results.

In this section, computer simulations are presented. A 3-link planar manipulator is used. The simulations demonstrate the motion of the end-effector of the manipulator reaching out to pick a floating object while avoiding collision with other floating objects.

Consider the 3-link planar manipulator depicted in figure 3.2. Its base is on the left side of the figure. Links 1 and 2 are 120 centimeters long each, and link 3 (the end-effector) 30 centimeters long. The large circle is the target (destination) and the six small circles the obstacles. In this section, seven examples are shown to cover all possible cases. The objective is to move the end-effector to the destination while avoiding collision with the six obstacles.

Example 1. Consider figure 3.3. In this experiment, the destination and obstacles are stationary. The obstacles are arranged as a trap around the destination. If the end-effector moves under the trap, it cannot reach the destination because of its kinematics limitation. The movements of the manipulator are shown in this figure: The end-effector moves up and then toward the destination. Figure 3.4 shows the relative positions (x and y components, x in the horizontal direction, and y in the vertical direction) of the obstacles and destination with respect to the end-effector as functions of time.

Example 2. Consider figure 3.5. In this experiment, the destination is stationary and the obstacles moving. Initially, the obstacles are set as a trap around the destination. Each obstacle is moving downward and to the left. The manipulator moves its end-effector up in the beginning and toward the destination when the obstacles clear the trap around the destination. Figure 3.6 shows the relative positions (x and y components, x in the horizontal direction, and y the vertical direction) of the obstacles and destination with respect to the end-effector as functions of time.
Example 3. Consider figure 3.7. In this experiment, the destination is moving up and out of the trap by the obstacles. The obstacles are set as a trap around the initial position of the destination. The obstacles are stationary. The manipulator moves its end-effector upward to meet the destination at the outside of the trap. Figure 3.8 shows the relative positions (x and y components, x the horizontal direction, y the vertical direction) of the obstacles and destination with respect to the end-effector as functions of time.

Example 4. Consider figure 3.9. In this experiment, the destination and obstacles are moving. Initially, the obstacles are set as a trap around the destination. As the obstacles move, an opening is created and the manipulator moves its end-effector to reach the destination through this opening. Figure 3.10 shows the relative positions (x and y components, x the horizontal direction, y the vertical direction) of the obstacles and destination with respect to the end-effector as functions of time.

Example 5. Consider figure 3.11. In this experiment, the destination is stationary, three obstacles are moving and three other obstacles are stationary. The moving three obstacles initially block the destination from the end-effector. As they move, an opening is created. The three stationary obstacles stay behind the destination. The manipulator moves its end-effector up at first, trying to go over the trap in a similar motion to that of example 1. As soon as it sees the opening, the manipulator moves its end-effector through this opening and to the destination. Figure 3.12 shows x and y components, x the horizontal direction, y the vertical direction) of the obstacles and destination with respect to the end-effector as functions of time.

Example 6. Consider figure 3.13. In this experiment, the destination and obstacles are stationary. The obstacles are surrounding the destination. In this case, the opening between any two obstacles is too small for the end-effector in its opening position to get through. This is an illustration of an impossible case. The manipulator moves its end-effector up and reaches out to its limitation but cannot get to the destination. Figure 3.14
shows $x$ and $y$ components, $x$ the horizontal direction, $y$ the vertical direction) of the obstacles and destination with respect to the end-effector as functions of time.

Example 7. Consider figure 3.15. In this experiment, the destination and the obstacles are stationary. The obstacles are not blocking the destination. The destination is out of reach from the manipulator. This is another example of an impossible case. The manipulator moves its end-effector as close to the destination as it can reach. Figure 3.16 shows $x$ and $y$ components, $x$ the horizontal direction, $y$ the vertical direction) of the obstacles and destination with respect to the end-effector as functions of time.

Figure 3.2. Kinematics of a 3-Link, 3-DOF Manipulator.
Figure 3.3. Stationary Destination and Stationary Obstacles
Figure 3.4. Relative Positions of Obstacles and Destination wrt to End-Effector
Figure 3.5. Stationary Destination and Moving Obstacles
Figure 3.6. Relative Positions of Obstacles and Destinations wrt End-Effector
Figure 3.7. Moving Destination and Stationary Obstacles
Figure 3.8. Relative Positions of Obstacles and Destination wrt End-Effector
Figure 3.9. Moving Destination and Moving Obstacles
Figure 3.10. Relative Positions of Obstacles and Destination wrt End-Effector
Figure 3.11. General Case: Combination of Moving and Stationary Objects
Figure 3.12. Relative Positions of Ostacles and Destination wrt End-Effector
Figure 3.13. Impossible Case
Figure 3.14. Relative Position of Obstacles and Destination wrt End-Effector
Figure 3.15. Impossible Case.
Figure 3.16. Relative position of Obstacles and Destination with respect to End-Effector
3.7. Convergence.

In this section, the convergence of the algorithm (i.e. approaching the target to within an arbitrarily small distance) presented in section 2.4 is discussed. Based on the calculation, the convergence is confirmed if at the end of each interval, the difference between the state of the robot and that of the destination is smaller than the difference at the beginning of the interval. For example, for each interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), let \(\epsilon(t)\) be the difference between the state of the robot and that of the destination. Convergence occurs when:

\[
\| \epsilon(t_{\text{present}} + \Delta t) \|_2 < \| \epsilon(t_{\text{present}}) \|_2 \quad \text{(3.7.1)}
\]

In section 2.4, the error at the time \(t = t_{\text{present}} + \Delta t\) for the case of no obstacles is:

\[
\epsilon(t_{\text{present}} + \Delta t) = M \epsilon(t_{\text{present}}) + \xi \quad \text{(3.7.2)}
\]

where \(M\) and \(\xi\) have been defined previously in (3.4.13a) and (3.4.13b), which yields the inequality:

\[
\| \epsilon(t_{\text{present}} + \Delta t) \|_2 \leq \| M \|_2 \cdot \| \epsilon(t_{\text{present}}) \|_2 + \| \xi \|_2 \quad \text{(3.7.3)}
\]

If (3.7.1) is satisfied, then the following condition must be met:

\[
\| M \|_2 \cdot \| \epsilon(t_{\text{present}}) \|_2 + \| \xi \|_2 < \| \epsilon(t_{\text{present}}) \|_2 \quad \text{(3.7.4a)}
\]

or, equivalently,
\[(1 - \| M \|_2) \| e(t_{\text{present}}) \|_2 > \| \xi \|_2 \quad (3.7.4b)\]

For the case of obstacles, the error in (3.7.2) is modified to be:

\[\tilde{e}(t_{\text{present}} + \Delta t) = \tilde{M} \tilde{e}(t_{\text{present}}) + \tilde{\xi} \quad (3.7.5)\]

where \(\tilde{M}\) and \(\tilde{\xi}\) have been defined previously in (3.4.14a) and (3.4.14b), which yields the inequality:

\[\| \tilde{e}(t_{\text{present}} + \Delta t) \|_2 \leq \| \tilde{M} \|_2 \cdot \| \tilde{e}(t_{\text{present}}) \|_2 + \| \tilde{\xi} \|_2 \quad (3.7.6)\]

Similarly, the convergence is satisfied if the following condition is true:

\[(1 - \| \tilde{M} \|_2) \| e(t_{\text{present}}) \|_2 > \| \tilde{\xi} \|_2 \quad (3.7.7)\]

3.8. Conclusion.

In this chapter, the problem of motion planning of a manipulator mounted on a stationary platform for its end-effector to track a moving target while avoiding collision with moving obstacles has been investigated. To address the incompleteness of data which prevents the position planning stage, the position and velocity path planning problems were combined into one single optimization problem. This problem, then, is divided into a sequence of small optimization problems, each yielding a solution over a short interval of time. This interval is the duration between consecutive data available from the external sensors. At the beginning of each interval, new data is acquired and the small optimization
problem makes use of this information in planning the motion within this interval of time. The obstacles are formulated into a source of joint torques required to drive the end-effector away from these obstacles using formulation of an artificial potential energy that generates a force perpendicular to the motion of an object moving in the energy field. An analytical example was given to demonstrate the problem setup. Simulations have been shown to demonstrate the motion of a manipulator reaching out to a moving target while avoiding collision with obstacles, moving and/or stationary.
Chapter 4
Conclusion

In this chapter, a summary of the results from the previous two chapters is given. The solutions to the problem of motion planning for a multi-link manipulator and to the navigation problem are summarized. A brief discussion to other possible applications and extensions is given.

4.1. Introduction.

The motion planning problem for a robot system can be stated as follows: To design a path which includes position and velocity of a robot system from its initial state to a desired destination while avoiding collision with moving obstacles. The destination may also be moving. A robot system can either be a multi-link manipulator or a mobile robot.

It has been shown throughout this work that the above problem can be treated at the level of computation rather than at the level of artificial intelligence reasoning. The requirement of avoiding the moving obstacles are formulated into the potential energy function: The obstacles are considered sources of artificial potential energies that increase monotonically with the chance of the robot colliding with the obstacles. This function is used as a cost to minimize the effort in the motion planning problem. An adaptive
algorithm has been developed to utilize the fact that sensor data can only be obtained at execution. This algorithm, taking in new data from sensors, computes the immediate complete motion which consists of position, velocity, and acceleration within one small interval of time until the next set of new data arrives.

In most previous approaches, the problem of motion planning was often divided into two parts: (i) path planning, and (ii) trajectory planning. In (i), a path is a set of positions, either in Cartesian space (xyz coordinate system) or in joint space (pitch, roll, and yaw coordinate system). In (ii), a trajectory is a set of velocities and accelerations given in one of the two spaces mentioned above. In this way, complete information of the environment is assumed at the outset in those formulations. The planning is done only once at the beginning. Motion is carried out by sending the desired position, velocity and acceleration to the controller of the robot system, point by point, for execution. Some systems accept forces or torques instead. In this case, the direct dynamic relation is needed to transform the motion into these forms.

The above previous approaches have been modified for the situation when the environment is rapidly changing. However, in this method, a path is still planned at the beginning with the assumption that the complete information on the stationary part of the environment is available. Then, a real-time trajectory planner is developed. This planner, constantly monitoring the environment for any blockage in the path, adjusts the velocity and acceleration to avoid the obstacles that cross its path by one of the following means: (i) slow down or speed up along the path to avoid collision, or (ii) deviate from the path for a moment to avoid collision. The first technique, has an advantage of guaranteeing the robot to stay in the optimal path, which is often the shortest path. However, if the obstacles behave irrationally, i.e. one obstacle might move into the path and stop there, slowing down and waiting for the obstacle to clear the path will not solve the problem. The second approach treats this particular situation. However, there are other difficulties in the second
approach, mainly the calculation of how long the deviation from the path is necessary and when the robot should get back to its original path.

4.2. Summary of Results.

The following results have been obtained in this dissertation: (i) formulation of a dynamically changing workspace into closed form mathematical expressions, which was inserted in the cost function for motion planning of the robot system; (ii) formulation of a novel position and velocity dependent artificial energy function to obtain a solution to the dynamic collision avoidance problem; (iii) an adaptive algorithm for the complete motion planning problem; (iv) applications of the solution to the motion planning of a multi-link manipulator and a mobile robot system; and (v) computer simulations to illustrate the results.

In (i), the potential field, in the generalized form developed in this dissertation, is a function of the obstacles' positions and velocities with respect to those of the robot.

In (ii), obstacles are represented in the form of artificial potential energy functions in the criterion function which is minimized in obtaining the desired solution. The novel aspect of our contribution is that the above function allows us to incorporate not only the positions but also the velocities of the obstacles.

In (iii), an adaptive algorithm to solve for the complete motion of the robot is formulated. The solution is often given in terms of position, velocity, and acceleration of the robot system. It has also been shown that the solution in terms of torques can be obtained, as in chapter 3. Assuming that there is a new set of data from the external sensors arriving every $\Delta t$ milliseconds, the algorithm, making use of this set of fresh data, computes the complete motion for the robot within that $\Delta t$ interval. The procedure is repeated at the beginning of every interval when new data arrives.
In (iv), applications are extended to the problem of motion planning for a multi-link manipulator and also for a mobile robot. For a manipulator, it is desired for the end-effector to reach out toward a floating object so that the hand can grasp it. The floating object can either be resulted from the micro-gravity space, or from the coordination of the movement of the base when the robot is asked to pick up an object as it moves. For a mobile robot, the problem is sometimes referred to as the navigation problem. A mobile robot is asked to reach a moving destination, for example, chasing and rescuing an astronaut floating away in space. In both applications, obstacles are moving around the environment, and blocking some possible paths.

In (v), computer simulations for the two applications mentioned above are presented in the format of animations. The simulations are run on a PC, showing a manipulator and a mobile robot reaching a destination.

4.3. Future Directions.

Applications of the real-time techniques developed in this dissertation can be extended to several problems. Consider an unmanned space vehicle to be landed on some planet. The general map of the landing site may be known in advance, but fine details of the road still unknown, these may be "seen" through sensors such as cameras, laser range finders. The technique developed here can be implemented on such a vehicle so that it can roam around on an unknown terrain without human needing to plan the direction of its motion. Another interesting application is to the tactical jet fighters evading heat-seeking missiles while in combat. The missiles are considered obstacles to be avoided. A peacet ime application is the automation of cars on the streets. In this case, other cars are considered obstacles.
For the robot manipulator system, application can be applied to the space shuttle's RMS and the Space Station RMS, remote manipulator systems, often used to launch satellites into orbit. The results can also extended to the problem of coordinating many manipulators. In this scenario, one manipulator needs to avoid collision with the links of another manipulator and possibly itself. Each link can be treated as an obstacle so that the other arm may avoid collision with it.

The work developed in this dissertation has the end-effector avoiding collision with obstacles. Developments can be done to treat the complete manipulator. The technique has been outlined in chapter 3, each point of the link be treated as a point mass. Integration over the length of the link can be carried out to sum the effects of the potential energy fields on each link.

4.4. Conclusion.

The problem of motion planning for a robot system has been treated as one part in the overall computational effort. Both the path and trajectory are computed in real-time as reaction to data from the outside environment, which are provided by the external sensors at execution. An adaptive algorithm has been devised to cope with the partially unknown environment. This approach, even though bringing some work traditionally done at an artificial intelligence reasoning level to a lower level of computation, is not intended to eliminate the AI reasoning level. Rather, it allocates some work to the local number-crunching computer, eliminates communication time between the AI machine and the conventional computer, and speeds up the execution to allow faster reactions to a rapidly changing environment.
References


[66] Sturm, C. Other Demonstrations of the Same Theorem. J. de Mathematiques Pures et Appliques 1 (1836).


Appendix A
Solution of a Regulator Problem

Consider a regulator problem for a linear time invariant system:

$$\min_{u(t)} \int_{t_i}^{t_f} \left[ x^T(t) R_1 x(t) + u^T(t) R_2 u(t) \right] dt$$  \hspace{1cm} (A.1)

s.t.

$$\dot{x}(t) = A x(t) + B u(t)$$  \hspace{1cm} (A.2)

where $x(t)$ is the state, (A.2) the state-variable equation of the system which is characterized by the matrices $A$, $B$, and $C$, $x_d(t)$ the reference state (or desired goal), $u(t)$ the controlled input, $R_1$ and $R_2$ the positive-definite constant weight matrices.

Using the variational method, let $u(t) = u^*(t) + \varepsilon \tilde{u}(t)$ and the corresponding states $x(t) = x^*(t) + \varepsilon \tilde{x}(t)$, where $u^*(t)$ is the solution of (A.1) and (A.2). Then, the variation $\tilde{x}(t)$ can be expressed as:

$$\dot{\tilde{x}}(t) = A \tilde{x}(t) + B \tilde{u}(t)$$  \hspace{1cm} (A.3)
and the objective function of (A.1) can be written as:

\[
\int_{t_i}^{t_f} [ \mathbf{x}^T(t) \mathbf{R}_1 \mathbf{x}(t) + \mathbf{u}^T(t) \mathbf{R}_2 \mathbf{u}(t) ] \, dt = \\
\int_{t_i}^{t_f} [ \mathbf{x}^*^T(t) \mathbf{R}_1 \mathbf{x}^*(t) + \mathbf{u}^*^T(t) \mathbf{R}_2 \mathbf{u}^*(t) ] \, dt \\
+ \varepsilon^2 \int_{t_i}^{t_f} [ \tilde{\mathbf{x}}^T(t) \mathbf{R}_1 \tilde{\mathbf{x}}(t) + \tilde{\mathbf{u}}^T(t) \mathbf{R}_2 \tilde{\mathbf{u}}(t) ] \, dt \\
+ 2 \varepsilon \int_{t_i}^{t_f} [ \tilde{\mathbf{x}}^T(t) \mathbf{R}_1 \mathbf{x}^*(t) + \tilde{\mathbf{u}}^T(t) \mathbf{R}_2 \mathbf{u}^*(t) ] \, dt \tag{A.4}
\]

which is at its minimum when \( \varepsilon = 0 \) and:

\[
\int_{t_i}^{t_f} [ \tilde{\mathbf{x}}^T(t) \mathbf{R}_1 \mathbf{x}^*(t) + \tilde{\mathbf{u}}^T(t) \mathbf{R}_2 \mathbf{u}^*(t) ] \, dt = 0 \tag{A.5}
\]

The solution of (A.3), where the initial condition for \( \tilde{\mathbf{x}}(t) \) is zero at \( t = t_i \) is:

\[
\tilde{\mathbf{x}}(t) = \int_{t_i}^{t} \exp\{ \mathbf{A}(t - \tau) \} \mathbf{B} \tilde{\mathbf{u}}(\tau) \, d\tau \tag{A.6}
\]

Substitute (A.6) into (A.5):
\[ \int_{t_i}^{t_f} \tilde{u}^T(t) \{ R_z u^*(t) + B^T \int_{t}^{t_f} \exp^T(A(\tau - t)) R_1 x^*(\tau) \, d\tau \} \, dt = 0 \]  

(A.7)

Equation (A.7) is valid if and only if:

\[ u^*(t) = -R_z^{-1} B^T \int_{t}^{t_f} \exp^T(A(\tau - t)) R_1 x^*(\tau) \, d\tau \]  

(A.8)

Let us define \( p(t) \) as:

\[ p(t) = \int_{t}^{t_f} \exp^T(A(\tau - t)) R_1 x^*(\tau) \, d\tau \]  

(A.9)

then, \( p(t) \) can be combined with \( x^*(t) \) into a linear system:

\[
\begin{bmatrix}
    x^*(t) \\
    \dot{p}(t)
\end{bmatrix} = 
\begin{bmatrix}
    A & -B R_z^{-1} B^T \\
    -R_1 & -A^T
\end{bmatrix}
\begin{bmatrix}
    x(t) \\
    p(t)
\end{bmatrix}
\]  

(A.10)

where the boundary conditions are: \( p(t_i) = 0 \) and \( x^*(t_f) = x_i \),

\[ x^*(t) = \Theta_{11}(t, t_f) x(t_f) \]  

(A.11)

\[ p(t) = \Theta_{21}(t, t_f) x(t_f) \]  

(A.12)

where:
\[
\exp\left[ \begin{array}{cc}
A & -BR_2B^T \\
-R_1 & -A^T 
\end{array} \right] (t-t_i) \right] = \begin{bmatrix}
\Theta_{11}(t,t_i) \Theta_{12}(t,t_i) \\
\Theta_{21}(t,t_i) \Theta_{22}(t,t_i)
\end{bmatrix}
\] (A.13)

Eliminate \( x(t_i) \) in (A.12) using (A.11), \( p(t) \) is derived as:

\[
p(t) = \Theta_{21}(t,t_i) \Theta_{11}^{-1}(t,t_i) x^*(t)
\] (A.14)

Equation (A.14) above can be substituted into (A.9) and (A.8) for a solution \( u^*(t) \) that optimizes the problem in (A.1) and (A.2). This solution contains \( x^*(t) \) and therefore often known as a feedback solution.

One can compute the solution of (A.1) and (A.2) in terms of its initial condition, \( x(t_i) = x_i \) as follows. Let \( t = t_i \) in equation (A.11), then \( x^*(t_i) \) can be written as:

\[
x^*(t_i) = \Theta_{11}^{-1}(t_i,t_i) x(t_i)
\] (A.15)

Substituting (A.15) into (A.12), the solution \( p(t) \) can be written in terms of the initial condition \( x(t_i) \):

\[
p(t) = \Theta_{21}(t_i,t_i) \Theta_{11}^{-1}(t_i,t_i) x(t_i)
\] (A.16)

this solution is often known as the open-loop solution.

Instead of solving for the right hand side of (A.13), one may solve for \( p(t) \) as follows: Let \( p(t) = K x(t) \), then, equation (A.10) yields:

\[
0 = R_1 + A^T K + KA - KB R_2^{-1} B^T K
\] (A.17)
Appendix B
Solution of an Optimization Problem

Consider an optimization problem:

\[
\min_{\tau_v(t)} \int_{t_{\text{present}}}^{t_{\text{present}}+\Delta t} \left[ [x_{R(\text{no})}(t) - x_{D}(t)]^T R_1 [x_{R(\text{no})}(t) - x_{D}(t)] + \tau_v(t)^T R_2 \tau_v(t) \right] dt
\]  

\hspace{1cm} (B.1)

s.t.

\[
\dot{x}_{R(\text{no})}(t) = A x_{R(\text{no})}(t) + B \nu(t)
\]  

\hspace{1cm} (B.2a)

\[
\dot{x}_{D}(t) = A x_{D}(t)
\]  

\hspace{1cm} (B.2b)

\[
\tau_v(t) = f[ x_{R(\text{no})(t_{\text{present}})} ] \nu(t) + g[ x_{R(\text{no})(t_{\text{present}})} ]
\]  

\hspace{1cm} (B.2c)

\[
R_1 = \text{diag}( a_1, a_2, \ldots, a_N, a_{N+1}, a_{N+2}, \ldots, a_{2N} )
\]  

\hspace{1cm} (B.2d)

\[
R_2 = \text{diag}( b_1, b_2, \ldots, b_N )
\]  

\hspace{1cm} (B.2e)

\[
A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}
\]  

\hspace{1cm} (B.2f)

\[
B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\]  

\hspace{1cm} (B.2g)
If \( v(t) = v \), a constant, for the time interval \([t_{\text{present}}, t_{\text{present}} + \Delta t]\), then equation (B.2a) can be rewritten as:

\[
x_{R(\text{no})}(t) = x_{R}(t_{\text{present}}) + A x_{R}(t_{\text{present}}) (t - t_{\text{present}}) + B v (t - t_{\text{present}}) + \frac{1}{2} A B v (t - t_{\text{present}})^2
\]  

(B.3)

Similarly, equation (B.2b) can be rewritten as:

\[
x_{D}(t) = x_{D}(t_{\text{present}}) + A x_{D}(t_{\text{present}}) (t - t_{\text{present}})
\]  

(B.4)

Substituting (B.2c), (B.3), and (B.4) into the objective function in (B.1), the problem becomes an unconstrained optimization problem:

\[
\min_v \Delta t \left[ f[x_R(t_{\text{present}})] v + g[x_R(t_{\text{present}})] \right]^T R_2 \left[ f[x_R(t_{\text{present}})] v + g[x_R(t_{\text{present}})] \right] + v^T \left[ \frac{\Delta t^5}{20} \Gamma_{11} + \frac{\Delta t^3}{3} \Gamma_{12} \right] v + v^T \left[ \Delta t^2 \Gamma_{12} B^T (x_R(t_{\text{present}}) - x_{D}(t_{\text{present}})) + \frac{\Delta t^3}{3} \Gamma_{11} B^T A^T (x_R(t_{\text{present}}) - x_{D}(t_{\text{present}})) + \frac{\Delta t^4}{4} \Gamma_{11} B^T (x_R(t_{\text{present}}) - x_{D}(t_{\text{present}})) \right]
\]

(B.4)

where:

\[
\Gamma_{11} = \text{diag}(a_1, a_2, ..., a_N) \quad \text{ (B.4a)}
\]

\[
\Gamma_{12} = \text{diag}(a_{N+1}, a_{N+2}, ..., a_{2N}) \quad \text{ (B.4b)}
\]
which yields the solution:

\[
v^* = -\frac{1}{2} \left[ \frac{\Delta t^5}{20} \Gamma_{11} + \frac{\Delta t^3}{3} \Gamma_{12} + \Delta t f^T x_R(t_{\text{present}}) R_2 f x_R(t_{\text{present}}) \right]^{-1} \left[ 2 \Delta t f^T x_R(t_{\text{present}}) R_2 g x_R(t_{\text{present}}) + \Delta t^2 \Gamma_{12} B^T (x_R(t_{\text{present}}) - x_D(t_{\text{present}})) + \frac{\Delta t^3}{3} \Gamma_{11} B^T A^T (x_R(t_{\text{present}}) - x_D(t_{\text{present}})) + \frac{\Delta t^4}{4} \Gamma_{11} B^T (x_R(t_{\text{present}}) - x_D(t_{\text{present}})) \right]
\]

(B.6)

From this solution, the torque \( \tau^*_r(t) \) can be computed using equations (B.2c) and (B.6):

\[
\tau^*_r(t) = f[x_R(t_{\text{present}})] v^*(t) + g[x_R(t_{\text{present}})]
\]

(B.7)

The state \( x^*_R(t_{\text{no}}) \) can also be computed using (B.3) and (B.6)

\[
x^*_R(t_{\text{no}}) = x_R(t_{\text{present}}) + A x_R(t_{\text{present}}) (t - t_{\text{present}}) + B v^*(t) (t - t_{\text{present}}) + \frac{1}{2} A B v^*(t) (t - t_{\text{present}})^2
\]

(B.8)
Appendix C
Derivation of an Integral

Consider an integral:

\[ \int_{t_{\text{present}}}^{t_{\text{present}} + \Delta t} \frac{[x_R(t) - x_{O(n)}(t)]^T A [x_R(t) - x_{O(n)}(t)]}{[x_R(t) - x_{O(n)}(t)]^T A B B^T A^T [x_R(t) - x_{O(n)}(t)]} \, dt \quad (C.1) \]

where:

\[ A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad (C.2) \]

\[ B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (C.3) \]

and:

\[ x_{O(n)}(t) = x_{O(n)}(t_{\text{present}}) + (t - t_{\text{present}}) A x_{O(n)}(t_{\text{present}}) \quad (C.4) \]

\[ x_R(t) = x_R(t_{\text{present}}) + (t - t_{\text{present}}) A x_R(t_{\text{present}}) + 
(t - t_{\text{present}}) B [w + v^*] + \frac{1}{2} (t - t_{\text{present}})^2 A B [w + v^*] \quad (C.5) \]
Substituting (C.4) and (C.5) into the term \([ x_R(t) - x_{O(n)}(t) ]\) in (C.1), we get:

\[
[ x_R(t) - x_{O(n)}(t) ] = [ x_{R(t \text{ present})} - x_{O(n)(t \text{ present})} ] + \\
(t - t_{\text{present}}) A [ x_{R(t \text{ present})} - x_{O(n)(t \text{ present})} ] + \\
(t - t_{\text{present}}) B [ w + v^* ] + \frac{1}{2} (t-t_{\text{present}})^2 A B [ w + v^* ]
\]

(C.6)

Let us partition the vector \([ x_{R(t \text{ present})} - x_{O(n)(t \text{ present})} ]\) as:

\[
[ x_{R(t \text{ present})} - x_{O(n)(t \text{ present})} ] = \\
\begin{bmatrix}
\eta_n \\
\dot{\eta}_n
\end{bmatrix}
\]

(C.7)

then, equations (2.4.2.5c) and (2.4.2.5d) in chapter 2, quoted again below,

\[
\eta_n = [ I \ 0 ] [ x_{R(t \text{ present})} - x_{O(n)(t \text{ present})} ]
\]

\[
\dot{\eta}_n = [ 0 \ 1 ] [ x_{R(t \text{ present})} - x_{O(n)(t \text{ present})} ]
\]

(C.8a)

(C.8b)

are equivalent to (C.7). Substituting (C.2), (C.8a), (C.8b), and (C.6) into (C.1), the numerator in (C.1) becomes:

\[
[x_R(t) - x_{O(n)(t)}]^T A [ x_R(t) - x_{O(n)(t)} ] = \frac{1}{2} [ w + v^* ]^T [ w + v^* ] (t - t_{\text{present}})^3 + \\
\frac{3}{2} \eta_n^T [ w + v^* ] (t - t_{\text{present}})^2 + \\
\eta_n^T [ w + v^* ] (t - t_{\text{present}}) + \\
\dot{\eta}_n^T \eta_n (t - t_{\text{present}}) + \eta_n^T \dot{\eta}_n
\]

(C.9)
Similarly, the denominator in (C.1) becomes:

\[
[x_R(t) - x_{O(n)}(t)]^T A B B^T A^T [x_R(t) - x_{O(n)}(t)] = \frac{1}{4} [w + v^*]^T [w + v^*] (t - t_{\text{present}})^4 + \\
\eta_n^T [w + v^*] (t - t_{\text{present}})^3 + \\
\eta_n^T [w + v^*] (t - t_{\text{present}})^2 + \\
\eta_n^T \eta_n (t - t_{\text{present}})^2 + \\
2 \eta_n^T \dot{\eta}_n (t - t_{\text{present}}) + \eta_n^T \eta_n 
\]

(C.10)

Let:

\[
\tau = (t - t_{\text{present}}) 
\]

then:

\[
d\tau = dt 
\]

(C.12)

The integral in (C.1) can be rewritten as:

\[
\int_{t_{\text{present}}}^{t_{\text{present}} + \Delta t} \frac{[x_R(t) - x_{O(n)}(t)]^T A [x_R(t) - x_{O(n)}(t)]}{[x_R(t) - x_{O(n)}(t)]^T A B B^T A^T [x_R(t) - x_{O(n)}(t)]} dt = \\
\int_0^{\Delta t} \frac{\frac{1}{2} a \tau^3 + \frac{3}{2} b \tau^2 + c \tau + d}{\frac{1}{4} a \tau^4 + b \tau^3 + c \tau^2 + 2 d \tau + e} d\tau 
\]

(C.13)

where:

\[
a = [w + v^*]^T [w + v^*] 
\]

(C.14)

\[
b = \dot{\eta}_n^T [w + v^*] 
\]

(C.15)
\[ c = \eta_n^T \{ w + v^* \} + \eta_n^T \dot{\eta}_n \]  
\[ d = \eta_n^T \dot{\eta}_n \]  
\[ e = \eta_n^T \eta_n \]

The right hand side of (C.13) can be derived as:

\[
\int_0^{\Delta t} \frac{1}{4} \frac{a \tau^4 + b \tau^3 + c \tau^2 + 2d \tau + e}{\frac{1}{2} a \tau^3 + \frac{3}{2} b \tau^2 + c \tau + d} \, dt = \frac{1}{2} \ln \left\{ \frac{\Delta t^4}{4} a + \Delta t^3 b + \Delta t^2 c + \Delta t d + e \right\} - \frac{1}{2} \ln \{ e \}
\]

where the constants a, b, c, d, and e were defined in (C.14), (C.15), (C.16), (C.17), and (C.18).
Appendix D
Norm of a Square Matrix

Consider a square matrix $M$. Its $l_2$ norm, defined as:

$$
\| M \|_2 = \max_{\| c \|_2} \frac{\| M c \|_2}{\| c \|_2} \quad (D.1)
$$

for all vector $c \neq 0$. Such a norm, can be computed as:

$$
\| M \|_2 = \sqrt{\max\{ \text{eigenvalue} (M^T M) \}} \quad (D.2)
$$

For the matrix $M$ defined in equation (2.5.8) of chapter 2, which is recited below,

$$
M = \begin{bmatrix}
(1-0.5\Delta t^2) & 0 & (\Delta t-0.5\sqrt{3}\Delta t^2) & 0 \\
0 & (1-0.5\Delta t^2) & 0 & (\Delta t-0.5\sqrt{3}\Delta t^2) \\
(\Delta t) & 0 & (1-\sqrt{3}\Delta t) & 0 \\
0 & (\Delta t) & 0 & (1-\sqrt{3}\Delta t)
\end{bmatrix}, \quad (D.3)
$$

its norm can be computed as follows. First, the product of $M^T$ and $M$ is derived:
\[
M^T M = \begin{bmatrix}
[(1 - \frac{\Delta t^2}{2}) + \Delta t^2] I & [(1 - \frac{\Delta t^2}{2})(\Delta t - \sqrt{3} \Delta t^2) - (\Delta t - \sqrt{3} \Delta t^2)] I \\
[(1 - \frac{\Delta t^2}{2})(\Delta t - \sqrt{3} \Delta t^2) - (\Delta t - \sqrt{3} \Delta t^2)] I & [(\Delta t - \sqrt{3} \Delta t^2)^2 + (1 - \sqrt{3} \Delta t^2)] I
\end{bmatrix}
\]
(D.4)

where the identity matrix \( I \) in the right hand side of (D.4) is of dimension 2 by 2. Then, the eigenvalues of \( M^T M \) can be derived by solving the characteristic equation:

\[
\det(\lambda I - M^T M) = 0
\]
(D.5)

For the matrix \( M \) defined in (D.3), equation (D.5) becomes:

\[
0 = \left[ 1 - 2\lambda + \lambda^2 - 2\sqrt{3} \Delta t + 2\sqrt{3} \Delta t \lambda + 4 \Delta t^2 - 4 \Delta t^2 \lambda - \sqrt{3} \Delta t^3 \\
+ \sqrt{3} \Delta t^3 \lambda + \frac{1}{4} \Delta t^4 - \Delta t^4 \lambda \right]^2
\]
(D.6)

with the solution:

\[
\lambda_1 = 1 - \sqrt{3} \Delta t + 2 \Delta t^2 - 0.5 \sqrt{3} \Delta t^3 + 0.5 \Delta t^4 \\
+ \sqrt{3} \Delta t^2 - 4\sqrt{3} \Delta t^3 + 7.75 \Delta t^4 - 3\sqrt{3} \Delta t^5 + 2.75 \Delta t^6 - 0.5\sqrt{3} \Delta t^7 + 0.25 \Delta t^8
\]
(D.7a)

\[
\lambda_2 = 1 - \sqrt{3} \Delta t + 2 \Delta t^2 - 0.5 \sqrt{3} \Delta t^3 + 0.5 \Delta t^4 \\
+ \sqrt{3} \Delta t^2 - 4\sqrt{3} \Delta t^3 + 7.75 \Delta t^4 - 3\sqrt{3} \Delta t^5 + 2.75 \Delta t^6 - 0.5\sqrt{3} \Delta t^7 + 0.25 \Delta t^8
\]
(D.7b)

\[
\lambda_3 = 1 - \sqrt{3} \Delta t + 2 \Delta t^2 - 0.5 \sqrt{3} \Delta t^3 + 0.5 \Delta t^4 \\
- \sqrt{3} \Delta t^2 - 4\sqrt{3} \Delta t^3 + 7.75 \Delta t^4 - 3\sqrt{3} \Delta t^5 + 2.75 \Delta t^6 - 0.5\sqrt{3} \Delta t^7 + 0.25 \Delta t^8
\]
(D.7c)
\[
\lambda_4 = 1 - \sqrt{3} \Delta t + 2 \Delta t^2 - 0.5 \sqrt{3} \Delta t^3 + 0.5 \Delta t^4
- \sqrt{3} \Delta t^2 - 4\sqrt{3} \Delta t^3 + 7.75\Delta t^4 - 3\sqrt{3} \Delta t^5 + 2.75\Delta t^6 - 0.5\sqrt{3} \Delta t^7 + 0.25\Delta t^8
\]  
(D.7d)

The \( l_2 \) norm of \( M \) is the maximum of the four eigenvalues above, or:

\[
\| M \|_2 = 1 - \sqrt{3} \Delta t + 2 \Delta t^2 - 0.5 \sqrt{3} \Delta t^3 + 0.5 \Delta t^4
+ \sqrt{3} \Delta t^2 - 4\sqrt{3} \Delta t^3 + 7.75\Delta t^4 - 3\sqrt{3} \Delta t^5 + 2.75\Delta t^6 - 0.5\sqrt{3} \Delta t^7 + 0.25\Delta t^8
\]  
(D.8)