Chapter 1

Introduction

1.1 What is Motion Strategy?

What is Motion Strategy, and why does the subject deserve a whole course? Across a wide array of disciplines, many problems arise that involve determining how to place complicated bodies into motion in complicated environments. A typical robotics task is to determine automatically how to navigate a mobile robot from one location to another in a cluttered building. Other examples from robotics include manipulator arms in a factory, satellites in space that are guided by thrusters, and humanoid robots in an ordinary household. Many disciplines involve reasoning about geometric information and performing simulations in a virtual environment. For example, the field of virtual prototyping is revolutionizing many industries by allowing designers of mechanical systems to evaluate a proposed design in simulation without having to construct a physical prototype. Imagine, for example, the time, expense, and lives that could be saved if it is determined early in an automobile design process that a particular vehicle could easily topple over under certain braking conditions. This would certainly beat learning about this problem after using a stunt driver to evaluate a physical prototype (or even worse, to discover the problem while the vehicle is in actual use). Other examples that involve virtual environments include character animation for video games and movies, and verification of wheelchair access in proposed architectural designs. Motion strategy problems can even arise in disciplines that appear to use very different principles. For example, in pharmaceutical drug design, chemists would like to determine whether flexible molecules can bend and twist themselves to fit into a protein cavity while exerting little energy.[1] One can imagine a "ball and stick" model of a molecule that is attempting to guide itself (by nature) into a pocket derived from a geometric model of the protein.

The intention of this course is to provide a unified treatment of motion strategy problems by focusing on the mathematical and algorithmic principles that are common to this wide array of disciplines. This would certainly not be possible without the wealth of existing research and literature on related topics. Although the primary slant of this course is on computer science aspects, the treatment is heavily influenced by concepts from areas that often fall outside of traditional computer science, such as control theory, robotics, and mechanics. The course is designed to appeal to students and researchers in computer science, and those outside of computer science who have a interest in motion strategy problems in their primary discipline. As much as possible, the course is designed to give a strong interdisciplinary appeal. Connections to existing applications are given with the hope that the reader will be inspired to build on these techniques in his or her discipline, and perhaps find new disciplines to which motion strategy concepts have yet to be applied.

1.2 A Motivational Problem

This section introduces and solves a simple motion strategy problem that serves as a motivational example. The problem formulation is easy to understand, and the solution is elegant. In Section 1.3 a broad class of other problems will be described, each of which can be considered as a generalization or extension of this simple problem.

[1] Once docking occurs between a drug molecule and a protein, a desired biological effect is initiated.

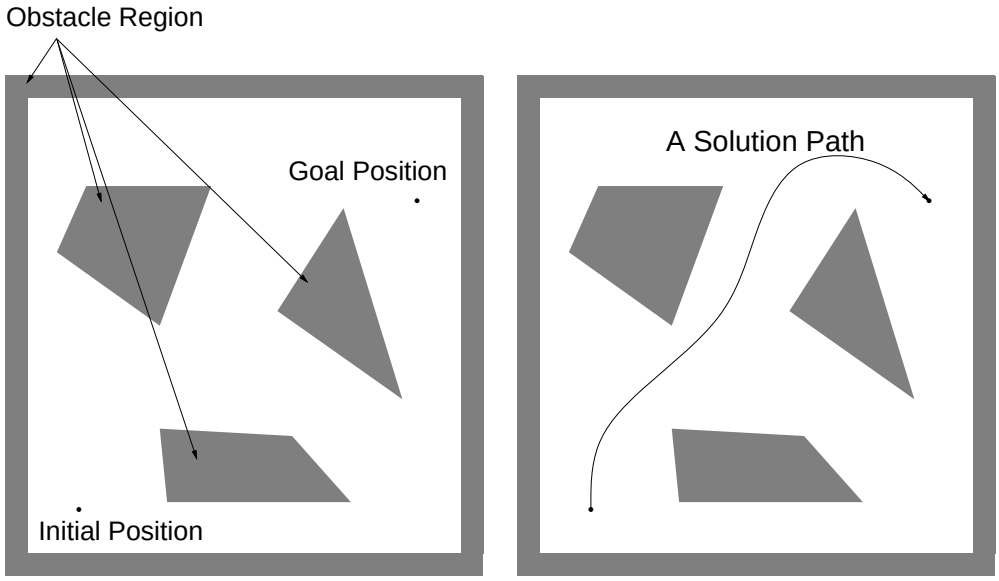

Figure 1.1: A simple illustration of the two dimensional path planning problem: a) The obstacles, initial position, and goal positions are specified as input; b) A path planning algorithm will compute a collision free path from the initial position to the goal position.

Suppose that we have a tiny mobile robot that can move along the floor in a building. The task is to determine a path that it should follow from a starting location to a goal location, while avoiding collisions. A reasonable model can be formulated by assuming that the robot is a moving point in a two-dimensional environment that contains obstacles.

Let $\mathcal{W} = \mathbb{R}^2$ denote a two-dimensional *world* which contains a point *robot*, denoted by $\mathcal{A}$. A subset, $\mathcal{O}$, of the world is called the *obstacle region*. Let the remaining portion of the world, $\mathcal{W} \setminus \mathcal{O}$ be referred to as the *free space*. The task is to design an algorithm that accepts an obstacle region defined by a set of polygons, an initial position, and a goal position. The algorithm must return a path that will bring the robot from the initial position to the goal position, while only traversing the free space.

DEFINITIONS:

Symbol	Name	Definition
$\mathcal{W}$	World (or Workspace)	$\mathbb{R}^2$ (Cartesian plane)
$\mathcal{A}$	Robot	A point in $\mathcal{W}$
$\mathcal{O}$	Obstacle Region	A polygonal subset $\mathcal{W}$ to be avoided by $\mathcal{A}$
$A \setminus B$	set minus	The set of all points in A but not in B
$\mathcal{W} \setminus \mathcal{O}$	Free space	A polygonal subset of $\mathcal{W}$ that can be traversed by $\mathcal{A}$
	Convex	X is convex if the line segment between any two points lies in X
	Complete Algorithm	An algorithm that always returns a solution when one exists

1.3 The Big Picture

A simple, elegant solution was possible for the path planning problem presented in Section 1.2; however, for the vast majority of motion strategy problems, we are not as fortunate. Major complications arise as more challenging problems are considered. This section surveys different kinds of problems and the difficulties they can cause. This also helps define the scope of the course.

The following problems can be viewed as extensions of the problem from Section 1.2:

- The robot is circular, instead of a point. In this case, it is not too difficult to "grow" the obstacles to

compensate.

- The robot is a polygonal shape that can translate. For complicated shapes, it already becomes difficult to use the "growing" intuition, but it is possible.
- The polygonal robot is also allowed to rotate. A systematic representation is needed; this will lead to the *configuration space*.
- The polygonal robot has to be steered like a car. In this case, it might be difficult to move the robot into certain configurations. Imagine having to parallel park the robot! If might even be the case that the robot does not have reverse. These complication lead to the study of it nonholonomic planning.
- The robot is a point in a 3D polyhedral environment. In this case, the trapezoidal decomposition technique can be nicely extended.
- The robot is a 3D polyhedron that can translate or rotate in a polyhedral world. The problem becomes very complicated, and will eventually be viewed as a search in a six-dimensional configuration space.
- The robot and obstacles are modeled using smooth, algebraic surfaces in a 3D environment. These representations will be handled in the configuration space.
- The robot is a satellite in space, and can only be moved by firing thrusters. In this case, the dynamics of the system must be taken into account.
- The robot is comprised of multiple components that are attached together. An example would be a humanoid robot.
- Unexpected obstacles can appear in the environment.
- Very little is known at all about the environment.
- The robot is unpredictable. For example, it is commanded to move east, but moves a little bit north in the process.
- The robot is not able to determine its precise location in the environment. This is known as the *localization* problem.
- The task is for the robot to manipulate an object in the environment.
- The robot must use a camera to search for other robots in the environment.

The main objective in this course is to present techniques that can analyze and solve many of these kinds of problems.

Bibliography