11-27-2013

Minimum-Energy Exploration and Coverage for Robotic Systems

Serge Antoun Salan

MINIMUM-ENERGY EXPLORATION AND COVERAGE FOR ROBOTIC SYSTEMS

by

Serge Antoun Salan

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Major: Computer Science

The University of Memphis

December 2013

ACKNOWLEDGMENT

I gratefully acknowledge Dr. David Lin and Dr. Evan Drumwright for their tremendous help during the entire process of writing this dissertation. This work could not be completed without their guidance and expertise in the matter.

ABSTRACT

Salan, Serge. A. Ph.D. The University of Memphis. December 2013. Minimum-Energy Exploration and Coverage for Robotic Systems. Major Professor: King-Ip Lin.

This dissertation is concerned with the question of autonomously and efficiently exploring three-dimensional environments. Hence, three robotics problems are studied in this work: the motion planning problem, the coverage problem and the exploration problem. The work provides a better understanding of motion and exploration problems with regard to their mathematical formulation and computational complexity, and proposes solutions in the form of algorithms capable of being implemented on a wide range of robotic systems.

Because robots generally operate on a limited power source, the primary focus is on minimizing energy while moving or navigating in the environment. Many approaches address motion planning in the literature, however few attempt to provide a motion that aims at reducing the amount of energy expended during that process. We present a new approach, we call integral-squared torque approximation, that can be integrated with existing motion planners to find low-energy and collision-free paths in the robot's configuration space.

The robotics coverage problem has many real-world applications such as removing landmines or surveilling an area. We prove that this problem is inherently difficult to solve in its general case, and we provide an approach that is shown to be probabilistically complete, and that aims at minimizing a cost function (such as energy.)

The remainder of the dissertation focuses on minimum-energy exploration, and offers a novel formulation for the problem. The formulation can be directly applied to compare exploration algorithms. In addition, an approach that aims at reducing energy during the exploration process is presented, and is shown through simulation to perform better than existing algorithms.

Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Overview

Exploration in robotics is the problem of moving a robot in an unknown environment to acquire information. When a robot has a priori knowledge of the environment, the problem is named *coverage*. The problems of exploration and coverage encompass a wide range of applications mentioned by the following motivational examples.

The robotics community has been primarily interested in exploration for the purpose of providing robots with the capability of autonomously building *maps*,
i.e. representations of the environment based on spatial information gathered by sensors over time (Wallgrün 2010). Robots can then use maps to navigate and operate in the environment. The frontier-based approach is a popular exploration method for building two-dimensional maps (Yamauchi 1997). Other approaches provide improvements by reducing the cost associated with the exploration (Makarenko et al. 2002; Gonzalez-Banos and Latombe 2002; Thrun et al. 2005).

Robots can become extremely valuable tools in areas where it is dangerous for humans to operate, e.g. disaster areas affected by natural or technological hazards, or

areas containing land mines. Nagatani et al. (2011) provide a mapping system that aims at deploying multiple robots in a disaster environment. The robots can then collect three-dimensional data and aid search and rescue crews. Acar et al. (2003) developed an algorithm that guides a robot in a minefield to detect and possibly clear mines.

Other applications of robotic exploration require the robot to provide specific information about an unknown environment. For example, Mars exploration rovers exploring the planet Mars (Washington et al. 1999; NASA 2013), where the mission's goal is to find geological evidence of past water on Mars.

Surveillance is also another application that requires a robot to examine its environment for detecting intrusion or searching for a person or an object of interest. Xu and Stentz (2011) designed an algorithm that can be used by a team of robots to patrol an urban area.

1.2 Current state of scholarship

Given a variety of applications, extensive research was performed on the problem and various solutions are offered. However, there exists important elements that previous research do not address:

- The exploration and coverage problems are typically presented without a complete *formulation*, i.e. a mathematical description of the robotic system, the objectives and the physical constraints. With a lack of formulation, the evaluation of algorithms and their comparison become intricate.

- To the best of the author's knowledge, there exists no publications concerned with the *computational complexity* of the exploration and coverage problems.

- Previous research in exploration and coverage do not provide an apt efficiency measure able to adequately quantify the *quality* of an exploration, and do not propose algorithms that seek to optimize such efficiency measure.

- Existing methods are only designed toward robots whose dynamics are simple to characterize (e.g. a mobile robot) or toward a specific type of robot. They do not possess enough *generality* to be applied on various types of robots and environments.

This dissertation investigates these crucial aspects of the exploration and coverage problems and proposes novel methods and algorithms.

1.3 Energy efficiency

Energy conservation is an important aspect of robot motion. Energy-efficient movements can augment the lifetime of a robot operating on a small energy source. Because exploring robots may operate on a limited power source, reducing the amount of energy expended during the exploration is valuable. The idea of minimizing energy during the exploration is proposed by Mei et al. (2006).

The amount of energy expended by a robot can be divided into mechanical and electrical energy (Mei et al. 2004). When compared to other components, the actuators have a high power consumption (Wang et al. 2005). Therefore this work is focused on the minimization of the amount of energy dissipated through the robot motors.

1.4 Contribution statement

1. The energy-optimal planning problem is applied on a general articulated robotic system. The proposed solution finds low-energy and collision-free paths, and can be applied on high-degree-of freedom robots.

2. The coverage problem is generalized and its polynomial-space hardness is shown.

3. A solution to the generalized coverage problem is proposed. The algorithm finds low-cost collision-free paths and is shown to be probabilistically complete.

4. The exploration problem is formulated as a multi-optimization problem. The formulation is general enough to be applied on many robotic systems operating under different types of constraints, e.g. energy or time constraints.

5. An exploration algorithm that seeks to minimize the total amount of energy expended during the exploration process is presented. The algorithm is implemented on an articulated robot operating in a three-dimensional workspace.

1.5 Organization

The rest of this manuscript is organized as follows. Chapter 2 provides an overview of subjects in robotics that are used subsequently. The chapter also has the purpose of introducing the main mathematical symbols used. Chapter 3 presents the major motion planning algorithms employed in robotics. Motion planning is an integral part of every exploration approach, necessary for finding continuous collision-free robot motions. The chapter also proposes an approach for the energy-optimal motion planning problem. The generalized coverage problem is presented in Chapter 4, and an overview of existing methods is provided. The chapter mainly explores the problem's computational complexity and introduces novel coverage algorithms. Chapter 5 provides a broad overview of exploration methods, and proposes a formulation to the problem. The chapter then shows a novel exploration algorithm designed to minimize the amount of energy expended during the robot's motion, and gives a comparison of different exploration strategies. The comparison is performed on an articulated robot in a three-dimensional environment.

Chapter 2

Preliminaries

This chapter gives a general overview of some topics in robotics that are used throughout this manuscript. The reader is referred to robotics textbooks for a greater detail on each topic. Most of the symbols and definitions used in the following chapters are introduced here.

2.1 Specifying the position of a robot

Before solving any problem in robotics, we must have a full description of the position of the robot. We can then use this information for planning and collision checking. A *configuration* is defined as the complete specification of the location of every point of a robot system. The robot's configuration is given by the vector

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}, \tag{2.1}$$

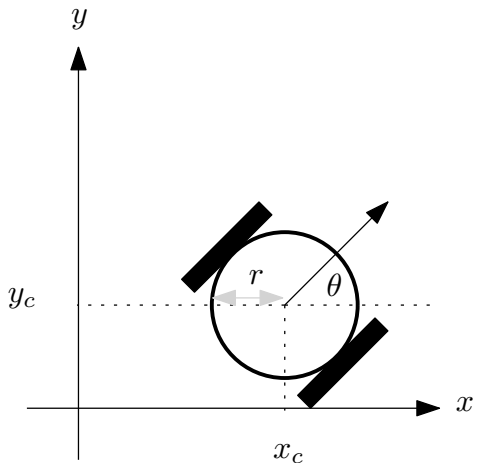


Figure 2.1: A circular mobile robot of radius r in a planar world. x_c and y_c specify the global position of the center of the robot, and θ is an angle that specifies the robot's orientation. The robot's configuration is given by $\mathbf{q} = [x_c, y_c, \theta]^\mathsf{T}$.

where every variable q_i in $\mathbf{q}$ is necessary for specifying the location of every point in the system. The dimension n of $\mathbf{q}$ is the number of *degrees of freedom* of the robot system. The space of all possible configurations is denoted by $\mathcal{C}$ and is known by the configuration space. $\mathbf{q}$ is a point in $\mathcal{C}$. Choset et al. (2005) provides a more rigorous definition of configuration as well as examples of the configuration space of some common robots.

We illustrate the definition of configuration by considering two types of robots in Figures 2.1 and 2.2. Note that the radius of the mobile robot in Figure 2.1 is constant and is not included in the configuration vector.

A robot typically operates in a two or three-dimensional Euclidean space known as the *workspace* and denoted by $\mathcal{W}$. The physical space occupied by a robot is a subset of $\mathcal{W}$ and is computed given the robot's configuration and geometry. It is denoted $\mathcal{R}(\mathbf{q})$.

The robot must be aware of obstacles while moving in its workspace. $\mathcal{W}^{\text{obs}}$ denotes the obstacle space and is a subset of $\mathcal{W}$. A robot contacts obstacles if and only if $\mathcal{R}(\mathbf{q})$ intersects $\mathcal{W}^{\text{obs}}$. The set of all configurations that do not cause the robot to touch obstacles is the free configuration space defined by

$$\mathcal{C}^{\text{free}} := \{\mathbf{q} \in \mathcal{C} \mid \mathcal{R}(\mathbf{q}) \cap \mathcal{W}^{\text{obs}} = \emptyset\}. \tag{2.2}$$

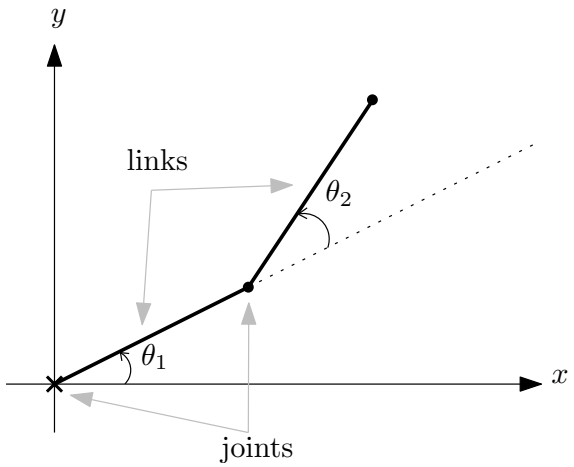
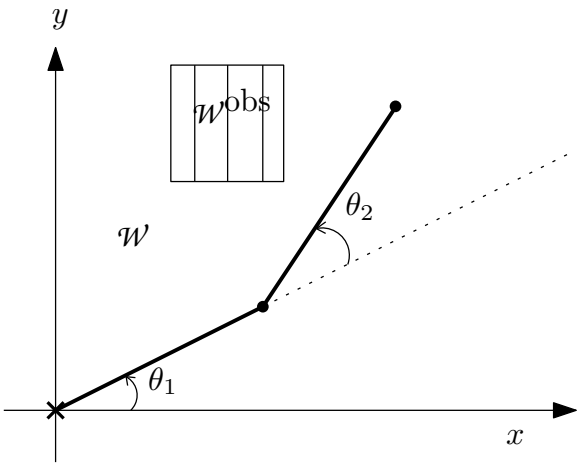


Figure 2.2: A robot arm that possesses two links in a planar world. The first joint has a fixed position. θ_1 and θ_2 specify the first and second joints' angles. The robot's configuration is given by $\mathbf{q} = [\theta_1, \theta_2]^\mathsf{T}$.

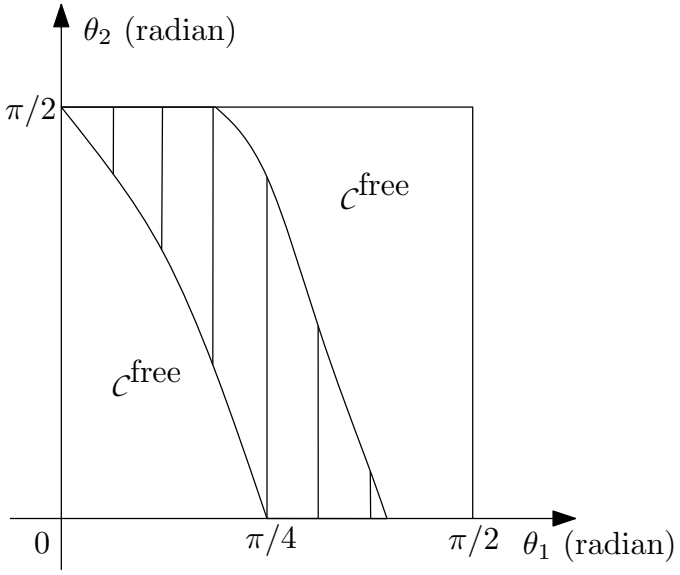
The relation between the robot's physical world and its configuration space is illustrated in Figure 2.3, where a two-degree-of-freedom planar robot arm is considered. Every point on the robot is completely specified by the values θ_1 and θ_2 that denote the angles (expressed in radians) of the first and second joint respectively, such that $\theta_1, \theta_2 \in [0, \frac{\pi}{2}]$. The robot does not occupy any volume and $\mathcal{R}(\mathbf{q})$ is formed by two line segments. The workspace contains a squared obstacle. The resulting configuration space is shown in Figure 2.3(b).

2.2 Perception

The robot gathers information about its environment through its sensors. The configuration of a sensor, denoted **s**, is the complete specification of the sensor's position and orientation. In a two-dimensional workspace, the vector **s** is an element of the special Euclidean group $SE(2)$, a three-dimensional manifold. If the workspace is three dimensional, **s** is an element of the special Euclidean group $SE(3)$, a six-dimensional manifold.



(a) Planar workspace of a robot arm that possesses two links θ_1 and θ_2 specifying the first and second joints' angles respectively. The workspace contains a single squared obstacle.



(b) Corresponding configuration space. $\mathcal{C}^{\text{free}}$ denotes the free configuration space. The striped region corresponds to the set of all configurations that result in collision with $\mathcal{W}^{\text{obs}}$.

Figure 2.3: The relation between the robot's physical world and its configuration space.

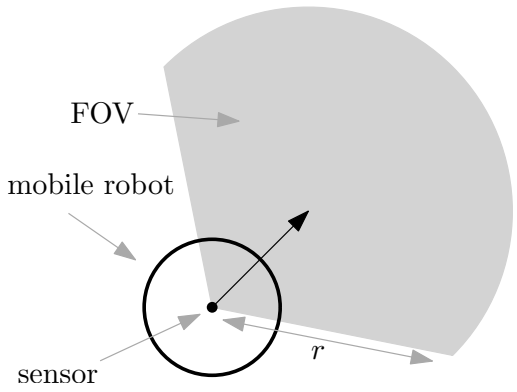


Figure 2.4: Mobile robot equipped with a sensor of range r in a planar world. The sensor is placed at the center of the robot. The region in gray is the robot's field of view (FOV).

The field of view is defined as the union of all observable regions of $\mathcal{W}$ from the robot's sensors. Two types of robots equipped with sensors are illustrated in Figures 2.4 and 2.5.

For a robot such as the one in Figure 2.4, finding $\mathbf{s}$ is straightforward given the robot's configuration. If the robot is a kinematic chain, e.g. the arm in Figure 2.5, a sensor's configuration $\mathbf{s}$ is computed by evaluating the robot's *forward kinematics* equations. Given the robot's configuration, the forward kinematics problem is to determine the position and orientation of the end-effector, or any tool mounted on the robot. Spong et al. (2006) show how to derive the forward kinematics equations for different types robot manipulators.

2.3 Uncertainty in robotics

In the previous sections, we introduce the concepts of a robot and a sensor configurations. It is crucial for a robot that the configuration variables are represented by accurate values. This accuracy allows the robot to localize itself, build a clear representation of its environment and perceive, identify and manipulate physical objects. However in real-world applications, there exists a great deal of uncertainty mainly produced by inaccurate sensor measurements and dynamic environments. In this work we

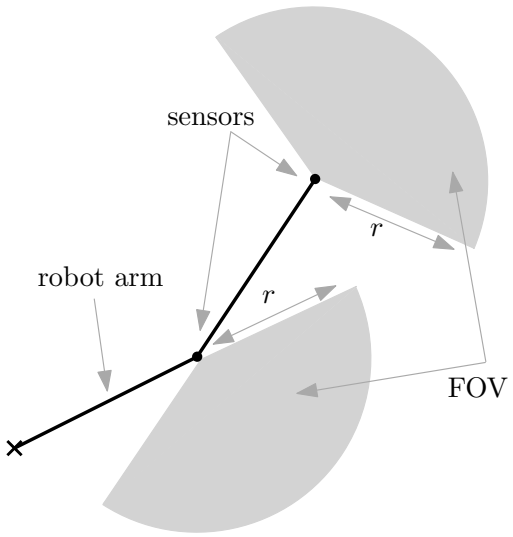


Figure 2.5: Robot arm equipped with two sensors of range r in a planar world. The sensors are mounted on the end-effector and the robot's "elbow". The regions in gray are the robot's field of view (FOV).

assume that the robot's environment is static and its configurations are known, i.e. there is no uncertainty. However, the algorithms presented in the following chapters can be adapted to deal with uncertainty by using statistical and probabilistic techniques. An overview of these techniques is provided by Thrun et al. (2005).

2.4 Motion of a robot

2.4.1 Path and trajectory

The motion of a robot can be represented by a continuous function $\sigma : [0,1] \rightarrow \mathcal{C}^{\text{free}}$, where $\sigma(0)$ and $\sigma(1)$ are the initial and final configurations respectively. Such representation is called a *path* and is merely a geometric description of the motion.

When a timing law is added to the motion, i.e. every configuration in the motion must be achieved at a specified time, the description is called a *trajectory*. A trajectory is a continuous and twice-differentiable function $\mathbf{q} : [0,T] \rightarrow \mathcal{C}^{\text{free}}$, where T is the time length of the trajectory. $\mathbf{q}(t)$ denotes the robot's configuration at time t. The first and second

derivatives of $\mathbf{q}(t)$ are denoted $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$ respectively. They are the robot's velocity and acceleration vectors at time t respectively.

We illustrate these concepts by providing the following example. Consider the robot in Figure 2.2 and suppose that the robot moves by changing the value of its first joint angle θ_1. Figure 2.6 gives an example of a trajectory that interpolates the angles 0, $\pi/2$ and 0 radians, at 0, 0.5 and 1 seconds respectively. The trajectory uses a timing law of cubic splines and ensures continuity of velocity and acceleration. Siciliano et al. (2009) describe various trajectory planning methods.

2.4.2 Dynamics of a n-link robot

We now consider a multi-link robotic system (e.g. the one in Figure 2.2) and we are interested in showing the relationship between force and motion. This relationship can be described mathematically with the dynamic equations of a robot. These equations can be written in different forms, and are derived by Spong et al. (2006) and Choset et al. (2005).

It is common to express the robot's dynamics in matrix form. The Euler-Lagrange equations of motion are applied to obtain the dynamic equations of a robot in the following form:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{u}. \tag{2.3}$$

The vector $\mathbf{u}$ represents the generalized forces acting on the robot. For a n-link robot, the generalized forces consist of motor torque. It is common to denote the vector representing motor torque by τ. $\mathbf{M}(\mathbf{q})$ is the robot's generalized inertia matrix (also called mass matrix). $\mathbf{M}(\mathbf{q})$ is is a symmetric positive definite matrix. The Coriolis and centrifugal forces are denoted by $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}$. The Coriolis and centrifugal forces are velocity dependent and indicate the inter-link dynamic interactions. $\mathbf{g}(\mathbf{q})$ is the vector of gravity forces. Since the inertia matrix $\mathbf{M}(\mathbf{q})$ is invertible, the robot's acceleration is given by

$$\ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})^{-1}\{\mathbf{u} - \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{g}(\mathbf{q})\}. \tag{2.4}$$

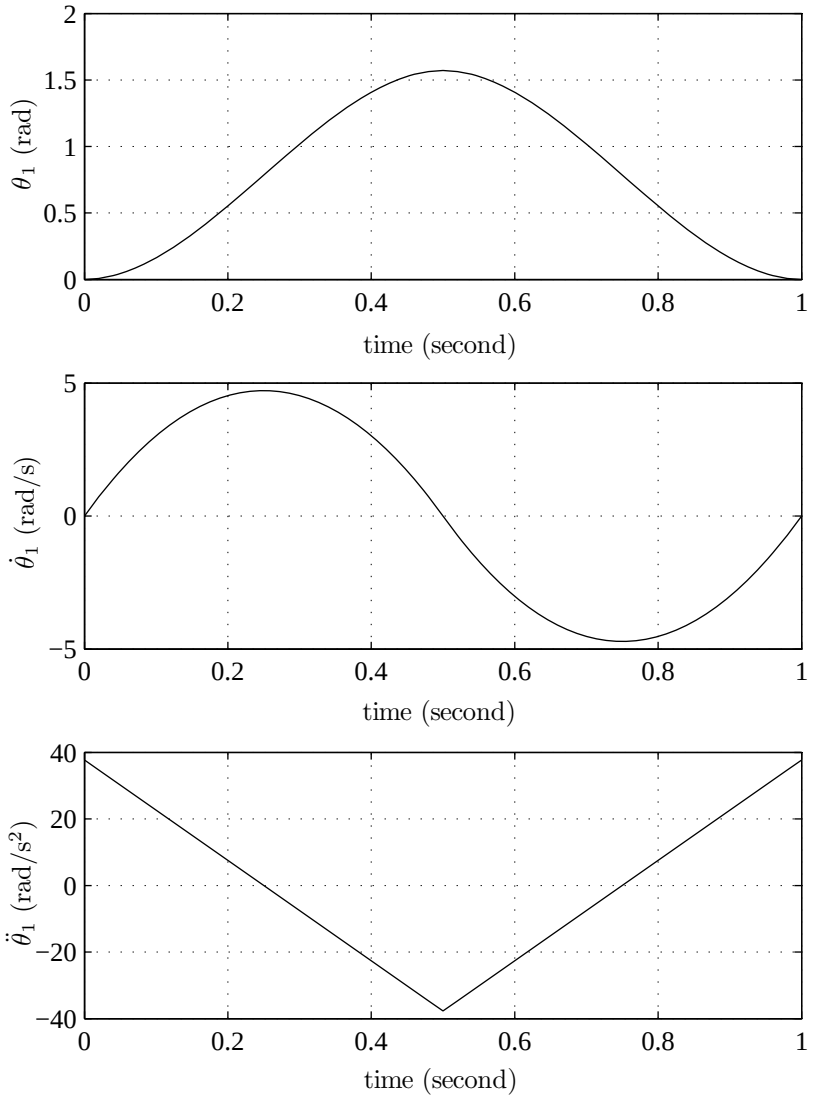


Figure 2.6: Trajectory with a timing law of cubic splines interpolating the angles 0, $\pi/2$ and 0 radians at 0, 0.5 and 1 seconds respectively. The corresponding velocity and acceleration vectors are denoted $\dot{\theta}_1$ and $\ddot{\theta}_1$ respectively.

2.4.3 Inverse dynamics

The inverse dynamics is an essential problem for controlling the motion of a robot; it is defined as follows. Let the variables $\mathbf{q}$ and $\dot{\mathbf{q}}$ be the robotic system's *state*. Given a description of a robotic system and its state, find the force $\mathbf{u}$ required to achieve a specified acceleration $\ddot{\mathbf{q}}$. The recursive Newton-Euler algorithm is used at different instances in the following chapters to solve the inverse dynamics problem. Important dynamics algorithms, including the recursive Newton-Euler algorithm, are described by Featherstone (2007) and Angeles (2003).

Chapter 3

Low-Energy Robot Motion

3.1 Introduction

Arguably one the most fundamental problems in robotics, the motion planning problem arises in many other disciplines including artificial intelligence, computer graphics and computational biology (LaValle 2006). In robotics, the basic motion planning problem is to find a collision-free path in the robot's configuration space, from an initial configuration to a goal region (or correctly report that such path does not exist.) In general, there can be a very large number of solutions to a single instance of the problem. It is then desirable to find a candidate path of minimum cost with respect to a given cost function. The problem defined by Karaman and Frazzoli (2010) is then known by the *optimal planning problem*.

In this chapter we express cost in terms of energy dissipated through the robot motors. The problem we address is the energy-optimal planning problem with zero velocities at end points. To the best of the author's knowledge this problem has no solution for a general articulated robotic system in the presence of obstacles. In our solution, we provide a cost prediction function, and combine it with the existing motion

planning algorithm RRT* (Karaman and Frazzoli 2010) to find low-energy and collision-free paths in the configuration space.

3.1.1 Sampling-based motion planners

Even in its most basic definition, the motion planning problem, also known as the mover's problem, is shown to be inherently difficult to solve (Reif 1979). *Sampling-based* planners are popular algorithms as they are computationally efficient and generally successful in solving the problem. Sampling-based planners generate samples from the configuration space and produce paths by connecting them. Such algorithms include the probabilistic roadmaps (PRM) and the rapidly-exploring random trees (RRT). PRM is introduced by Kavraki et al. (1996). The method first constructs a *roadmap*, a set of connected points in the free configuration space. This step is known by the learning phase. After that, PRM finds paths between pairs of initial and goal configurations. As a result, PRM is mostly effective in addressing the multiple-query variant of the motion planning problem, i.e. multiple initial and goal configurations within a static environment. On the other hand, the RRT algorithm (LaValle 1998) is more suitable as a single-query method. The RRT algorithm incrementally expands a tree structure, rooted at the initial configuration, in the collision-free space. The algorithm returns the path that connects the RRT root to the goal configuration.

PRM and RRT are probabilistically complete (Choset et al. 2005; LaValle and Kuffner 1999), i.e. the probability of their success approaches unity as the number of samples approaches infinity. However when solving the optimal planning problem, Karaman and Frazzoli (2011) show that the probability that they find an optimal solution is almost zero. Other methods that seek to address the optimal planning problem are discussed below.

3.1.2 Motion planning to minimize a cost function

The optimal planning problem has seen an increasing interest in the past decade. The work by Urmson (2003) introduces a biased expansion of a RRT; the tree reaches a goal configuration while favoring low-cost regions. Ferguson (2006) considers the idea of generating a series of paths. Their method repeatedly reconstructs the RRT to consistently improve the quality of the solution. The work by Jaillet et al. (2010) presents a sampling-based planner that uses stochastic optimization methods to globally optimize paths on a configuration-space costmap. This planner is combined with a gradient descent technique by Berenson (2011). They present ways to compute the gradient of the cost in the work space, the task space and the configuration space of the robot.

The methods cited above can generally find low-cost paths but do not ensure optimality. Karaman and Frazzoli (2011) introduce a sampling-based motion planner, called RRT*, proven to be asymptotically optimal, while preserving the exploration properties and the running time of RRT. When adding a new point to the tree, RRT* examines cumulative costs (between the point and the root) and reconnects the tree around the newly added point. Variations of the RRT* are offered by Karaman (2010) to account for kinodynamic constraints, and by Perez et al. (2012) who utilize a cost function based on linear quadratic regulators. They evaluate their method on underactuated systems.

3.1.3 Energy-optimal motion planning

The energy-optimal planning problem has seen an interest in industrial applications to reduce manufacturing costs. The problem is commonly formulated as an optimal control problem. Optimal control theory is described by Chachuat (2007). The problem of finding a minimum-energy point-to-point trajectory is solved on a three-degree-of-freedom industrial robot using numerical methods (von Stryk and Schlemmer 1994). The negative formulation of Pontryagin's maximum principle is applied by Galicki (1998) to guide a planar robot in a two-dimensional environment with

obstacles. Gregory et al. (2012) solve two planning problems on a planar robot manipulator with two degrees of freedom: the path tracking and the intersection-free trajectory planning problems. In their solution, the optimal control problem is reformulated into a calculus of variations problem.

There exists a large literature (not discussed here) on time-optimal planning. The bi-objective optimization problem of finding a control that minimizes time and energy has been studied by Shiller (1994) and Verscheure et al. (2008). Their work uses a utility function expressed as a weighted sum.

The methods cited in this section provide various solutions to the energy-optimal planning problem. However, these solutions only consider planar robots, low-degree-of-freedom robots, point-to-point motions or obstacle-free environments. We propose applying this problem on a general articulated robot in an environment with obstacles by using the existing motion planner RRT*. We require that the starting and ending velocities are zero. This condition encompass a large number of problems where a robot is initially idle and must safely reach a destination and come to a complete stop. However, the method presented here might not be practical for problems with nonzero or high velocities at endpoints. It is also shown, later in this chapter, that accurate energy prediction is difficult to achieve in high-speed motions.

This chapter is structured as follows. Section 3.2 defines the energy-optimal motion planning problem. Section 3.3 provides a detailed description of existing motion planners. Section 3.4 introduces a method, we call integral-squared torque approximation, integrated with RRT* to form a solution to the energy-optimal planning problem. We evaluate our method in a simulated environment in Section 3.5.

3.2 Problem Definition

The optimal planning problem is defined by Karaman and Frazzoli (2010). Some of their definitions must be restated here in order to present, later in this section, the energy-optimal planning problem.

3.2.1 Notation

Let $\mathcal{W}$ be a bounded region in a three-dimensional space that represents the robot's physical world, and let $\mathcal{W}^{\text{obs}} \subset \mathcal{W}$ be the occupied (also called obstacle) space.

Let $\mathcal{C}$ be the n-dimensional configuration space of the robot. $\mathcal{C}^{\text{free}}$ denotes the free state space. Let $\mathbf{q} \in \mathcal{C}$ be a state. The physical space occupied by the robot at $\mathbf{q}$ is denoted by $\mathcal{R}(\mathbf{q})$. $\mathbf{q} \in \mathcal{C}^{\text{free}}$ if and only if $\mathcal{R}(\mathbf{q})$ does not intersect with $\mathcal{W}^{\text{obs}}$.

3.2.2 Optimal planning problem

A continuous collision-free path is represented by the function $\sigma : [0,1] \to \mathcal{C}^{\text{free}}$. Let $\mathcal{S}$ denote the set of all continuous collision-free paths and let $c : \mathcal{S} \to \mathbb{R}^+$ be a function that assigns a nonnegative quantity to a path in $\mathcal{S}$.

Given an initial configuration $\mathbf{q}_0$ and a goal region $\mathcal{C}^{\text{goal}}$, the optimal motion planning problem is to find a path $\sigma \in \mathcal{S}$ that minimizes c or correctly report that such path does not exist. More formally:

$$\text{minimize} \quad : \quad c(\sigma) \tag{3.1}$$

$$\text{subject to} \quad : \quad \sigma(0) = \mathbf{q}_0 \tag{3.2}$$

$$\sigma(1) \in \mathcal{C}^{\text{goal}}. \tag{3.3}$$

3.2.3 Energy-optimal planning problem

The energy-optimal planning problem is formulated in the literature as an optimal control problem, where the amount of energy dissipated in the actuators during the motion is expressed as the integral of squared torque. Let t be the time variable and $(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ the state of the robot at t, where $\mathbf{q}(t)$ and $\dot{\mathbf{q}}(t)$ denote the configuration and velocity vectors respectively. Let $\tau(t)$ be the torque vector controlling the robotic system at t. The control function is defined over the interval $[0, T]$ where the final time T is prescribed. The energy-optimal planning problem with zero velocities at end points is to find a feasible control $\tau(t)$, $0 \leq t \leq T$, to:

$$\begin{aligned}
\text{minimize} \quad &: \quad J := \int_0^T \tau(t)^\mathsf{T} \tau(t)\, \mathrm{d}t && (3.4)\\
\text{subject to} \quad &: \quad (\mathbf{q}(0), \dot{\mathbf{q}}(0)) = (\mathbf{q}_0, \mathbf{0}) && (3.5)\\
&\quad\;\; \mathbf{q}(T) \in \mathcal{C}^{\text{goal}} && (3.6)\\
&\quad\;\; \dot{\mathbf{q}}(T) = \mathbf{0} && (3.7)\\
&\quad\;\; \mathbf{q}(t) \in \mathcal{C}^{\text{free}}\ \forall t && (3.8)\\
&\quad\;\; \ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})^{-1}\{\tau - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{g}(\mathbf{q})\}, && (3.9)
\end{aligned}$$

or correctly report that such control does not exist. $\mathbf{M}(\mathbf{q})$ is the robot's generalized inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ denotes the Coriolis and centrifugal forces, and $\mathbf{g}(\mathbf{q})$ is the vector of gravity forces. The optimization problem in (3.4) is addressed in the following section.

3.3 Existing Solutions

3.3.1 Rapidly-exploring random tree

The RRT algorithm, introduced by LaValle (1998), builds a tree rooted at $\mathbf{q}_0$ that expands in the free configuration space. At each iteration, a sample from $\mathcal{C}$ is generated,

i.e. a point in the n-dimensional configuration space. The nearest point in the tree is *extended* toward this sample to create a new point $\mathbf{q}$. The extension procedure moves one point towards another by a prespecified step. The running time of RRT is $O(K \log K)$, where K is the number of samples (Karaman and Frazzoli 2011).

A grown RRT is shown in Figure 3.1. The configuration space is two dimensional and the robot used is the one in Figure 2.2.

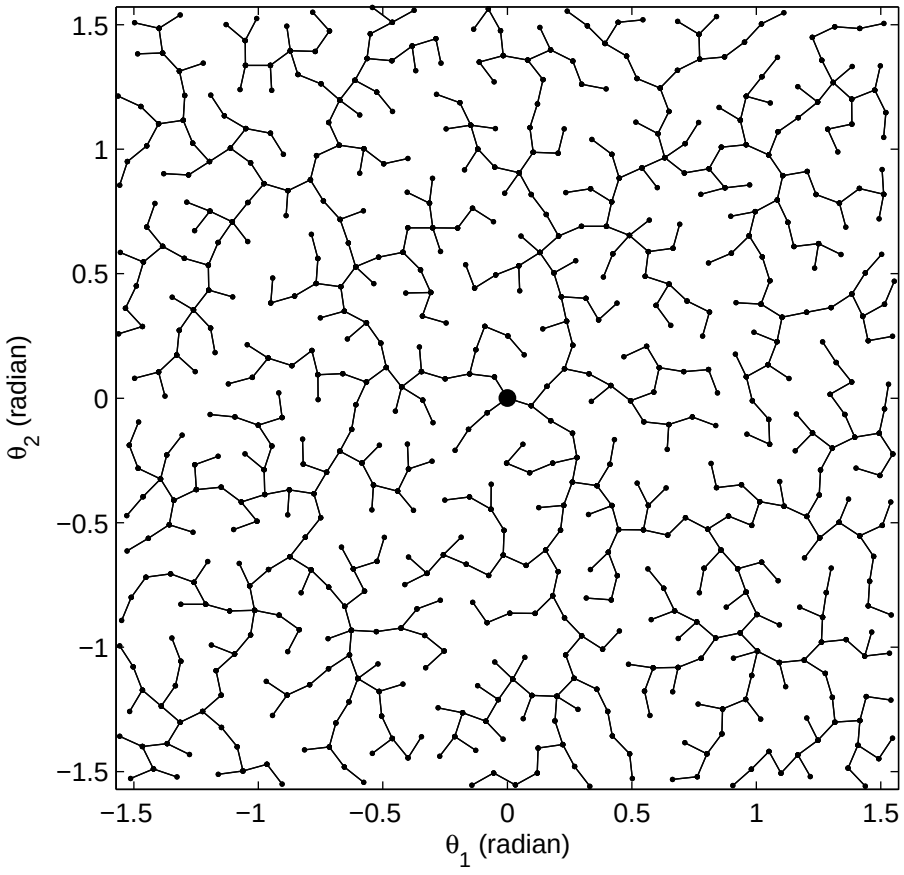


Figure 3.1: RRT in a two-dimensional configuration space applied on the robot in Figure 2.2.

3.3.2 Optimal rapidly-exploring random tree

The RRT* sampling-based motion planner is described by Karaman and Frazzoli (2010). Like RRT, the algorithm preserves a tree structure rooted at $\mathbf{q}_0$. At each iteration, the algorithm generates a sample from $\mathcal{C}$. The nearest point in the tree is extended toward this sample to create a new point $\mathbf{q}$. Points that lie inside a n-ball centered at $\mathbf{q}$ are examined. The point that minimizes the cumulative cost between $\mathbf{q}$ and the root of the tree is chosen to be the parent of $\mathbf{q}$. In addition, the points inside the n-ball are reconnected if the path through $\mathbf{q}$ has a smaller cost. The running time of RRT* is $O(K \log K)$, where K is the number of samples.

A grown RRT* is shown in Figure 3.2. The configuration space is two dimensional and the robot used is the one in Figure 2.2. The RRT* minimizes the Euclidean distance between configurations. To produce an accurate comparison with RRT, the same samples from Figure 3.1 are used here.

When implementing RRT*, a cost function must be specified. The Euclidean distance (i.e. the distance in the Euclidean configuration space) is often used as a performance criterion. However, for the energy-optimal planning problem, the Euclidean distance cannot accurately reflect the cost on a dynamical system such as the one in (3.9). To address the energy-optimal planning problem, we propose a new method to measure cost, described in the following section.

3.4 Integral-squared torque approximation

In this Section, a path σ is described as a finite sequence of configurations $\mathbf{q}_0, ..., \mathbf{q}_k$, where $\mathbf{q}_0$ is a vector denoting the initial configuration of the robot. To ensure continuity, we assume that the robot follows the line segments connecting the path points. A path is collision free if the line segments connecting the path points are inside $\mathcal{C}^{\text{free}}$.

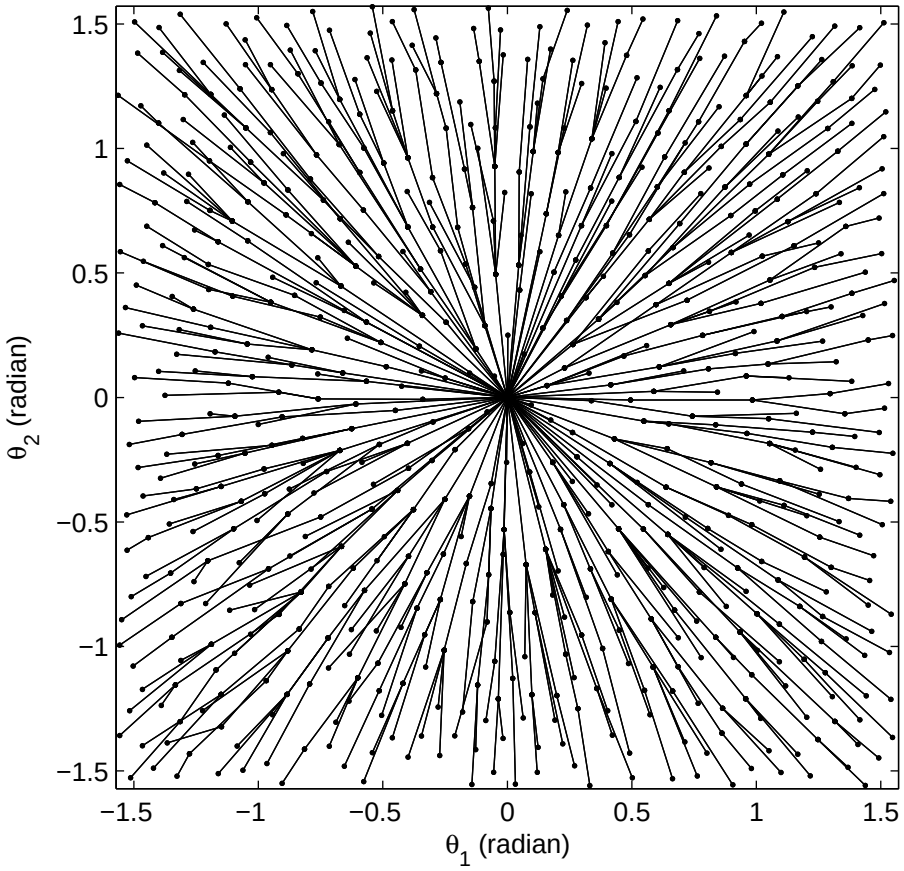


Figure 3.2: RRT* in a two-dimensional configuration space applied on the robot in Figure 2.2.

We assume that the robot uses a predefined control law (described in Section 3.5) that finds a control function given a path σ and a final time T.

Consider the path $\sigma = \{\mathbf{q}_0, ..., \mathbf{q}_k\}$, where $\mathbf{q}_k$ is a configuration to be added to an RRT*. To implement the algorithm in the context of energy-optimal planning, we must be able to assign an energy cost on the path σ (or any other path joining $\mathbf{q}_k$ to the root $\mathbf{q}_0$.) If $\mathbf{q}_k \notin \mathcal{C}^{\text{goal}}$, energy cannot be computed exactly because a final time T_k and a velocity $\dot{\mathbf{q}}_k$ are not specified by the problem definition for non-goal configurations. Even if T_k and $\dot{\mathbf{q}}_k$ can be determined (e.g. by carrying out a search in the state space), the exact calculation of (3.4) increases the time-complexity of the search problem, i.e. the running time of the motion planning algorithm RRT* is multiplied by a linear factor. The reason is that the cost will depend on a trajectory interpolating all points from $\mathbf{q}_0$ to $\mathbf{q}_k$, and such trajectory must be planned for each call to the cost function. This is true for any timing law having continuous acceleration at path points (Siciliano et al. 2009).

We present an approximation of (3.4) which we call *integral-squared torque approximation* (ISTA). The method aims at accomplishing the following:

- Effectively assigning energy costs to any paths in the configuration space.
- Maintaining a $n \log n$ running time for the planning algorithm, by providing a cost function that has an additive property. The additive property also plays a role in the proof of optimality of RRT* that relies on the assumption that the cost function is additive (Karaman and Frazzoli 2010). (i.e. the sum of the costs of two paths is equal to the cost of their concatenation.)

We denote by $\hat{c}_k$ the cost associated to the path $\{\mathbf{q}_0, ..., \mathbf{q}_k\}$. $\hat{c}_k$ is an additive cost given by

$$\hat{c}_k = \begin{cases} 0 & \text{if } k = 0 \\ \hat{c}_{k-1} + \hat{J}_k & \text{if } k > 0, \end{cases} \tag{3.10}$$

where $\hat{J}_k$ represents the (approximated) amount of energy needed to travel from $\mathbf{q}_{k-1}$ to $\mathbf{q}_k$. $\hat{J}_k$ is computed by executing the following steps:

1. A trajectory $\Pi : [0, t_f] \rightarrow \mathbb{R}^n$ is planned, Π interpolates the configurations $\mathbf{q}_{k-1}$ and $\mathbf{q}_k$ at 0 and t_f respectively, and describes a point-to-point motion. The trajectory is determined by finding a fifth order polynomial that satisfies the following constraints:

$$\Pi(0) = \mathbf{q}_{k-1} \tag{3.11}$$

$$\dot{\Pi}(0) = \mathbf{0} \tag{3.12}$$

$$\ddot{\Pi}(0) = \mathbf{0} \tag{3.13}$$

$$\Pi(t_f) = \mathbf{q}_k \tag{3.14}$$

$$\dot{\Pi}(t_f) = \mathbf{0} \tag{3.15}$$

$$\ddot{\Pi}(t_f) = \mathbf{0}. \tag{3.16}$$

2. The trajectory Π is discretized into m points ($m > 1$) evenly spaced in time from 0 to t_f, where t_f is given by $t_f = (m-1)\Delta t$, and Δt designates a small time interval between two consecutive points. m and Δt are parameters that must be determined prior to the motion planning. Our experience indicates that a small number of points suffices for obtaining good approximations. In our implementation, $m = 5$ and $\Delta t = 0.1$ second.

3. The cost of moving along Π is computed as follows:

$$\hat{J}_k = \sum_{i=0}^{m-2} \tau_k(t_i)^{\mathsf{T}} \tau_k(t_i)\, \Delta t, \tag{3.17}$$

where the torque vector $\tau_k(t_i)$ is determined by solving the inverse dynamics problem at $t_i = i\Delta t$.

3.5 Evaluation

The evaluation is made entirely in simulation, which allows us to obtain results from multiple trials, and an accurate reproduction of initial conditions. The robot is dynamically simulated using Featherstone's method for articulated bodies (Featherstone 1987). The simulated robot, shown in Figure 3.3, is an anthropomorphic manipulator with a fixed base: the arm possesses six degrees of freedom and the gripper two degrees of freedom.

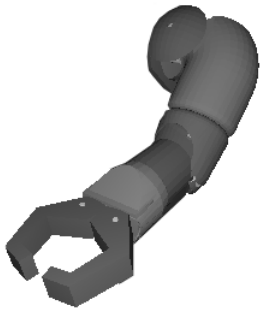


Figure 3.3: The simulated manipulator used in the evaluation.

The following control law is used in all simulations. To execute a path in the configuration space, the robotic arm is driven by torque computed by a composite inverse dynamics / PID controller, where the velocities and accelerations are obtained with a timing law of cubic splines.

3.5.1 Quality of the approximation

In this first evaluation, we are interested in measuring the success of ISTA in approximating the energy equation in (3.4). Consequently, we apply the following method.

A large number of paths of variable lengths are sampled from $\mathcal{S}$. A path is sampled as follows. An initial and final configuration are sampled from $\mathcal{C}^{\text{free}}$, then a RRT is used to connect the two configurations.

The paths are ranked using the energy equation in (3.4), which give us the true ranking. The paths are then ranked by the following metrics: ISTA and the Euclidean distance (ED). Using the Spearman rank correlation coefficient, we measure the strength of the associations between the true ranking and the rankings given by the two metrics. Values that are close to 1 indicate a strong association, whereas values that are close to 0 indicate a weak association. We use this measure to quantify the quality of the approximation for ISTA and ED.

This evaluation method is repeated and the Spearman rank correlation coefficient is recalculated for different average speed of the robotic arm. We modify the average speed by rescaling the trajectory, i.e. by modifying the final time T. The average speed is increased until a feasible actuation to follow a path cannot be found. We say that we reached an inadmissible velocity region.

The results are obtained from 5000 paths and are shown in Figure 3.4. For an average speed between 0.2 and 1.8 radian per second, we observe that the correlation is close to 1 for ISTA and varies between 0.4 and 0.8 for ED. For an average speed larger than 1.8 the correlation for ISTA and ED consistently decrease, which seems to suggest that accurate energy prediction is harder to achieve in high-speed motions. We conclude that ISTA can effectively provide an approximation of the amount of energy dissipated in the actuators during the motion of a robotic arm.

3.5.2 Motion planning

In this section, we present a comparative evaluation of ISTA. Two motion planners that both use the RRT* algorithm are compared. The first planner is denoted by τ-RRT* and uses ISTA to calculate cost. The cost function in the second planner, denoted by d-RRT*, is the Euclidean distance. The evaluation is performed in the environment in Figure 3.5. We quantify the effectiveness of the motion planners as follows.

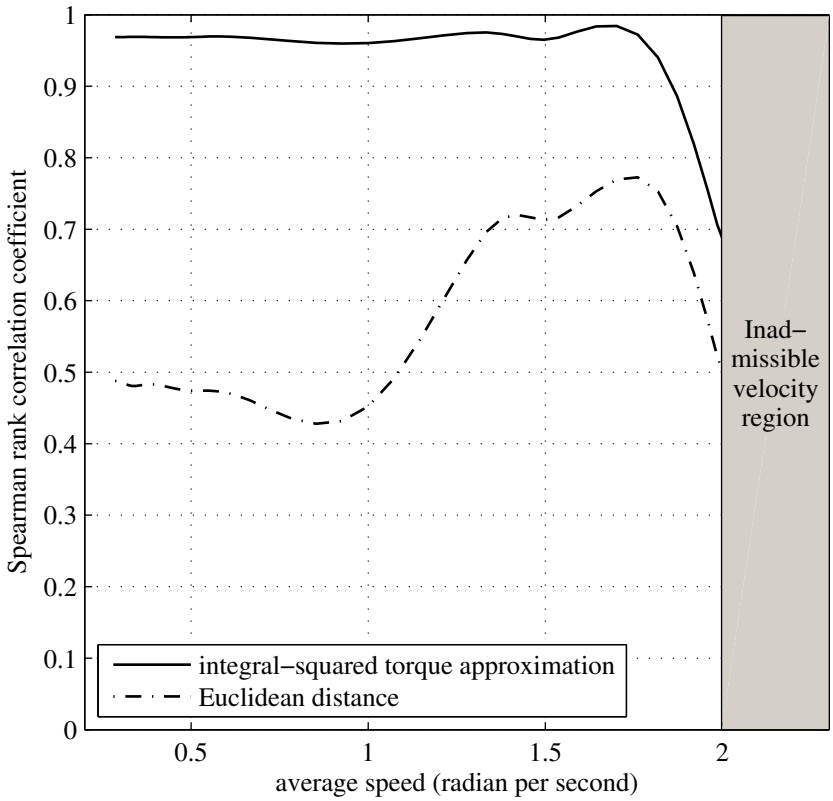


Figure 3.4: The Spearman rank correlation coefficient is used as a measure of success of the integral-squared torque approximation and the Euclidean distance to approximate the energy equation in (3.4).

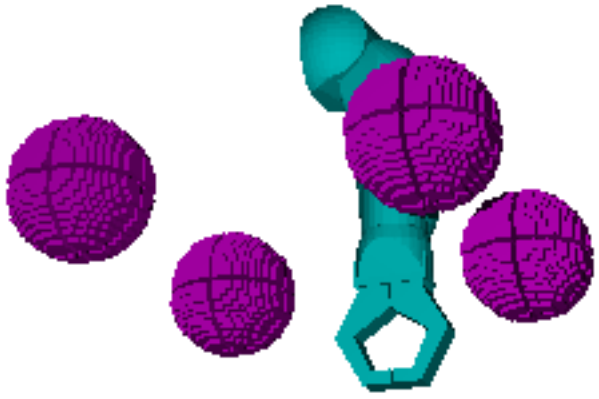


Figure 3.5: The simulated environment has four spheres (obstacles) surrounding the arm.

The evaluation is made on a large number of instances of the motion planning planning problem. The dimension of the joint space is $n = 6$ (we fix the gripper). For each instance, an initial and a goal configuration are sampled from $\mathcal{C}^{\text{free}}$. The planners τ-RRT* and d-RRT* are then applied on the samples to obtain a path in the configuration space. The quality of the path returned by each planning algorithm, after 3000 iterations, is quantified using the energy equation in (3.4). The results are shown in Table 3.1. Each value is an average obtained from 100 instances. Table 3.1 shows results for different average speed of the arm. The average speed is modified by changing the final time T. We observe that the average amount of energy expended by the robot is overall roughly 15% lower when τ-RRT* is used. Figure 3.6 shows that the running time of τ-RRT* slightly increases when compared to the running time of d-RRT*.

Table 3.1: Average amount of energy expended for the motion planning algorithms τ-RRT* and d-RRT*.

Average Speed (radian per second)	τ-RRT* (unit energy)	d-RRT* (unit energy)
0.5	148	169
0.6	118	135
0.8	88	100
1.0	73	83
1.2	58	66
1.6	44	50

We illustrate the comparative evaluation obtained in this section by providing the following two examples.

Moving around an obstacle

The purpose in this example is to show that by increasing the number of iterations of RRT*, the quality of the solution found can deteriorate when an inappropriate cost function is used. In this example the RRT* is expanded only in the shoulder joint space of the robot, placed in the environment in Figure 3.5 (the shoulder has two degrees of

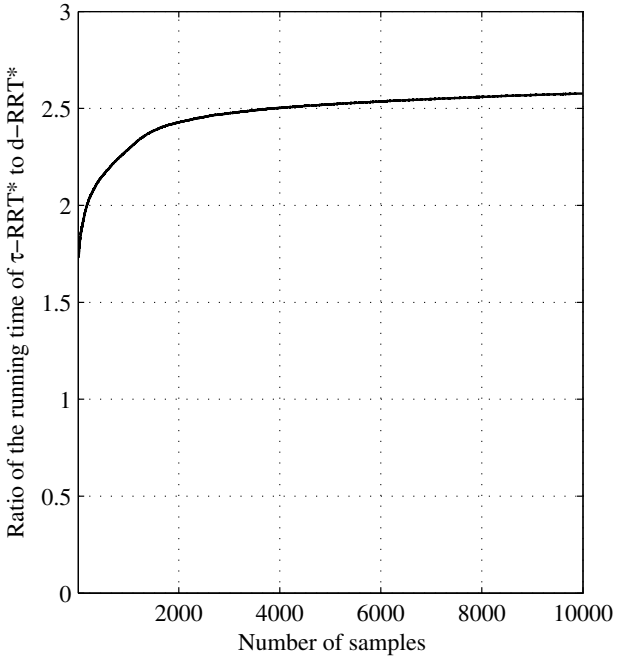


Figure 3.6: The ratio of the running time of τ-RRT* to d-RRT* is shown for an increasing number of samples (i.e. number of iterations).

freedom). The robot has to find its way around a sphere. τ-RRT* and d-RRT* are applied on the same initial and final conditions. The results are reported in Figure 3.7 after 1000 and 5000 iterations of the algorithms.

We observe that τ-RRT* finds the best solution; the path found is identical at 1000 and 5000 iterations. On the other hand, the quality of the path returned by d-RRT* worsen as the number of iterations increases. Note that the path found by d-RRT* at 5000 iterations has the shortest length.

Lowering the arm

In this second example, we show that the length of a path in the joint space and the amount of energy expended during the motion can be, in some cases, inversely proportional. The robot arm is placed in an environment without obstacles, it is initially

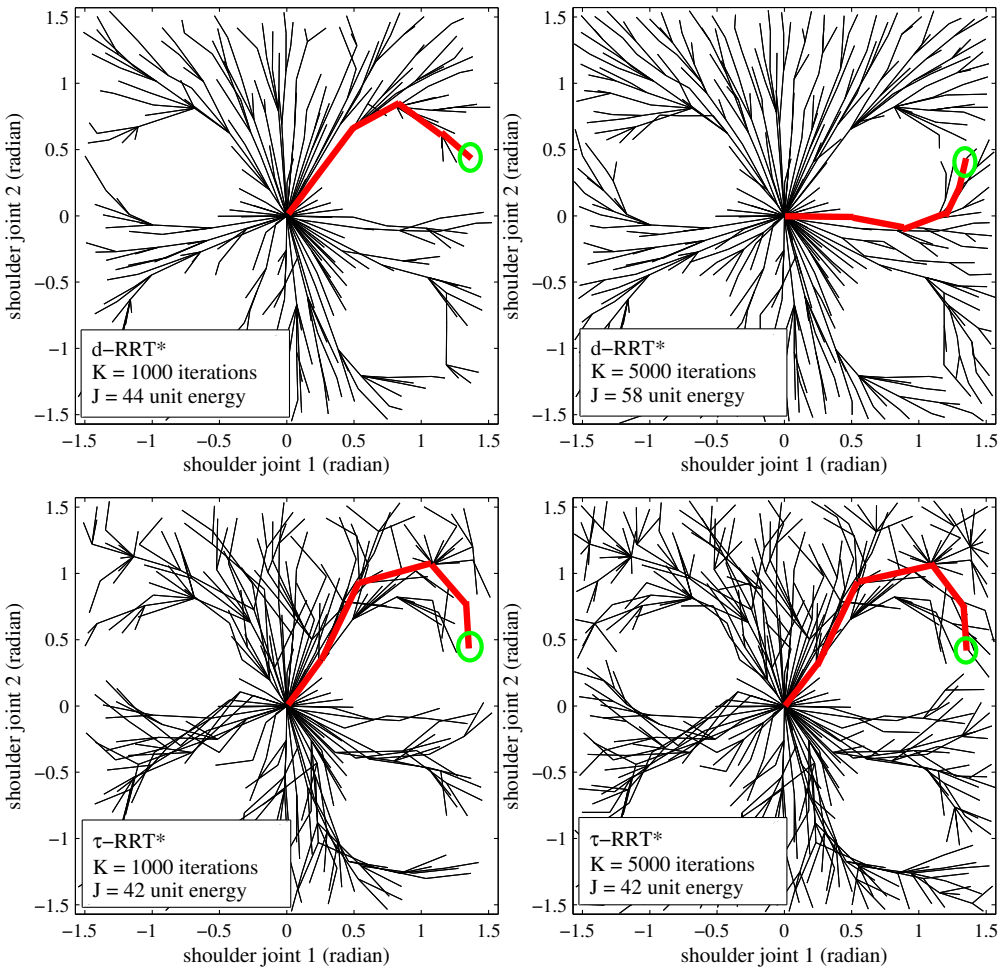


Figure 3.7: Expansion of d-RRT* (top) and τ-RRT* (bottom) in the shoulder joint space of the robot arm in an environment with obstacles (Figure 3.5). The tree is shown after 1000 iterations (left) and 5000 iterations (right). The goal region is represented by a green circle and the path to the goal by bold red lines. The amount of energy to move the robot along each path, calculated via (3.4), is displayed.

raised and instructed to be lowered. τ-RRT* and d-RRT* are expanded only in the (first) shoulder and elbow joint space. The results are reported in Figures 3.8 and 3.9.

The path found by d-RRT* shows little change in the position of the elbow joint, whereas the path found by τ-RRT* describes a curve that causes the elbow joint to bend. The results show that, by flexing the arm, the length of the path increases but the amount of energy is considerably reduced.

3.6 Conclusion

This chapter provides the method integral-squared torque approximation (ISTA) that assigns a nonnegative quantity to a path in the configuration space. This quantity approximates the amount of energy expended to move the robot on the path. ISTA is integrated with the motion planner RRT* to find low-energy collision-free paths in the robot's configuration space.

Section 3.5 presents a method to quantify the quality of an approximation of energy by a metric. ISTA and the Euclidean distance are used to rank a large number of paths sampled from $\mathcal{S}$. The results indicate a strong association between ISTA's ranking and the true ranking (that uses the energy equation) of the paths. The main results in this chapter show that ISTA is able to improve the solution obtained by the planner RRT* when compared to the Euclidean distance, by consistently reducing energy.

ISTA can be integrated with other motion planners, e.g. the probabilistic roadmap; particularly in the query phase, to find low-energy paths inside the roadmap represented by a weighted graph. ISTA can also be utilized to assign weights in other graph-based robotics problems, e.g. coverage and exploration. These two robotics problem are studied in the next two chapters.

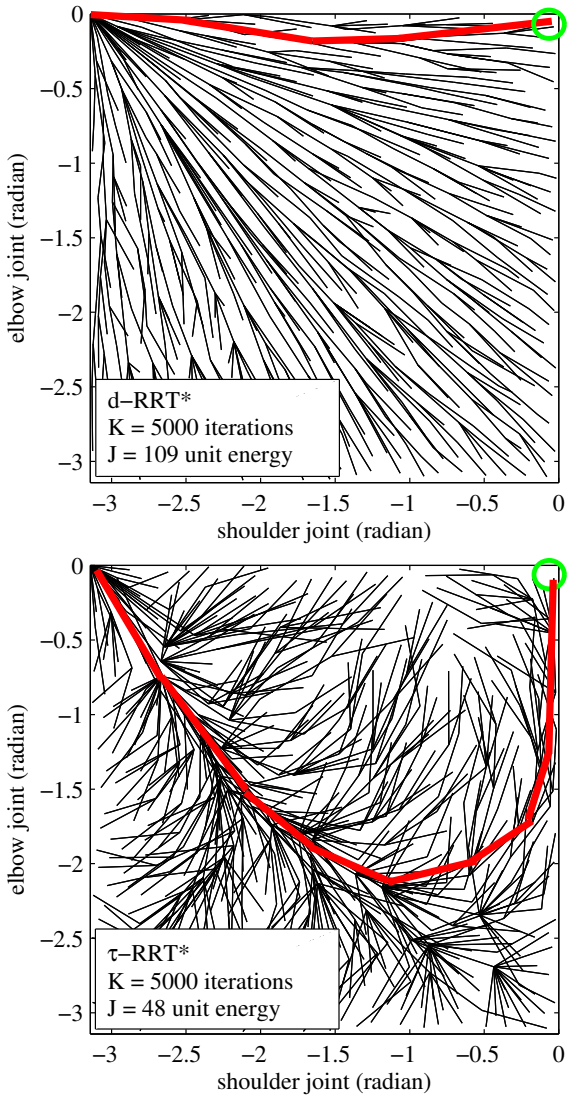


Figure 3.8: Expansion of d-RRT* (top) and τ-RRT* (bottom) in the shoulder and elbow joint space of the robot arm. The tree is shown after 5000 iterations. The goal region is represented by a green circle and the path to the goal by bold red lines. The amount of energy to move the robot along each path, calculated via (3.4), is displayed.

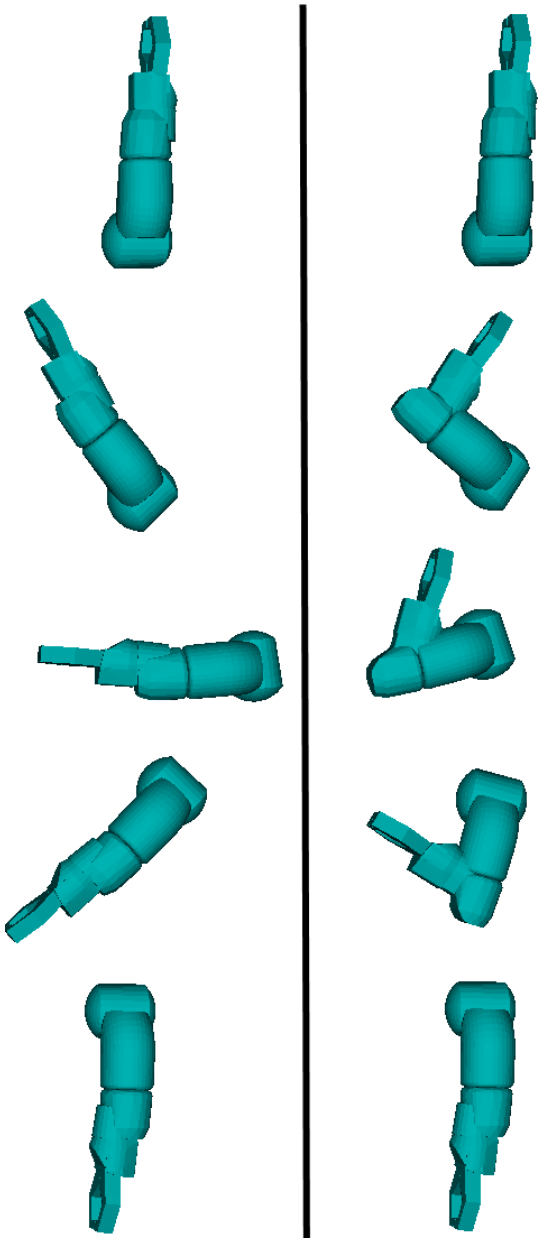


Figure 3.9: Motion of the arm resulting from the paths (in Figure 3.8) found by d-RRT* (left) and τ-RRT* (right).

Chapter 4

Generalized Coverage Problem

4.1 Introduction

This chapter studies the computational complexity of the coverage problem, and proposes an algorithm that has the property of being probabilistically complete, and aims at reducing a cost function associated with the exploration path. The robotics coverage problem has important applications that include de-mining, surveillance, search-and-rescue, cleaning, surface painting etc. Informally, the problem consists of finding a continuous motion that results in the coverage of a region in the robot's workspace. Depending on the application, the coverage can be done using tools mounted on the robot (e.g. de-mining equipment) or range sensors.

Because of its wide range of applications, the problem does not have a unified formulation in the literature. In this chapter, we seek to propose a unified formulation that encompass a large majority of coverage problems and can be applied to different types of robots and environments (i.e. two and three-dimensional environments). The resulting problem is then called generalized coverage problem (GCP).

This chapter is structured as follows. Section 4.2 presents an overview of related work. Section 4.3 provides preliminary materials. GCP is defined in Section 4.4, its

polynomial-space hardness is shown, and a probabilistically complete algorithm is described. In Section 4.5, GCP is recasted as an optimization problem, and a new algorithm that minimizes the cost associated with covering a region is provided.

4.2 Related work

4.2.1 Motion planning

Coverage is closely related to motion planning. In both problems, the robot must move through a path in its configuration space while being subjected to physical constraints (e.g. obstacles in the environment). In robotics, the basic motion planning problem is to find a collision-free path in the robot's configuration space, from an initial configuration to a goal region (or correctly report that such path does not exist.)

Even by this most basic definition, the motion planning problem, also known as the *piano mover's problem* (LaValle 2006), is shown to be inherently difficult to solve (Reif 1979). Computationally efficient, probabilistically complete or resolution complete sampling-based planners are generally successful at solving the problem. Such algorithms include the probabilistic roadmaps (PRM) and the rapidly-exploring random trees (RRT) (Kavraki et al. 1996; LaValle 1998). They are discussed in Chapter 3. The coverage algorithms in this chapter uses two existing sampling-based planners: the rapidly-exploring random trees (RRT) and the optimal rapidly-exploring random trees (RRT*).

4.2.2 Coverage in a two-dimensional workspace

Early approaches in robotic coverage consider mobile robots in two-dimensional environments. A spanning-tree-based algorithm is proposed by Gabriely and Rimon (2001). The algorithm assumes that the environment is subdivided into cells. The work by

Choset (2001) provides an overview of other early coverage algorithms used in a two-dimensional workspace. The survey classifies coverage algorithms depending on the cellular decomposition used (i.e. heuristic, approximate or exact cellular decomposition).

Two-dimensional coverage is applied on a variety of applications mentioned here. A coverage algorithm developed by Acar et al. (2003) guides a robot in a minefield to detect and possibly clear mines. Atkar et al. (2005) applies the problem of coverage to the spray painting of simple surfaces. A coverage method that uses areal unmanned vehicles is provided by Xu et al. (2011). Their method decomposes the workspace into free-space regions and computes the shortest path that visits each region. Surveillance is yet another application that requires a robot to cover its environment for detecting intrusion or searching for a person or an object of interest. An algorithm that can be used by a team of robots to patrol an urban area is designed by Xu and Stentz (2011).

The methods cited above are designed toward robots operating in a planar space, and are thus not necessarily practical for a three-dimensional workspace.
Three-dimensional coverage is examined below.

4.2.3 Coverage in a three-dimensional workspace

Three-dimensional coverage can be applied in the construction of a model of an object or a scene. In the literature, it is sometimes known under the name *view planning problem* (VPP). The objective is to create a three-dimensional model of an object using a minimum number of *viewpoints*, i.e. sensors' position and orientation. A theoretical framework to VPP is provided and its similarity to the known set cover problem is shown (Scott et al. 2001). Wang et al. (2007) propose to minimize the traveling cost, i.e. the total (Euclidean) distance traveled by the robot. The problem is then called *traveling VPP*. They provide an integer programming formulation to traveling VPP, and an LP rounding algorithm. Their approach assumes that a discrete number of viewpoints is known a priori.

Other three-dimensional coverage methods include the work by Breitenmoser et al. (2010). They propose an approach to cover a nonplanar surface by dividing it into Voronoi regions. Their approach can be applied on a group of robots performing simultaneous coverage. The coverage of a three-dimensional urban environment is studied by Cheng et al. (2008). The robot used is an unmanned areal vehicle with a conical-field-of-view sensor attached on its underside, and the environment has a cylindrical shape.

More recent work uses manipulator robots to cover three-dimensional surfaces (Hess et al. 2012). They transform an instance of the coverage problem to an instance of the generalized traveling salesman problem. The solution found is a path that minimizes a cost function. Their approach provides collision-free paths over the covered surface only (i.e., for the end effector), and not for the entire robot. Three-dimensional coverage is accomplished in a collision-free manner by (Englot and Hover 2012; Hover et al. 2012). Their method employs sampling-based motion planners to find a path in the robot's configuration space that results in the coverage of a complex surface such as the hull of a ship. Subsequently, the planner RRT* is used to locally optimize the path found. In contrast, our method aims at globally reducing the cost of the solution.

4.3 Preliminaries

4.3.1 Notation

Let $\mathcal{W}$ be a bounded region in a three-dimensional space that represents the robot's physical world. Let $\mathcal{C}$ be the n-dimensional configuration space of the robot, and $\mathbf{q}$ a configuration in $\mathcal{C}$.

Let $\mathcal{W}^{\text{obs}}$ denote the portion of $\mathcal{W}$ occupied by obstacles. The physical space occupied by the robot at $\mathbf{q}$ is denoted by $\mathcal{R}(\mathbf{q})$, and $\mathcal{C}^{\text{free}}$ denotes the free configuration space. $\mathbf{q} \in \mathcal{C}^{\text{free}}$ if and only if $\mathcal{R}(\mathbf{q})$ does not intersect with $\mathcal{W}^{\text{obs}}$.

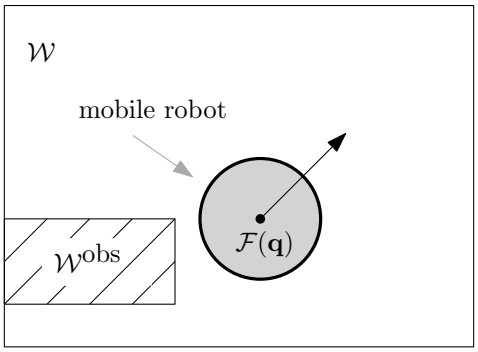


Figure 4.1: Mobile robot in a planar world $\mathcal{W}$. The region in gray corresponds to $\mathcal{F}(\mathbf{q})$ and coincides with the physical space occupied by the robot at configuration $\mathbf{q}$. The striped region corresponds to the obstacle space $\mathcal{W}^{\text{obs}}$.

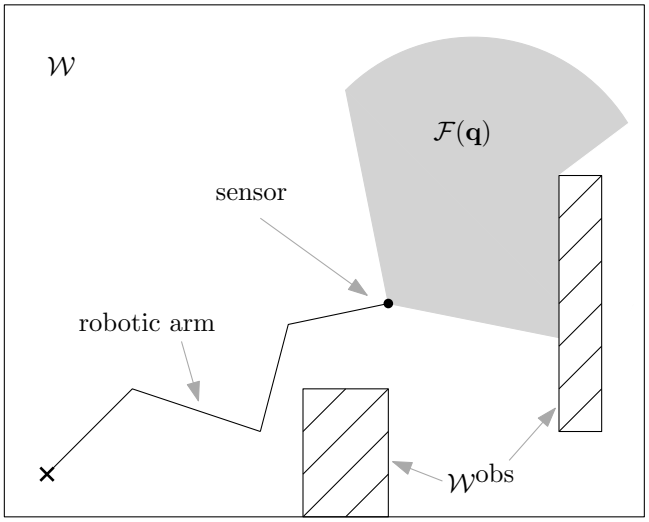


Figure 4.2: Robotic arm equipped with a sensor in a planar world $\mathcal{W}$. The sensor is placed at the end effector. The region in gray corresponds to $\mathcal{F}(\mathbf{q})$ and coincides with the robot's field of view at configuration $\mathbf{q}$. The striped region corresponds to the obstacle space $\mathcal{W}^{\text{obs}}$.

The power set of $\mathcal{W}$ is the set of all subsets of $\mathcal{W}$ and is denoted $\mathcal{P}(\mathcal{W})$. Let $\mathcal{F} : \mathcal{C} \to \mathcal{P}(\mathcal{W})$ be a function that uniquely associates a configuration to a subset of $\mathcal{W}$. The definition of $\mathcal{F}$ is illustrated by the following examples in Figures 4.1 and 4.2, where $\mathcal{W}$ is a planar world. In the first example (Figure 4.1) $\mathcal{F}(\mathbf{q})$ corresponds to the the physical space occupied by a mobile robot. In the second example (Figure 4.2) $\mathcal{F}(\mathbf{q})$ corresponds to the field of view of a robot arm. Note that in the first example we have $\mathcal{F}(\mathbf{q}) \equiv \mathcal{R}(\mathbf{q})$. In both examples $\mathcal{F}(\mathbf{q})$ is a subset of $\mathcal{W}$.

4.3.2 Definition of coverage

Let $\mathcal{W}'$ be a subset of $\mathcal{W}$ and let the function $\sigma : [0,1] \to \mathcal{C}^{\text{free}}$ be a continuous collision-free path. A robot moving along σ is said to *cover* $\mathcal{W}'$ (or more simply σ covers $\mathcal{W}'$) if and only if

$$\mathcal{W}' \subseteq \bigcup_{\mathbf{q} \in \sigma} \mathcal{F}(\mathbf{q}). \tag{4.1}$$

Equation (4.1) states that $\mathcal{W}'$ is a subset of the union of all $\mathcal{F}(\mathbf{q})$ on the path σ. When the robot reaches the last configuration $\sigma(1)$ on the path, the covered region $\bigcup_{\mathbf{q} \in \sigma} \mathcal{F}(\mathbf{q})$ contains $\mathcal{W}'$ if (4.1) is true.

4.4 Generalized coverage problem

4.4.1 Problem formulation

The generalized coverage problem (GCP) is defined as follows. Given an initial configuration $\mathbf{q}^{\text{init}}$, a goal region $\mathcal{W}^{\text{goal}} \subseteq \mathcal{W}$, and a function $\mathcal{F}$ that associates a configuration to a subset of $\mathcal{W}$, find a continuous collision-free path σ such that

$$\sigma(0) = \mathbf{q}^{\text{init}} \tag{4.2}$$

and

$$\mathcal{W}^{\text{goal}} \subseteq \bigcup_{\mathbf{q} \in \sigma} \mathcal{F}(\mathbf{q}), \tag{4.3}$$

or correctly report that such path does not exist. A solution to GCP is a path that starts at $\mathbf{q}^{\text{init}}$ and drives the robot to cover $\mathcal{W}^{\text{goal}}$.

The problems cited in Section 4.2 are therefore subsumed under GCP:

- The workspace $\mathcal{W}$ is three dimensional and two-dimensional coverage is a special case of GCP.

- $\mathbf{q}$ can fully specify the position of a robot whether it is a manipulator, a mobile robot or an unmanned areal vehicle.

- The definition of $\mathcal{F}$ is general enough to include different types of sensors or tools mounted on the robot.

4.4.2 Computational complexity

In this section, we show that GCP is unlikely to have a solution that runs in polynomial time.

Theorem 1. *The generalized coverage problem is polynomial-space hard.*

Proof. We prove that GCP is polynomial-space hard by showing that the motion planning problem (MPP) is polynomial-time reducible to GCP. MPP is a well-studied problem in robotics, it is also known under the name generalized mover's problem. MPP is known to be polynomial-space complete (Reif 1979; Canny 1988).

Recall that in MPP, we are given an initial and a goal configuration in $\mathcal{C}$ respectively denoted $\mathbf{p}^{\text{init}}$ and $\mathbf{p}^{\text{goal}}$. The problem is to find a continuous collision-free path σ such that $\sigma(0) = \mathbf{p}^{\text{init}}$ and $\sigma(1) = \mathbf{p}^{\text{goal}}$, where $\sigma(0)$ and $\sigma(1)$ correspond to the initial and final configurations on the path respectively.

A reduction algorithm from MPP to GCP takes the input of MPP $\mathbf{p}^{\text{init}}$ and $\mathbf{p}^{\text{goal}}$ and constructs an instance of GCP as follows.

$$\mathbf{q}^{\text{init}} \leftarrow \mathbf{p}^{\text{init}}, \tag{4.4}$$

$$\mathcal{W}^{\text{goal}} \leftarrow \mathcal{W}', \tag{4.5}$$

$$\mathcal{F}(\mathbf{q}) \leftarrow \begin{cases} \mathcal{W}' & \text{if } \mathbf{q} = \mathbf{p}^{\text{goal}} \\ \emptyset & \text{otherwise,} \end{cases} \tag{4.6}$$

where $\mathcal{W}'$ is an arbitrary subset of $\mathcal{W}$.

By assigning all the values taken by $\mathcal{F}$ not corresponding to $\mathbf{p}^{\text{goal}}$ to the empty set, the only way to cover $\mathcal{W}^{\text{goal}}$ is to reach the configuration $\mathbf{p}^{\text{goal}}$ exactly. Therefore, a solution to the instance of GCP constructed from (4.4), (4.5) and (4.6) is a path σ such that $\sigma(0) = \mathbf{p}^{\text{init}}$ and $\sigma(1) = \mathbf{p}^{\text{goal}}$. □

When compared to the coverage problem, the exploration problem appears to be more challenging: In the exploration problem, the robot has no knowledge at all, or only partial knowledge about the environment to be explored and must increase its knowledge during the exploration. Therefore, we make the following conjecture.

Conjecture 2. *The exploration problem is polynomial-space hard.*

4.4.3 Proposed solution

The following algorithm makes use of sampling-based motion planners and seeks to find a path that covers the goal region. Algorithm 1 starts by constructing a rapidly-exploring random tree (RRT) denoted $\mathbf{T}$. The RRT algorithm takes two parameters: the root $\mathbf{q}^{\text{init}}$ and the number of samples K. RRT incrementally expands a tree structure in $\mathcal{C}^{\text{free}}$, rooted at $\mathbf{q}^{\text{init}}$, in the collision-free space. Note that K is also a parameter to Algorithm 1.

Algorithm 1 An approach to the generalized coverage problem.

```
1: T ← RRT(q^init, K)
2: if W^goal ⊆ ⋃_{q∈T} F(q) then
3:     σ ← preorder traversal of T
4:     return σ
5: else
6:     return "failure"
7: end if
```

Algorithm 1 then verifies that $\mathbf{T}$ covers the goal region $\mathcal{W}^{\text{goal}}$. If this condition is true, Algorithm 1 returns a path σ that traverses the tree entirely, starting at $\mathbf{q}^{\text{init}}$. If not, the algorithm reports failure.

To ensure continuity, we assume that the path σ contains the line segments connecting vertices.

4.4.4 Completeness Analysis

In this section, preliminary definitions are given, then followed by the analysis of Algorithm 1.

Definitions

Let $\mathbf{q}_1$ and $\mathbf{q}_2$ be two configurations in $\mathcal{C}^{\text{free}}$. $\mathbf{q}_1$ is *reachable* from $\mathbf{q}_2$ if and only if they can be connected by a continuous collision-free path. This property is denoted by $\mathbf{q}_1 \sim \mathbf{q}_2$.

A solution σ to GCP is *correct* if and only if all the following conditions hold.

1. σ is continuous.
2. σ is collision-free.
3. Equation (4.2) is satisfied.
4. Equation (4.3) is satisfied.

An algorithm ALG(K), where K designates the number of samples used, is *probabilistically complete* if and only if the following proposition is true. If a solution exists, the probability that ALG(K) returns a correct solution approaches unity as K approaches infinity.

Probabilistic completeness

This section demonstrates the probabilistic completeness of Algorithm 1. The following assumption is made: The workspace $\mathcal{W}$ is represented by a *map*, i.e. a *finite* list

of objects in the environment (Thrun et al. 2005). For example, $\mathcal{W}$ is represented by a grid that partitions the workspace into a finite number of grid cells.

Lemma 3. *A path σ returned by Algorithm 1 is correct.*

Proof. To prove that σ is correct, all four conditions provided by the definition of correctness must hold. By Algorithm 1, σ is a preorder traversal of a RRT denoted $\mathbf{T}$ that begins at $\mathbf{q}^{\text{init}}$, thus Equation (4.2) is satisfied. Because a RRT maintains a tree structure, it is connected, therefore assuming that σ contains the line segments connecting the vertices of $\mathbf{T}$, σ is continuous. Moreover, a RRT only expands in $\mathcal{C}^{\text{free}}$: all the vertices and edges of $\mathbf{T}$ are in $\mathcal{C}^{\text{free}}$. Hence σ is collision free. Finally, σ contains all the configurations in $\mathbf{T}$ and σ is computed only if $\mathbf{T}$ covers $\mathcal{W}^{\text{goal}}$, therefore Equation (4.3) is satisfied, which completes the proof. □

Lemma 4. *If a solution to GCP exists, then there exists a finite and nonempty set A such that A covers $\mathcal{W}^{goal}$ and $\forall \mathbf{q} \in A, \mathbf{q} \sim \mathbf{q}^{init}$.*

Proof. If a solution to GCP exists, then we have a nonempty set

$$\mathcal{X} = \{\mathbf{q} \in \mathcal{C}^{\text{free}} \mid \mathbf{q} \sim \mathbf{q}^{\text{init}} \wedge \mathcal{F}(\mathbf{q}) \text{ intersects } \mathcal{W}^{\text{goal}}\} \tag{4.7}$$

that covers $\mathcal{W}^{\text{goal}}$. A subset A of $\mathcal{X}$ is constructed by the following procedure. At each step a configuration $\mathbf{q}'$ is selected from $\mathcal{X}$ such that

$$\{\mathcal{F}(\mathbf{q}') \setminus \bigcup_{\mathbf{q} \in A} \mathcal{F}(\mathbf{q})\} \text{ intersects } \mathcal{W}^{\text{goal}}. \tag{4.8}$$

The procedure halts when A covers $\mathcal{W}^{\text{goal}}$. Since $\mathcal{W}^{\text{goal}}$ is represented by a finite list of objects, there are finitely many steps in the procedure, and A is finite. □

Theorem 5. *If a solution to the generalized coverage problem exists, the probability that Algorithm 1 returns a correct path converges to 1 as the number of samples used approaches infinity.*

Proof. By Lemma 4, it is sufficient for a path σ to reach all configurations in A to solve GCP. Since the RRT algorithm is probabilistically complete and all configurations in A are reachable from $\mathbf{q}^{\text{init}}$, the probability that $\mathbf{T}$ reaches all configurations in A approaches one as the number of samples K approaches infinity. It is also shown that, if a solution exists, the probability that RRT fails to find one decreases exponentially with the number of samples (Frazzoli et al. 2002; LaValle and Kuffner 1999). By Lemma 3, the returned path is correct and the theorem follows. □

4.5 Optimal coverage

In practice it is necessary to reduce the *cost* associated with covering a goal region. The cost is a quantity that might represent the amount of energy expended by the robot, or the time taken to complete the coverage.

First, we recast GCP as an optimization problem, then we address it by proposing a new algorithm.

4.5.1 Problem formulation

Let $\mathcal{S}$ denote the set of all continuous collision-free paths and let $c : \mathcal{S} \to \mathbb{R}^+$ be a function that assigns a nonnegative quantity to a path in $\mathcal{S}$.

Given an initial configuration $\mathbf{q}^{\text{init}}$, a goal region $\mathcal{W}^{\text{goal}} \subseteq \mathcal{W}$, and a function $\mathcal{F}$ that associates a configuration to a subset of $\mathcal{W}$, the problem is to find a continuous collision-free path σ to

$$\text{minimize} \quad : \quad c(\sigma) \tag{4.9}$$

$$\text{subject to} \quad : \quad \sigma(0) = \mathbf{q}^{\text{init}} \tag{4.10}$$

$$\mathcal{W}^{\text{goal}} \subseteq \bigcup_{\mathbf{q} \in \sigma} \mathcal{F}(\mathbf{q}). \tag{4.11}$$

4.5.2 Proposed solution

A solution to the optimization problem in (4.9) is described in Algorithm 2. The algorithm preserves the probabilistic completeness of Algorithm 1 and improves the quality of the returned path.

Algorithm 2 An approach to the problem in (4.9).

1: $\mathbf{T} \leftarrow \text{RRT}(\mathbf{q}^{\text{init}}, K)$
2: **if** $\mathcal{W}^{\text{goal}} \subseteq \bigcup_{\mathbf{q} \in \mathbf{T}} \mathcal{F}(\mathbf{q})$ **then**
3: $A \leftarrow$ minimum-size subset of $\mathbf{T}$ covering $\mathcal{W}^{\text{goal}}$ (Algorithm 3)
4: $\sigma^* \leftarrow$ minimum-cost continuous collision-free path starting at $\mathbf{q}^{\text{init}}$ and traversing all configurations in A (Algorithm 4)
5: **return** σ^*
6: **else**
7: **return** "failure"
8: **end if**

Algorithm 2 starts by constructing a RRT rooted at $\mathbf{q}^{\text{init}}$, then verifies that $\mathbf{T}$ covers the goal region $\mathcal{W}^{\text{goal}}$. If this condition is true, the algorithm finds a minimum-size subset A of $\mathbf{T}$ that also covers $\mathcal{W}^{\text{goal}}$. This is described in Algorithm 3. Then, Algorithm 2 finds a path that connects all the configurations in A (starting at $\mathbf{q}^{\text{init}}$) while minimizing the cost function c. This is described in Algorithm 4. If the initial condition is false, the algorithm reports failure.

Minimum-size set covering the goal region

We now describe an approximate solution the problem of finding a minimum-size set A that covers $\mathcal{W}^{\text{goal}}$ given $\mathbf{T}$. This problem is very similar to the NP-hard discrete optimization problem: the set cover problem (SCP). A simple greedy approximation algorithm produces a solution to SCP whose value is within a factor of $O(\log n)$ of the value of the optimal solution, and remains the current best algorithm (Vazirani 2001). Other research also use approximation algorithms to SCP in the context of coverage and object inspection (Danner and Kavraki 2000; Scott et al. 2001; Englot and Hover 2011).

Algorithm 3 is an adaptation of the greedy solution for SCP. Initially, the set A consists of $\mathbf{q}^{\text{init}}$, and the region $\mathcal{F}(\mathbf{q}^{\text{init}})$ is removed from $\mathcal{W}^{\text{goal}}$. At each iteration, the algorithm chooses the configuration $\mathbf{q}$ in $\mathbf{T}$ that corresponds to the largest uncovered region. The size of the corresponding uncovered region is given by $|\mathcal{F}(\mathbf{q}) \cap \mathcal{W}^{\text{goal}}|$. Then, $\mathbf{q}$ is added to A and the region $\mathcal{F}(\mathbf{q})$ is removed from $\mathcal{W}^{\text{goal}}$. The algorithm terminates when $\mathcal{W}^{\text{goal}}$ is empty.

Algorithm 3 An adaptation of the greedy algorithm for SCP that finds a subset of $\mathbf{T}$ covering $\mathcal{W}^{\text{goal}}$.

1: $A \leftarrow \{\mathbf{q}^{\text{init}}\}$
2: $\mathcal{W}^{\text{goal}} \leftarrow \mathcal{W}^{\text{goal}} \setminus \mathcal{F}(\mathbf{q}^{\text{init}})$
3: **while** $\mathcal{W}^{\text{goal}} \neq \emptyset$ **do**
4: $\quad \mathbf{q} \leftarrow$ configuration in $\mathbf{T}$ that maximizes $|\mathcal{F}(\mathbf{q}) \cap \mathcal{W}^{\text{goal}}|$
5: $\quad A \leftarrow A \cup \{\mathbf{q}\}$
6: $\quad \mathcal{W}^{\text{goal}} \leftarrow \mathcal{W}^{\text{goal}} \setminus \mathcal{F}(\mathbf{q})$
7: **end while**
8: **return** A

Minimum-cost path

Algorithm 4 is an approximate solution to the following problem. Given a set $A = \{\mathbf{q}_1, ..., \mathbf{q}_{|A|}\}$, a cost function c and an initial configuration $\mathbf{q}^{\text{init}} \in A$, find a continuous collision-free path σ^* that visits each configuration in $|A|$ once, starting at $\mathbf{q}^{\text{init}}$. Figure 4.3 gives an example where the configuration space is two-dimensional. The configurations that the robot must visit (i.e. the elements of A) are highlighted. The figure also shows that all configurations are inter-reachable since they all belong to a RRT (provided by Algorithm 2.) The proposed solution works as follows. First, an instance of the traveling salesman problem (TSP) is constructed, then, an approximation algorithm to TSP is used to find σ^*.

The problem at hand is a combination of the multi-query MPP and TSP. The RRT algorithm is a probabilistically complete solution for MPP, however it is shown that RRT converges to a non-optimal solution with respect to a cost function (Karaman and Frazzoli

Algorithm 4 An approach to the minimum-cost path problem. Finds a near-optimal path starting at $\mathbf{q}^{\text{init}}$ and traversing all configurations in A.

for $i = 1$ to $|A|$ **do**
  $\mathbf{T}' \leftarrow \text{RRT}^*(\mathbf{q}_i, K', c)$
  for $j = 1$ to $|A|, j \neq i$ **do**
    $\gamma_{i,j} \leftarrow$ path in $\mathbf{T}'$ s.t. $\gamma_{i,j}(0) = \mathbf{q}_i \wedge \gamma_{i,j}(1) = \mathbf{q}_j$
    $w_{i,j} \leftarrow c(\gamma_{i,j})$
  end for
end for
$\sigma^* \leftarrow \text{Approx-TSP}(A, \gamma, w, \mathbf{q}^{\text{init}})$
return σ^*

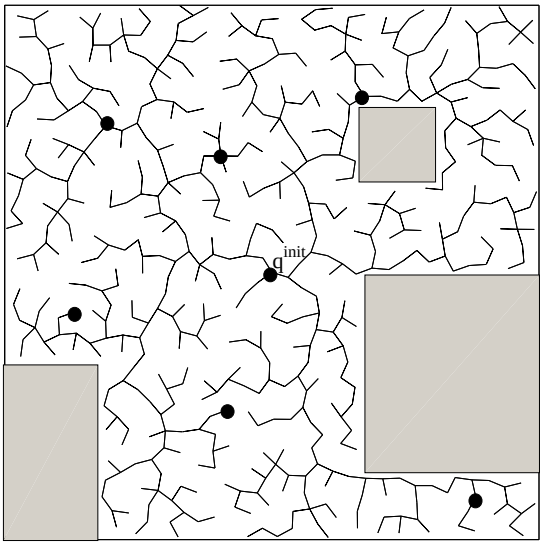


Figure 4.3: An instance of Algorithm 2, part 1. Two-dimensional configuration space of a robot. The grey rectangles correspond to regions that are not in $\mathcal{C}^{\text{free}}$. The bold points are the configurations in A that the robot must visit; the center point is $\mathbf{q}^{\text{init}}$. The RRT connecting all points is also shown.

2011). The RRT* algorithm, proposed by Karaman and Frazzoli (2010), preserves the probabilistic completeness of RRT and is shown to be asymptotically optimal; it is therefore used in Algorithm 4. In contrast to RRT, RRT* examines cumulative costs between newly added points and the root, and reconnects the tree around a newly added points.

The tree holding the RRT* is denoted by $\mathbf{T}'$. A new RRT* is constructed for each configuration $\mathbf{q}_i$ in A. The following parameters are used: a root $\mathbf{q}_i$, a number of samples K' and a cost function c. $\mathbf{T}'$ connects the root $\mathbf{q}_i$ to every configuration $\mathbf{q}_j$ in A. The path joining $\mathbf{q}_i$ and $\mathbf{q}_j$ is denoted by $\gamma_{i,j}$, and its weight (i.e. the cost associated to the path) by $w_{i,j}$. The probability that RRT* fails to find a path from $\mathbf{q}_i$ to $\mathbf{q}_j$ decreases exponentially with the number of samples K' (Karaman and Frazzoli 2011). Nevertheless, it is not guaranteed that the algorithm finds one. If no path is found, we assume that Algorithm 4 uses the path from $\mathbf{T}$ (in Algorithm 2) connecting the two configurations. Figure 4.4 shows the expansion of a single RRT* minimizing the Euclidean distance in a two-dimensional configuration space. The figure highlights the paths found by the tree that connect the root to every configuration in A.

An instance of TSP is now constructed: we possess a complete graph that consists of a set of vertices A and a nonnegative cost w associated to each edge. A symmetric cost function c implies that

$$w_{i,j} = w_{j,i}. \tag{4.12}$$

The $3/2$-approximation algorithm by Christofides (1976) can then be used by Algorithm 4. In the more general asymmetric case (i.e. if c is not symmetric), the equality in (4.12) is not necessarily satisfied and other approximation algorithms should be considered. For the asymmetric version of TSP, an $O(\log n)$-approximate solution is given by Frieze et al. (1982). The work by Asadpour et al. (2010) provides a better solution whose value is within a factor of $O(\log n/\log\log n)$.

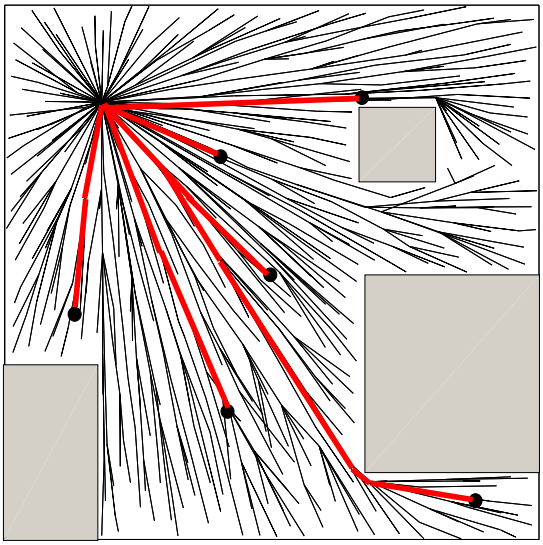


Figure 4.4: An instance of Algorithm 2, part 2. Expansion of a RRT* minimizing the Euclidean distance. The same example from Figure 4.3 is used. The paths connecting the root to every configuration in A are highlighted in red. This step is repeated $|A| = 7$ times by Algorithm 4.

In Algorithm 4, the approximate solution of TSP is denoted by Approx-TSP. The approximation algorithm finds an ordered set of configurations whose first element is $\mathbf{q}^{\text{init}}$. Then, a continuous collision-free path connecting the ordered set is directly formed using γ. The path is assigned to σ^* and is returned by Algorithm 4. Figure 4.5 shows the solution found by Algorithm 4 for the example used above.

4.5.3 Performance analysis

To simplify the performance analysis of Algorithm 2, the following assumptions are made:

- $K = K'$. This assumption is realistic since both the RRT and RRT* trees grow in the same configuration space.
- Approx-TSP runs in $O(|A|^3)$ time (Christofides 1976).

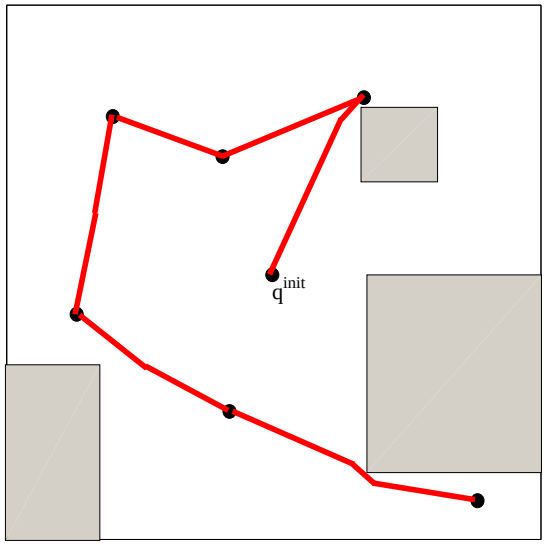


Figure 4.5: An instance of Algorithm 2, part 3. Solution found by Algorithm 4 to the instance given in Figure 4.3. The solution is a path that starts at $\mathbf{q}^{\text{init}}$ and visits every configuration in A.

Constructing the RRT in Algorithm 2 can be performed in $O(K \log K)$ time (Karaman and Frazzoli 2011).

In Algorithm 3, at each iteration, K configurations must be evaluated to find the one that maximizes $|\mathcal{F}(\mathbf{q}) \cap \mathcal{W}^{\text{goal}}|$. This is repeated $|A|$ times. The running time of Algorithm 3 is then $O(|A|K)$.

In Algorithm 4, the RRT* algorithm runs in $O(K \log K)$ (Karaman and Frazzoli 2011) and is called $|A|$ times. The running time of Algorithm 4 is thus given by $O(|A|K \log K + |A|^3)$ which also corresponds to that of Algorithm 2.

4.5.4 Completeness Analysis

The following theorems demonstrate the probabilistic completeness of Algorithm 2.

Lemma 6. *Algorithm 2 returns a correct path* σ^*.

Proof. The planner RRT* finds paths that are free of collisions and continuous. The path σ^*, that results from joining the branches of multiple RRT*, begins at $\mathbf{q}^{\text{init}}$ and visits all configurations in A once—which is sufficient to cover $\mathcal{W}^{\text{goal}}$. □

Theorem 7. *If a solution to the generalized coverage problem exists, the probability that Algorithm 2 returns a correct path converges to 1 as the number of samples used approaches infinity.*

Proof. Algorithm 2 modifies Algorithm 1 by merely altering the returned path, shown to be correct by Lemma 6. And the primary condition (in line 2) is identical in both algorithms, thus the proof follows from Theorem 5. □

4.6 Conclusion

This chapter formulates the generalized coverage problem (GCP) and proves its polynomial-space hardness. A probabilistically complete algorithm (Algorithm 1) based on the rapidly-exploring random trees is provided. The algorithm is optimized (in Algorithm 2) by reducing the cost associated with covering a goal region. This is achieved by minimizing the number of configurations needed to cover the goal region, and by using the optimal rapidly-exploring random trees (RRT*) algorithm to create an instance of the traveling salesman problem.

Algorithm 2 is implemented in the following chapter on a manipulator robot that possesses a range sensor. The cost function used is the one developed in Chapter 3.

Chapter 5

Minimum-Energy Exploration

5.1 Introduction

This chapter is concerned with the problem of autonomous robot exploration in a three-dimensional world. It provides a better understanding of exploration problems with regard to their mathematical formulation and computational complexity, and presents a means to evaluate the performance of future algorithms designed to solve it. In addition, an approach capable of being implemented on a wide range of robotic systems is proposed. The presented algorithm possesses completeness properties, and typically chooses navigation paths that increase the robot's knowledge of its environment while expending minimal energy. It is compared to other exploration approaches capable of being implemented on high-degree-of-freedom robots. The comparison, performed in simulation, shows the significance of reducing energy during the exploration process.

When performing a task like exploration, it is important for a robot to conserve energy. Energy-efficiency can augment the lifetime of a robot operating on a small energy source. When compared to other components, the actuators have a high power consumption (Wang et al. 2005). In this chapter, we study the exploration problem while focusing on energy dissipated through the robot motors.

The robotic exploration problem has been studied extensively. It has been phrased by Thrun et al. (2005) as a decision-theoretic problem. However it is typically presented without a complete formulation, i.e. a mathematical description of the robotic system, the exploration objectives and the physical constraints. With a lack of formulation, the evaluation of exploration algorithms and their comparison become intricate.

The following is a list of goals achieved in this chapter.

1. The exploration problem is formulated as a multi-optimization problem. The formulation is general enough to be applied on many robotic systems operating under different types of constraints, e.g. energy or time constraints.

2. The polynomial-space hardness of the exploration problem is shown.

3. A probabilistically complete exploration algorithm that seeks to minimize the total amount of energy expended during the exploration process is presented. The proposed approach finds low-energy and collision-free paths, and can be applied on high-degree-of-freedom robots.

4. The algorithm is implemented on an articulated robot operating in a three-dimensional workspace. A comparison with other exploration strategies is produced from a large number of trials in a simulated environment. The evaluation methodology is directly obtained from the problem formulation. The results confirm the importance of minimizing energy during the exploration process (as asserted previously.)

The remainder of this chapter is organized as follows. Section 5.2 provides a broad overview of exploration methods. A formulation of the energy-optimal exploration problem is provided in Section 5.3 and the inherent difficulty of the problem is determined. Section 5.4 provides a description of existing approaches able to be implemented on articulated robots in a three-dimensional environment. Section 5.5

proposes an algorithm that generates exploration paths that generally minimize energy dissipation through the robot's motors. The proposed algorithm is evaluated using a simulated anthropomorphic arm with range sensor in Section 5.6, and a comparison with other exploration strategies is given.

5.2 Related work

5.2.1 Exploration in topological maps

Early approaches in robotic exploration were conducted by Kuipers and Byun (1991) and Dudek et al. (1991). They propose a hierarchical map, characterized by a topological layer represented as a graph, where vertices are distinctive places in the environment. Traditionally, mapping methods are divided into topological maps that model the connectivity of distinctive places in the environment, and metric maps that model its geometric properties (Thrun 2003). Exploration methods that use metric maps (e.g. occupancy grid maps) are examined below.

5.2.2 Exploration with mobile robots

Yamauchi (1997) introduces an exploration method that uses an occupancy grid and is based on the concept of *frontiers*, regions of the environment that lie between the explored space and the unexplored space; the robot explores by traveling to these regions. Other methods (Makarenko et al. 2002; Gonzalez-Banos and Latombe 2002; Thrun et al. 2005) include a navigation cost to the expected information gain at a position in the map; the robot moves to the position that maximizes the difference between information gain and cost. A comparison between different exploration strategies for mobile robots is conducted by Holz et al. (2010). The criterion used in their comparative evaluation is the total length of the robot's trajectory.

5.2.3 Exploration with manipulator robots

The methods cited above are designed toward mobile robots and are thus not necessarily practical for an articulated robot. Other exploration methods concentrate on manipulator robots that have a sensor attached to their end effector. This type of robot is used to create a 3D model of an object of interest in the environment (Whaite and Ferrie 1997; Torabi and Gupta 2012) by using the *next best view* (NBV) paradigm. The goal is to determine the next (best) sensor configuration to scan the object with a minimum number of scans. A NBV approach is also used by Wang and Gupta (2007) to explore the robot's configuration space with a manipulator. Their work introduces a measure, called *configuration space entropy*, that expresses the robot's ignorance of its configuration space. This measure is used to find the next sensor configuration that maximizes the robot's knowledge of its configuration space. Another method (Renton et al. 1999) explores by scanning selected *targets* in the workspace, where a target is a three-dimensional point in the unknown space. This exploration strategy is less constrained than other NBV approaches because the manipulator's sensor is not required to reach an exact configuration (since a target can be viewed by an infinite number of configurations.) The NBV methods produce, at each iteration, one or more desired operational space configurations for a sensor. These methods rely upon a planner to find a collision-free path to a goal configuration. Other methods (discussed below) for exploration with a manipulator only use an external planner in exceptional cases.

A function that maps a configuration to a *rating* is introduced by Kruse et al. (1996). Their exploration approach searches along the gradient of the rating function to find a goal configuration. In most cases, the goal configuration can be reached through a linear path. The rating function uses a weight that specifies the importance given to acquire new information at the expense of navigation cost. The rapidly-exploring random tree (RRT) algorithm (Kuffner and LaValle 2000) (which has been employed heavily in motion planning for exploration of high-dimensional spaces) is used as an integral part of the

exploration approach (Freda et al. 2008, 2009). RRTs bias search into the largest Voronoi regions (LaValle 2006), which makes them efficient for solving motion planning problems. The RRT-based method expands a RRT in a predefined subset of the configuration space. The subset contains all the configurations included in an n-ball of a predefined radius. This constrained exploration is necessary to execute a "world-based depth-first traversal" of the robot's physical world and to prevent continuous backward and forward motion between distant configurations.

5.2.4 Energy-optimal motion planning and exploration

In contrast to the work cited above, our work focuses on minimizing actuator energy dissipation during the exploration. Energy-optimal motion planning has been studied extensively (von Stryk and Schlemmer 1994; Galicki 2000; Choset et al. 2005; Gregory et al. 2012) but has been limited to planar robots, low-degree-of-freedom robots, or point-to-point motions. Energy-optimal robot exploration has surprisingly been studied little. The method of Mei et al. (2006) uses the concept of frontiers (Yamauchi 1997) in order to direct a mobile robot via a locomotion strategy toward minimizing energy. They compare their method to a simple greedy strategy.

A vein of research by Willow Garage (Rusu et al. 2009; Sucan et al. 2010) has motivated our work. Their use of multiple sensors—3D laser range sensors in particular—to construct models of indoor environments for motion planning raises the question of what algorithms should be used to explore an environment efficiently.

5.3 Problem definition

The energy-optimal exploration problem (EEP) is formulated here as a bi-objective optimization. It is informally defined as follows. Given a robot initial state (i.e. its initial configuration and velocity vectors), the goal is to maximize its knowledge of an unknown

environment while minimizing the amount of energy expended during that process. We require that the exploration itself does not cause the robot to contact obstacles; we assume that the environment is static (in order to avoid contact) and that the robot's configuration is known throughout the exploration process, i.e. there is no proprioceptive uncertainty. The transformation to a formal description follows.

5.3.1 Notation

Let $\mathcal{W}$ be a bounded region in a three-dimensional space that represents the robot's physical world. The portion of $\mathcal{W}$ unknown to the robot at time t is denoted $\mathcal{W}_t^{\mathrm{u}}$. The known portion, at time t, is subdivided into the free space and the occupied (also called obstacle) space, respectively denoted $\mathcal{W}_t^{\mathrm{free}}$ and $\mathcal{W}_t^{\mathrm{obs}}$. We have

$$\mathcal{W} = \mathcal{W}_t^{\mathrm{u}} \cup \mathcal{W}_t^{\mathrm{free}} \cup \mathcal{W}_t^{\mathrm{obs}}, \tag{5.1}$$

$$\mathcal{W}_t^{\mathrm{u}} \cap \mathcal{W}_t^{\mathrm{free}} = \mathcal{W}_t^{\mathrm{free}} \cap \mathcal{W}_t^{\mathrm{obs}} = \mathcal{W}_t^{\mathrm{obs}} \cap \mathcal{W}_t^{\mathrm{u}} = \emptyset. \tag{5.2}$$

As shown in (5.1) and (5.2), $\mathcal{W}$ is equal to the union of the three disjoint regions $\mathcal{W}_t^{\mathrm{u}}$, $\mathcal{W}_t^{\mathrm{free}}$ and $\mathcal{W}_t^{\mathrm{obs}}$. We denote their respective volumes by $|\mathcal{W}_t^{\mathrm{u}}|$, $|\mathcal{W}_t^{\mathrm{free}}|$ and $|\mathcal{W}_t^{\mathrm{obs}}|$.

Let $\mathcal{C}$ be the n-dimensional configuration space of the robot. $\mathcal{C}_t^{\mathrm{free}}$ denotes the free configuration space at time t. Let $\mathbf{q} \in \mathcal{C}$ be a configuration. The physical space occupied by the robot at $\mathbf{q}$ is denoted by $\mathcal{R}(\mathbf{q})$. $\mathbf{q} \in \mathcal{C}_t^{\mathrm{free}}$ if and only if $\mathcal{R}(\mathbf{q})$ does not intersect with $\mathcal{W}_t^{\mathrm{u}}$ and $\mathcal{W}_t^{\mathrm{obs}}$.

5.3.2 Exploration algorithm and termination

The exploration is carried out by an algorithm that is applied to the robot's knowledge of the world at time t (i.e. $\mathcal{W}_t^{\mathrm{u}}$, $\mathcal{W}_t^{\mathrm{free}}$ and $\mathcal{W}_t^{\mathrm{obs}}$) to obtain a feasible control $\mathbf{u}(t)$. The algorithm stops when a termination criterion is reached; let T denote the time

length of the exploration. $|\mathcal{W}_T^{\mathrm{u}}|$ represents the volume of the region unexplored at T. Ideally, the robot keeps exploring until the totality of the world is known, i.e. $|\mathcal{W}_T^{\mathrm{u}}| = 0$. However, this termination criterion is impractical: regions in the unknown space might be unobservable or unreachable due to the robot's geometry. We propose an alternative criterion for terminating the exploration in Section 5.5.

The control vector $\mathbf{u}(t)$ represents the torque vector at time t denoted by $\tau(t)$. Because torque is proportional to current in a DC motor, the integral-squared torque can be used as an approximation of energy dissipated in the robot's motors (Chevallereau et al. 2009; Haq et al. 2012). The total energy expended by the actuators is thus given by

$$J_T := \int_0^T \tau(t)^{\mathsf{T}} \tau(t)\, \mathrm{d}t, \tag{5.3}$$

where J_T is equal to zero if no torque is applied to the robot and is positive otherwise.

5.3.3 Problem formulation

We can quantify the informal statement given at the beginning of this section by proposing the problem of finding $\tau(t)$, $0 \leq t \leq T$, to:

$$\begin{aligned}
\text{minimize} \quad &: \quad [|\mathcal{W}_T^{\mathrm{u}}|, J_T]^{\mathsf{T}} && (5.4)\\
\text{subject to} \quad &: \quad (\mathbf{q}(0), \dot{\mathbf{q}}(0)) = (\mathbf{q}_0, \dot{\mathbf{q}}_0) && (5.5)\\
&\quad\ \ \mathbf{q}(t) \in \mathcal{C}_t^{\text{free}}\ \forall t && (5.6)\\
&\quad\ \ \ddot{\mathbf{q}} = \mathbf{M}(\mathbf{q})^{-1}\{\tau - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{g}(\mathbf{q})\}, && (5.7)
\end{aligned}$$

where $(\mathbf{q}_0, \dot{\mathbf{q}}_0)$ is the robot's initial state, $\mathbf{M}(\mathbf{q})$ is the robot's generalized inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ denotes the Coriolis and centrifugal forces, and $\mathbf{g}(\mathbf{q})$ is the vector of gravity forces. Equation (5.7) is the well known equation for multi-rigid-body dynamics.

A solution to (5.4) is a set of control functions (known as the Pareto optimal set) instead of a single optimal solution. We transform EEP into a single-objective optimization problem by applying the constrained objective function method (Haimes et al. 1971). More importance is awarded to the first criterion $|\mathcal{W}_T^{\mathrm{u}}|$; the second criterion J_T is used to form an additional constraint. The resulting problem consists of finding $\tau(t)$, $0 \leq t \leq T$, to:

$$\begin{aligned} \text{minimize} \quad &: \quad |\mathcal{W}_T^{\mathrm{u}}| && (5.8) \\ \text{subject to} \quad &: \quad \text{Equations (5.5, 5.6, 5.7)} \\ &\quad\;\; J_T \leq J^{\max}, && (5.9) \end{aligned}$$

where the parameter $J^{\max}$ is an upper bound for the amount of energy expended during the exploration.

When using the bounded objective function method we choose to place the second criterion J_T as a constraint. As mentioned above, $|\mathcal{W}_T^{\mathrm{u}}|$ is treated as a more important criterion, however there are other arguments in favor of this decision. Given that the workspace $\mathcal{W}$ is bounded, then for a large $J^{\max}$, the function $|\mathcal{W}_T^{\mathrm{u}}|$ will surely converge to a single value; which makes the comparison between different exploration methods easier. This can be seen in the results section. Furthermore, The value of $J^{\max}$ can have a practical significance, e.g. it can represent the robot's total amount of power supply.

EEP can be generalized to account for other constraints, i.e. one can add to the constraints a set of inequalities of the form $C_T \leq C^{\max}$, where C_T is an objective function and $C^{\max}$ is its corresponding upper bound. For example, the time length of the exploration can be limited by adding a positive value $T^{\max}$ and the constraint

$$\int_0^T \mathrm{d}t \leq T^{\max}. \tag{5.10}$$

With this method, formulating the following problem becomes straightforward: maximize the robot's knowledge of an unknown environment while minimizing the amount of energy expended and the time length of the exploration.

5.3.4 Complexity analysis

In this section, we show that a solution to EEP that runs in polynomial time is unlikely to be found. This can be shown by proving that the problem is at least in PSPACE-hard.

Theorem 8. *The energy-optimal exploration problem is polynomial-space hard.*

Proof. We prove that EEP is polynomial-space hard by showing that the motion planning problem is polynomial-time reducible to EEP. Motion planning is well-studied in robotics, it is known to be polynomial-space complete (Reif 1979; Canny 1988). An instance of the motion planning problem is an initial and a goal configuration in $\mathcal{C}$ respectively denoted $\mathbf{p}_0$ and $\mathbf{p}_1$. The problem is to find a continuous collision-free path from $\mathbf{p}_0$ to $\mathbf{p}_1$.

Let $\mathcal{V}(\mathbf{q})$ denote the field of view of the robot at $\mathbf{q}$, a region in $\mathcal{W}$. An instance of EEP can be constructed, in constant time, from an instance of the motion planning problem as follows:

$$\mathbf{q}_0 \leftarrow \mathbf{p}_0 \tag{5.11}$$

$$\mathcal{V}(\mathbf{q}) \leftarrow \begin{cases} \mathcal{W}_0^{\mathrm{u}} & \text{if } \mathbf{q} = \mathbf{p}_1 \\ \emptyset & \text{otherwise.} \end{cases} \tag{5.12}$$

$$J^{\max} \leftarrow \infty. \tag{5.13}$$

Equation (5.12) assigns to the field of view the empty set for any configuration not equal to $\mathbf{p}_1$. Therefore, if $\mathbf{p}_1$ can be reached from $\mathbf{p}_0$, an algorithm solving EEP must move the robot to $\mathbf{p}_1$ to explore the unknown space $\mathcal{W}_0^{\mathrm{u}}$. The exploration is not limited by

the total energy expended, as stated in Equation (5.13), and stops when $\mathbf{p}_1$ is reached. Thus, the solution found is a continuous collision-free path starting at $\mathbf{p}_0$ and ending at $\mathbf{p}_1$; which completes the proof. □

Because the problem is inherently difficult to solve, standard techniques found in calculus of variations and optimal control theory cannot be applied to find optimal solutions. For this reason, suboptimal solutions must be considered and are discussed in the following two sections.

5.4 Existing solutions

This section describes exploration algorithms that can be implemented on robots that possess multiple degrees of freedom and can operate in a three-dimensional workspace. The following existing approaches (evaluated in Section 5.6) optimize different criteria to carry out an efficient exploration, however they do not explicitly minimize energy.

5.4.1 Rating functions

The *rating functions* (RF) approach, described in Kruse et al. (1996), introduces a function that maps a configuration $\mathbf{q} \in \mathcal{C}^{\text{free}}$ to a *rating*. The rating combines the predicted amount of new information obtained at $\mathbf{q}$ and the Euclidean distance between the current configuration $\mathbf{q}_0$ and $\mathbf{q}$. At each iteration of the algorithm, the gradient of the rating function is calculated, and safe configurations along the gradient are examined. The algorithm chooses the one with the highest rating. When a local maximum is attained, the search proceeds to a random configuration that has a positive rating.

If $\mathbf{q}$ cannot be reached through a linear path, a path planning algorithm must be called, e.g. the rapidly-exploring random tree planner.

5.4.2 Maximal expected entropy reduction

The *maximal expected entropy reduction* (MER) method is described by Wang and Gupta (2007) and is a continuation of the work by Yu and Gupta (2004). This work introduces a measure, called *configuration space entropy*, that expresses the robot's ignorance of its configuration space. MER defines a function of a sensor's configuration $\mathbf{s}$, called the expected entropy reduction function, and denoted $\widetilde{ER}$. The goal is to move the robot to the configuration that maximizes $\widetilde{ER}(\mathbf{s})$ at each iteration. $\widetilde{ER}(\mathbf{s})$ is approximated by performing a summation of marginal entropies over a large enough number of samples in $\mathcal{C}$.

MER performs a sensor-based incremental construction of a probabilistic road map (Yu and Gupta 2000) whose nodes are configurations in $\mathcal{C}^{\text{free}}$. MER uses this roadmap to plan a path to the next desired sensor's configuration.

5.4.3 Sensor-based exploration tree

The *sensor-based exploration tree* (SET) was developed by Freda et al. (2008); it was generalized to multiple sensors in the work by Freda et al. (2009). More specifically, we are interested in *SET with local growth*. As shown by Freda et al. (2008), it performs better than the alternative (*SET with global growth*).

At each iteration, SET computes the robot's *local free boundary* (LFB) at the current configuration $\mathbf{q}_0$. The LFB contains all points of the frontier (see Section 5.1) that are both in $\mathcal{C}^{\text{free}}$ and can be viewed from the sensor at an *admissible* configuration. A configuration $\mathbf{q}$ is admissible if the sensor position at $\mathbf{q}$, denoted $s(\mathbf{q})$, is within a maximum distance ρ of $s(\mathbf{q}_0)$, and the line segment between $s(\mathbf{q})$ and $s(\mathbf{q}_0)$ does not intersect a know obstacle.

Let $\mathcal{D}$ denote the subset of $\mathcal{C}$ containing all the configurations included in the n-ball of radius δ and center $\mathbf{q}_0$. If the LFB is not empty, SET locally expands an RRT (rooted at $\mathbf{q}_0$) in $\mathcal{D}$. Then, the algorithm extracts from the RRT all admissible configurations. The robot moves to the admissible configuration that maximizes the predicted amount of new

information obtained by the sensor. If the LFB is empty, the robot backtracks to a previous configuration.

If the RRT fails to find a configuration that increases the predicted information, a second tree is expanded in $\mathcal{D}$ without collision detection. Only the admissible configurations are added to the lazy tree. The robot moves to a safe configuration in the lazy tree that increases the predicted information after a path planner is called. In the case where both the RRT and the lazy tree fail, the robot backtracks to a previous configuration.

5.5 Proposed Solution

The algorithms described in the previous section do not attempt at explicitly finding exploration paths that reduce energy. To address EEP we propose an alternative exploration algorithm described here.

5.5.1 Effectiveness of exploration paths

Let $\mathbf{q}_0$ be the robot current configuration and $\sigma = (\mathbf{q}_0, ..., \mathbf{q}_k)$ be a path in $\mathcal{C}^{\text{free}}$. We seek to develop a utility function U that predicts the effectiveness of the move along a path σ. This measure combines the predicted volume of the region explored by the robot and the predicted amount of energy required to reach $\mathbf{q}_k$.

Predicted volume of the region explored

Let $\mathbf{s}_k$ denote the sensor's operational space configuration (i.e. a point in $SE(3)$) at $\mathbf{q}_k$. Given $\mathbf{q}_k$, $\mathbf{s}_k$ can be computed using the robot's forward kinematics. The observable portion of $\mathcal{W}$ from $\mathbf{s}_k$ is called the field of view (FOV). We denote by v_k the predicted increase in volume attained by moving to $\mathbf{q}_k$, we have $v_k \geq 0$. The quantity v_k is determined by finding the portion of the FOV that intersects $\mathcal{W}^{\text{u}}$ and is not obstructed by

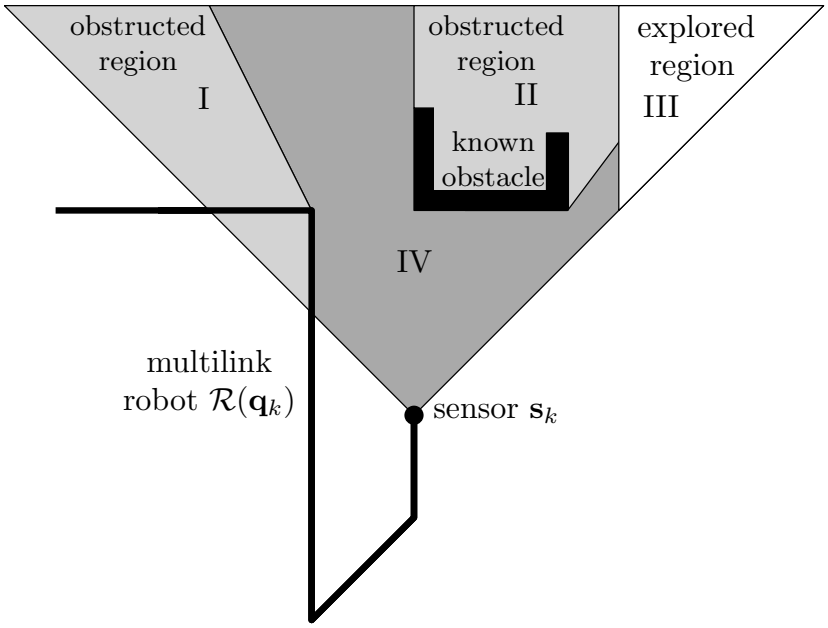


Figure 5.1: Field of view of a sensor placed on the end effector of a multilink robot, in a two-dimensional workspace. The FOV is subdivided into four regions. Region I is unknown and obstructed by the physical space occupied by the robot $\mathcal{R}(\mathbf{q}_k)$. Region II is unknown and obstructed by the known occupied space $\mathcal{W}^{\text{obs}}$. Region III intersects with the known free space $\mathcal{W}^{\text{free}}$. Region IV is the region of interest, it is in the FOV, and is neither explored nor obstructed. v_k is calculated by finding the area of Region IV.

$\mathcal{R}(\mathbf{q}_k)$ and $\mathcal{W}^{\text{obs}}$. This is illustrated in Figure 5.1. Note that v_k does not incorporate exploration due to sensor views at intermediate configurations between $\mathbf{q}_0$ and $\mathbf{q}_k$.

Predicted energy

The predicted amount of energy required to move along the path σ is denoted by c_k and is given by

$$c_k = \begin{cases} \varepsilon & \text{if } k = 0 \\ c_{k-1} + \hat{J}_k & \text{if } k > 0, \end{cases} \tag{5.14}$$

where ε is a positive infinitesimal quantity, and $\hat{J}_k$ represents the (approximated) amount of energy needed to travel from $\mathbf{q}_{k-1}$ to $\mathbf{q}_k$. We have $c_k > 0$.

We calculate $\hat{J}_k$ by using the integral-squared torque approximation method described in Chapter 3. $\hat{J}_k$ is then given by

$$\hat{J}_k = \sum_{i=0}^{m-2} \tau_k(t_i)^\mathsf{T} \tau_k(t_i) \Delta t, \tag{5.15}$$

where the torque vector $\tau_k(t_i)$ is determined by solving the inverse dynamics problem at $t_i = i\Delta t$.

Utility function

The predicted effectiveness of the move along the path σ is given as a function of v_k and c_k as follows:

$$U(\sigma) = f(v_k, c_k). \tag{5.16}$$

The function U is a utility function that combines information gain and cost to articulate the exploration objectives. Thrun et al. (2005) and Kruse et al. (1996) express U as a weighted sum, whereas Gonzalez-Banos and Latombe (2002) choose an exponential weighted utility function. In this work we choose the weighted product method. This type of functions is used in multi-objective optimization when dealing with quantities having different orders of magnitude (Marler and Arora 2004), such as v_k and c_k. Thus we have

$$U(\sigma) = {v_k}^{\alpha} {c_k}^{\alpha-1}, \tag{5.17}$$

where α is a weight with $0 < \alpha < 1$. A large α means that maximizing information gain is favored over minimizing cost. The function $U(\sigma)$ is equal to zero if $v_k = 0$ and is positive otherwise.

5.5.2 Path planning

The probabilistic roadmap and the rapidly-exploring random tree (RRT) algorithms have been used in the context of exploration (Wang and Gupta 2007; Freda et al. 2008, 2009). Such algorithms are shown to lack in asymptotic optimality with respect to a given cost function (e.g. Equation (5.14)) (Karaman and Frazzoli 2011). The nonoptimality of RRT is avoided by using RRT*, shown to be asymptotically optimal. We denote by τ-RRT* the RRT* algorithm that uses the cost in (5.14).

5.5.3 Algorithm

At each iteration Algorithm 5 expands a tree $\mathbf{T}$, from $\mathbf{q}_0$, using the τ-RRT* algorithm. The total number of configurations sampled by τ-RRT* is K (an exploration parameter.) The robot chooses the path σ^* in $\mathbf{T}$ that maximizes the utility function in (5.17). We assume that, while moving along the path σ^*, the robot senses the environment and updates a map (the robot's current representation of the physical world.) The processes of sensing the environment and updating the map are described in the next section.

The algorithm terminates if $U(\sigma^*)$ is equal to zero, i.e. $v_k = 0$ for all the configurations added to $\mathbf{T}$. Such condition signifies that the algorithm cannot find an informative configuration, i.e. a configuration that reduces the size of $\mathcal{W}^{\mathrm{u}}$.

5.5.4 Running time

Building the τ-RRT* is performed in $O(K \log K)$-time (Karaman and Frazzoli 2011), where K is the number of samples. Finding σ^* is performed in $O(K)$-time with a preorder traversal of $\mathbf{T}$.

Algorithm 5 An approach to the energy-optimal problem

exploration-completed $\leftarrow$ *false*
while *exploration-completed* $=$ *false* **do**
  $\mathbf{q}_0 \leftarrow$ current configuration of the robot
  $\mathbf{T} \leftarrow \tau$-RRT$^*(\mathbf{q}_0, K)$
  $\sigma^* \leftarrow$ path in $\mathbf{T}$ that maximizes (5.17)
  if $U(\sigma^*) > 0$ **then**
    move the robot along σ^*
  else
    exploration-completed $\leftarrow$ *true*
  end if
end while

5.5.5 Completeness analysis

We begin by defining the notion of completeness in the context of exploration. Let $\mathcal{W}'(\mathbf{q}_0)$ represent the explorable region of $\mathcal{W}$ given the initial configuration $\mathbf{q}_0$. A complete algorithm explores $\mathcal{W}'(\mathbf{q}_0)$ entirely, and we have

$$\mathcal{W}_0^{\mathrm{u}} \setminus \mathcal{W}_T^{\mathrm{u}} = \mathcal{W}'(\mathbf{q}_0). \tag{5.18}$$

We now show that Algorithm 5 possesses probabilistic completeness.

Theorem 9. *The probability that Algorithm 5 explores $\mathcal{W}'(\mathbf{q}_0)$ converges to 1 as the number of samples K approaches infinity.*

Proof. Suppose that Algorithm 5 terminates with

$$\mathcal{W}_0^{\mathrm{u}} \setminus \mathcal{W}_T^{\mathrm{u}} \subset \mathcal{W}'(\mathbf{q}_0), \tag{5.19}$$

then there exists a point $P \in \mathcal{W}'(\mathbf{q}_0)$ such that $P \notin \mathcal{W}_0^{\mathrm{u}} \setminus \mathcal{W}_T^{\mathrm{u}}$ (i.e. P is not explored.)

Since P is in the explorable region, there exists at least one configuration $\mathbf{q}$ such that $\mathbf{q}$ is reachable from $\mathbf{q}_0$ and P is in the robot's field of view at $\mathbf{q}$. We say that $\mathbf{q}$ is *informative*. At each iteration, Algorithm 5 creates a RRT tree that expands in the

configuration space. For a number of samples K approaching infinity, the probability that a RRT reaches $\mathbf{q}$ converges to 1 (LaValle and Kuffner 1999).

Algorithm 5 terminates if no informative configuration is found. Therefore, for a number of samples K approaching infinity the probabilities that $\mathbf{q}$ is not found and that P is not explored converge to 0. □

5.6 Evaluation

The evaluation is made entirely in simulation, which allows us to obtain results from multiple trials repeated on different environments. Simulation permits ready design of heterogeneous environments for experimentation and accurate reproduction of initial conditions.

5.6.1 Simulated environment

The robot is dynamically simulated using Featherstone's method for articulated bodies (Featherstone 1987). The simulated robot is an anthropomorphic manipulator with a fixed base: the arm possesses six degrees of freedom and the gripper two degrees of freedom. A simulated range sensor is positioned on the wrist of the arm. The "sensor" provides a 2D array of depth readings using the z-buffer algorithm. The manipulator and the sensor's FOV are depicted in Figure 5.2.

The simulated anthropomorphic robot arm, like a true human arm, possesses limited flexibility. As the sensor is placed on the wrist, there are many operational space configurations for the sensor that are impossible for the arm to reach. In order to give the robot some initial freedom of movement, the arm is assumed to have a "safe" region; as shown in Figure 5.3, the safe region extends in a sphere from the manipulator's shoulder to its elbow.

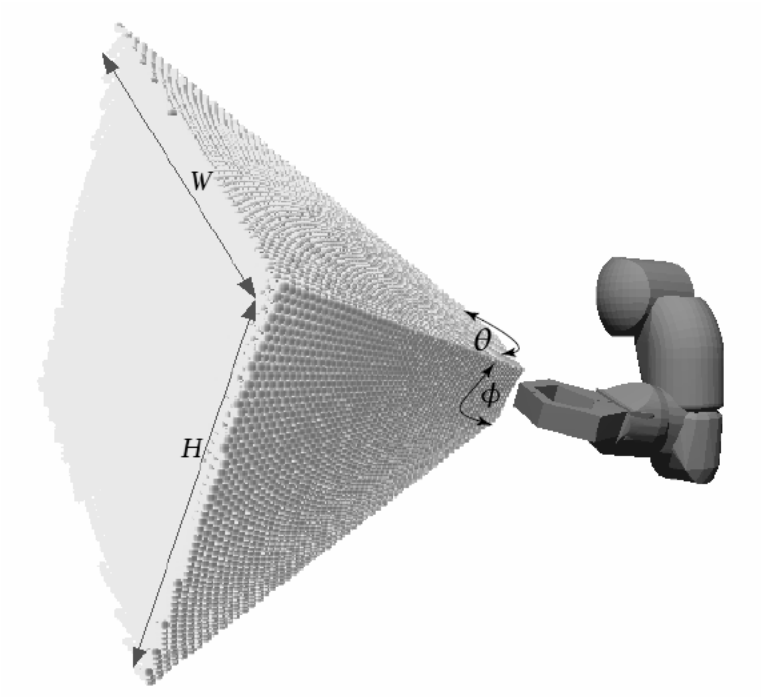


Figure 5.2: The virtual 3D laser range scanner uses a resolution of 50×50 (i.e. $H = W = 50$) and a FOV defined by $\theta = \phi = 0.8$ rad.

Figure 5.3: The safe region for the robot used in the simulations. The safe region, which is a region assumed to be clear of obstacles that extends in a sphere from the manipulator's shoulder to its elbow, is used to give the robot some freedom of movement for exploration.

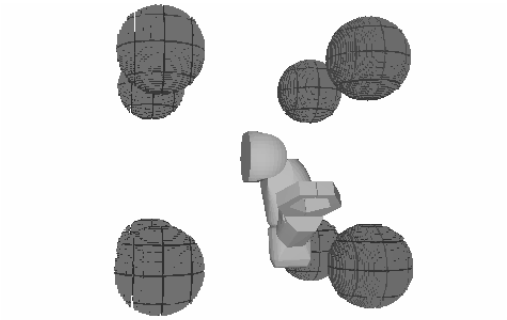
The depth readings provided by the sensor update an octree representation of the environment. The octree encodes both the unknown space $\mathcal{W}^{\text{u}}$ and the occupied space $\mathcal{W}^{\text{obs}}$. At the start of the exploration, all of the leaves of the octree are initialized to the unknown state. The safe region, which belongs to $\mathcal{W}^{\text{free}}$, is removed from the octree. The robot at configuration $\mathbf{q}$ is said to be in collision, i.e. $\mathbf{q} \notin \mathcal{C}^{\text{free}}$, if its geometric model intersects with the octree.

Three environments, with varied obstacle sizes and shapes, are employed. The first environment (E1), shown in Figure 5.4(a), is spacious and contains large obstacles. Figure 5.4(b) shows a cluttered space (E2), for which it is harder to plan a path between two configurations. The total volume of free space is comparatively largest in this environment. The third environment (E3), shown in Figure 5.4(c), is a confined space that gives the robot very little space to maneuver. The total volume of free space is comparatively smallest in E3. Note that, in all three environments, the robot is unable to explore the totality of the unknown space due to the arm's limited flexibility and the presence of obstacles that both prevent the arm from reaching certain configurations and obstruct the sensor's FOV.

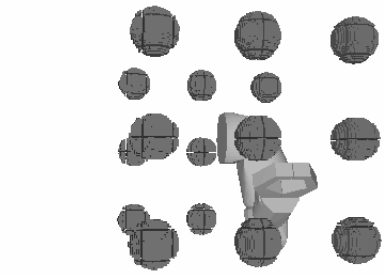
5.6.2 Results

Exploration algorithms

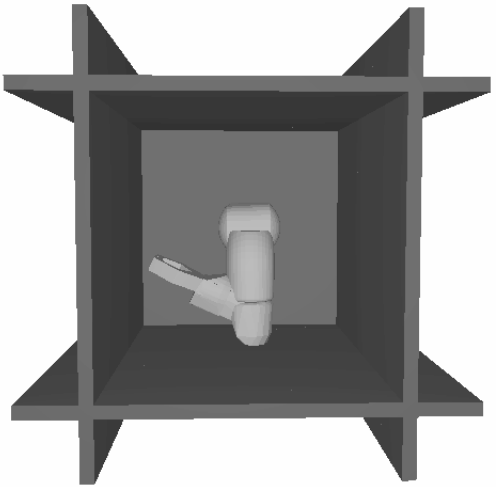
The algorithms implemented for the evaluation are summarized in Table 5.1. EXP-τ-RRT* is the algorithm presented in Section 5.5. The following three algorithms are similar to EXP-τ-RRT*, but they each retain a unique exploration component. EXP-τ-RRT uses the RRT path planner. EXP-d-RRT* substitutes the cost given in (5.14) by the Euclidean distance. The algorithm EXP-τ-RRT*-v does not employ the utility function in (5.17). It performs a greedier exploration by choosing, at each iteration, the path that maximizes the volume v_k.



(a) Environment E1



(b) Environment E2



(c) Environment E3

Figure 5.4: The three 3D environments used in the evaluation of the exploration algorithm on the simulated manipulator arm.

The following three algorithms are existing approaches described in Section 5.4, namely, rating functions (RF), maximal expected entropy reduction (MER) and sensor-based exploration tree (SET). The are used in our comparison because they can be implemented on robots that possess multiple degrees of freedom and can operate in a three-dimensional workspace

Finally, the algorithm COV is is the coverage algorithm described in Chapter 4, Algorithm 2. COV cannot be used as an exploration algorithm as it assumes a full knowledge of the workspace W. It is implemented here as a comparative reference.

Table 5.1: The exploration algorithms used in the evaluation.

Exploration Algorithm	Path Planner	Cost Function	Utility Function
EXP-τ-RRT*	RRT*	integral-sqr. torque	weighted product
EXP-τ-RRT	RRT	integral-sqr. torque	weighted product
EXP-d-RRT*	RRT*	Euclidean distance	weighted product
EXP-τ-RRT*-v_k	RRT*	integral-sqr. torque	volume
RF	RRT	Euclidean distance	weighted sum
MER	PRM	–	entropy reduction
SET	RRT	Euclidean distance	volume
COV	RRT/RRT*	integral-sqr. torque	–

The following parameters are chosen in the implementation:

- In Section 5.5, the parameters m and Δt respectively designate the number of points needed to calculate the sum in (5.14) and the small time interval between two consecutive points. Our experience indicates that a small number of points suffices for obtaining good approximations. In our implementation, $m = 5$ and $\Delta t = 0.1$ second.

- The weighted product method uses a quantity α set to 0.5. Such value of α gives equal importance to optimizing information gain and cost.

- The parameter K designates the number of configurations sampled by RRT and RRT*. We choose $K = 3000$. We find that with a smaller K the environments tend to be incompletely explored.

Evaluation

The evaluation is performed based on the optimal control problem defined in (5.8). Every exploration starts with identical initial conditions. To execute a path returned by an exploration algorithm, the robot arm is driven by torque computed by a composite inverse dynamics / PID controller, where the velocities and accelerations are obtained with a timing law of cubic splines.

The exploration halts if any of the following occurs:

1. The path planner cannot find a configuration whose utility function value is positive.
2. The amount of energy expended by the robot (computed via (5.3)) exceeds the upper bound $J^{\max}$.

However, the coverage algorithm COV is an exception and terminates only after the entire path is executed, and the goal region is totally covered.

Due to the fact that all of the exploration approaches are pseudorandom, we carry out 50 trials for each approach on every simulated environment. The comparison between the variants of Algorithm 5, in the environments E1, E2 and E3 is found in Figures 5.5, 5.6 and 5.7 respectively. The mean of the total volume explored is shown for values of $J^{\max}$ between 0 and 10 unit energy. The comparison between Algorithm 5 and the remaining algorithms in Table 5.1, in the environment E1, is found in Figure 5.8. (Similar results for the environments E2 and E3 were obtained and are not provided here.) The mean of the total volume explored is shown for values of $J^{\max}$ between 0 and 30 unit energy.

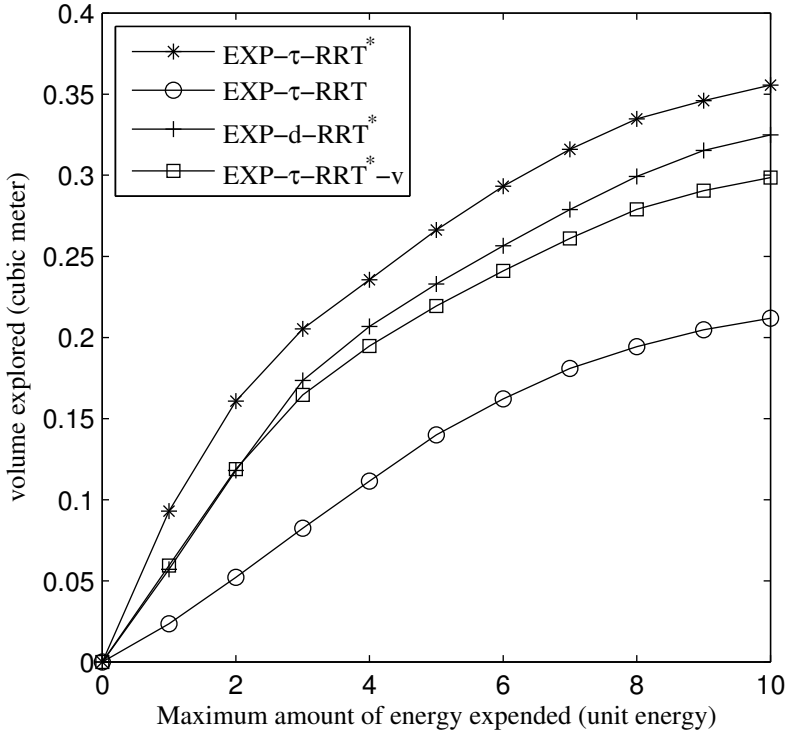


Figure 5.5: Volume explored for values of $J^{\max}$ between 0 and 10 unit energy in environment E1.

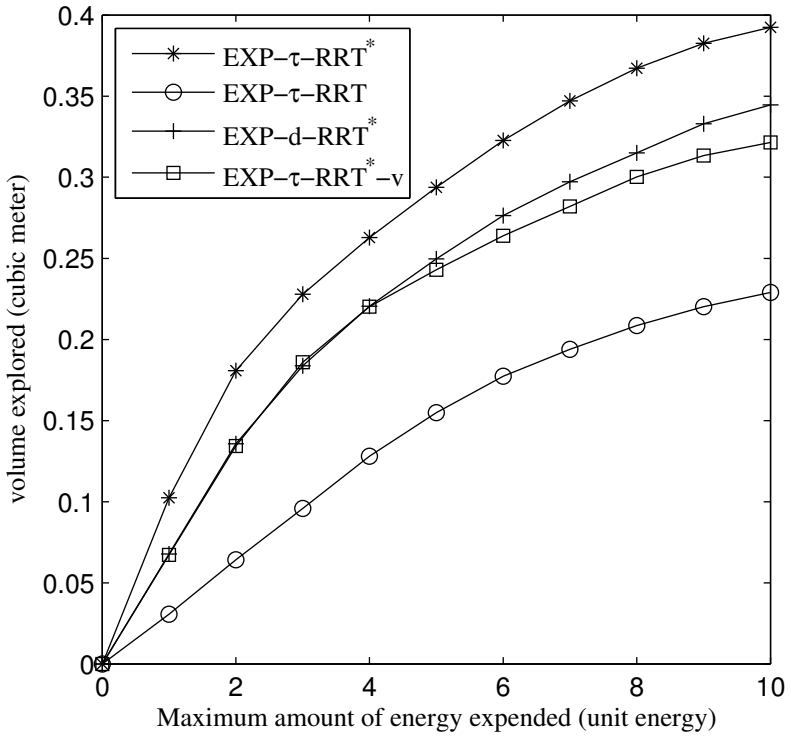


Figure 5.6: Volume explored for values of $J^{\max}$ between 0 and 10 unit energy in environment E2.

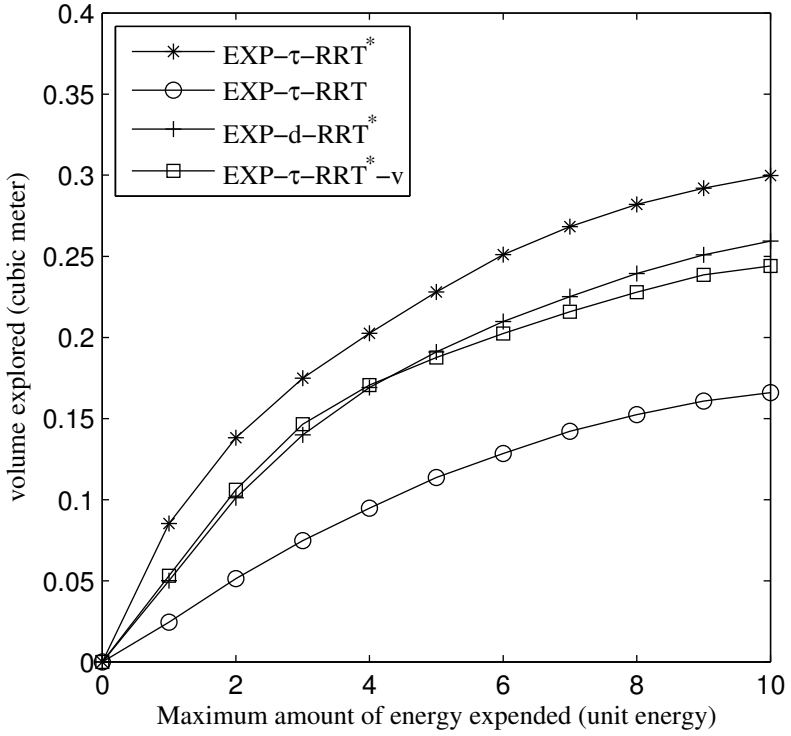


Figure 5.7: Volume explored for values of $J^{\max}$ between 0 and 10 unit energy in environment E3.

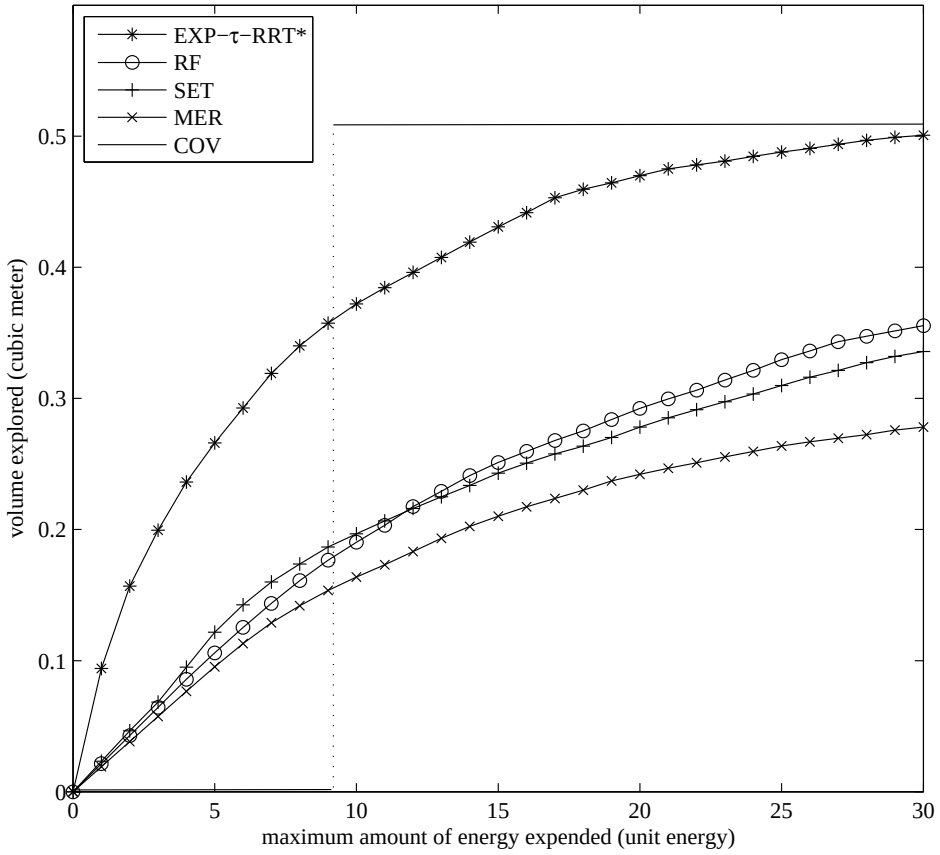


Figure 5.8: Volume explored for values of $J^{\max}$ between 0 and 30 unit energy in environment E1 compared to existing approaches.

Discussion

We observe that the relative efficiency of the algorithms is consistent among the three environments. The total volume explored is smaller in environment E3 where the robot's view is the most obstructed. We find that EXP-τ-RRT* and EXP-d-RRT* perform better than the other two methods as they are both successful in reducing unnecessary movement of the arm. However the energy-based cost function offers an advantage over the Euclidean distance. Our results show a better performance of the RRT* path planner when compared to RRT, as EXP-τ-RRT is consistently less efficient than the other three algorithms. Under the same conditions, EXP-τ-RRT* shows an improvement of over 20% in all environments when compared to EXP-τ-RRT*-v; which emphasizes the importance of using a utility function. We conclude that, based on the results, every component of Algorithm 5 contributes to a more efficient exploration.

The results also show that EXP-τ-RRT* performs better than other existing exploration approaches on average. Specifically, Figure 5.8 shows that EXP-τ-RRT* has the largest mean value when compared to RF, MER and SET. These algorithms do not attempt at reducing energy during the exploration process, which results in a less efficient exploration (with respect to the formulation in (5.8).)

Finally, the hypothesis (made in Section 5.3) that $|\mathcal{W}_T^{\mathrm{u}}|$ converges to a single value for a large $J^{\max}$ is confirmed by the results in Figure 5.8. Because Algorithm 5 is probabilistically complete, we can also deduce that, for environment E1, the explorable region's volume (i.e. $|\mathcal{W}'(\mathbf{q}_0)|$) is approximately equal 0.5 cubic meter.

5.7 Conclusion

This work formulates the energy-optimal exploration problem (EEP). It is expressed as a minimization of the volume explored. The energy function is used to form an additional constraint. This formulation of EEP allows a direct comparison of exploration

algorithms. The energy constraint limit can be increased until the volume explored converges to a single value, for at least one of the algorithms used in the comparison.

We also show that EEP is inherently difficult to solve by proving that it is at least in PSPACE-hard. A probabilistically complete approach is presented. The algorithm typically chooses navigation paths that increase the robot's knowledge of its environment while expending minimal motor torques, thus reducing mechanical energy. It is compared in a simulated environment to other exploration approaches. The results confirm the validity of our formulation of EEP and show the importance of optimizing energy.

Chapter 6

Conclusion

6.1 Summary

Chapter 3 examines the problem of finding low-cost collision-free continuous paths to move a robotic system from an initial state to a goal state, where cost is measured as energy dissipated in the robot's actuators. The problem is known as the energy-optimal planning problem. The chapter introduces a cost approximation method that quantifies the amount of energy expended to move a robot along a path. The method is called integral-squared torque approximation (ISTA). Experiments performed in simulation show that there is a high correlation between the approximated cost and the true cost. The energy-optimal planning problem with zero velocities at end points is addressed by combining ISTA with the existing motion planner RRT*. Using a simulated six-degree-of-freedom manipulator robot, ISTA is shown to improve the solution of RRT* when compared to the Euclidean distance metric.

Chapter 4 examines the coverage problem in robotics and provides a generalized formulation. The proposed formulation is applicable to many types of robotic systems operating in heterogeneous environments. Furthermore, we prove that the generalized coverage problem (GCP) is polynomial-space hard by showing that the motion planning

problem is polynomial-time reducible to GCP. From this proof, it can be conjectured that the exploration problem is also polynomial-space hard. Finally, we provide a probabilistically complete algorithm that finds a continuous and collision-free path covering a goal region, while minimizing a cost functional.

Chapter 5 focuses on the problem of exploring a three-dimensional world with articulated robots. The work seeks to define the energy-optimal exploration problem and formulate it as a multi-optimization problem. The formulation helps in understanding the exploration objectives and can be directly applied to evaluate and compare different algorithms. Additionally, a novel approach that seeks to address this problem is introduce. The presented algorithm, evaluated in simulation, typically chooses navigation paths that increase the robot's knowledge of its environment while expending minimal motor torques.

6.2 Recommendations

6.2.1 Analytical characterization of ISTA

Future work includes an analytical characterization of the integral-squared torque method (ISTA) that would help support the experimental results obtained from Chapter 3. The robustness of ISTA can also be improved by generating trajectories (during the first step of the method) using alternative methods. For example, Lee et al. (2005) suggests the use of B-spline polynomials to express the motions of multibody systems.

6.2.2 Time-optimal motion and exploration

Fast motion is essential in increasing the productivity of robots. It can have an important impact on robot exploration, e.g. when robots are deployed in disaster areas, it

is crucial that information is obtained as fast as possible. Chapter 3 develops a method to find low-energy collision-free paths.

A natural extension to this work is to apply similar methods to the *time-optimal planning* problem, where the goal is to find a minimum-time continuous motion, while being subjected to actuators limits. There exists previous work on this problem (Choset et al. 2005; Verscheure et al. 2009). Future work in time-optimal planning will aim at reducing the computational cost of finding a solution, or minimizing time and energy simultaneously.

6.2.3 Approximate solution for the coverage problem

Chapter 4 proposes an algorithm (Algorithm 2) for the generalized coverage problem (GCP). The algorithm does not give a performance guarantee with respect to the cost of the path returned. Future work includes finding a ratio bound $\rho(n)$ for Algorithm 2 (or a similar algorithm to GCP) i.e. an algorithm that produces a solution whose value is within a factor of $\rho(n)$ of the value of the optimal solution. This can be achieved by the use of simplifying yet realistic assumptions. Wang et al. (2007) provide a polylogarithmic approximation algorithm to GCP by making the assumptions that there exists a finite number of sensor configurations that can be used in the coverage process, and that there is a given cost associated to each pair of sensor configurations.

6.2.4 Exploration with other robotic systems

The work on minimum-cost exploration and coverage can be extended by implementing the algorithms described in Chapters 4 and 5 on other popular robotic systems, e.g. a manipulator arm mounted on a mobile robot. Future work will include experiments with real robots in real environments to validate that it is as efficient in human-centric environments on mobile manipulators with arbitrary sensory placements (i.e. on the head, at the shoulder, etc.) as it is in pathological, simulated environments.

Bibliography

Acar E, Choset H, Zhang Y, Schervish M (2003) Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods. The International Journal of Robotics Research 22(7-8):441–466

Angeles J (2003) Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms. Springer-Verlag, Berlin

Asadpour A, Goemans MX, Madry A, Gharan SO, Saberi A (2010) An $O(\log n/\log\log n)$-approximation algorithm for the asymmetric traveling salesman problem. In: 21st Annual ACM-SIAM Symposium on Discrete Algorithms, pp 379–389

Atkar PN, Greenfield A, Conner DC, Choset H, Rizzi AA (2005) Uniform coverage of automotive surface patches. International Journal of Robotics Research 24(11):883–898

Berenson D (2011) Addressing cost-space chasms in manipulation planning. In: IEEE International Conference on Robotics and Automation, pp 4561–4568

Breitenmoser A, Metzger JC, Siegwart R, Rus D (2010) Distributed coverage control on surfaces in 3d space. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 5569–5576

Canny JF (1988) The Complexity of Robot Motion Planning. MIT Press, Cambridge, MA

Chachuat B (2007) Nonlinear and Dynamic Optimization: From Theory to Practice. Automatic Control Laboratory, École Polytechnique Fédérale de Lausanne, Switzerland

Cheng P, Keller J, Kumar V (2008) Time-optimal uav trajectory planning for 3d urban structure coverage. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 2750–2757

Chevallereau C, Grizzle JW, Shih CL (2009) Asymptotically stable walking of a five-link underactuated 3-d bipedal robot. IEEE Transactions on Robotics 25(1):37–50

Choset H (2001) Coverage for robotics - a survey of recent results. Annals of Mathematics and Artificial Intelligence 31:113–126

Choset H, Lynch KM, Hutchinson S, Kantor G, Burgard W, Kavraki LE, Thrun S (2005) Principles of Robot Motion: Theory, Algorithms, and Implementations. The MIT Press, Cambridge, Massachusetts

Christofides N (1976) Worst-case analysis of a new heuristic for the traveling salesman problem. Tech. rep., Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA

Danner T, Kavraki EE (2000) Randomized planning for short inspection paths. In: IEEE International Conference on Robotics and Automation, pp 971–976

Dudek G, Jenkin M, Milios E, Wilkes D (1991) Robotic exploration as graph construction. IEEE Transactions on Robotics and Automation 7(6):859–865

Englot B, Hover F (2011) Planning complex inspection tasks using redundant roadmaps. In: International Symposium on Robotics Research

Englot B, Hover F (2012) Sampling-based coverage path planning for inspection of complex structures. In: International Conference on Automated Planning and Scheduling, pp 29–37

Featherstone R (1987) Robot Dynamics Algorithms. Kluwer Academic Publishers, Boston/Dordrecht/Lancaster

Featherstone R (2007) Rigid Body Dynamics Algorithms. Springer, Boston

Ferguson D (2006) Anytime RRTs. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 369–5375

Frazzoli E, Dahleh MA, Feron E (2002) Real-time motion planning for agile autonomous vehicles. Journal of Guidance, Control, and Dynamics 25(1):116–129

Freda L, Oriolo G, Vecchioli F (2008) Sensor-based exploration for general robotic systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 2157–2164

Freda L, Oriolo G, Vecchioli F (2009) An exploration method for general robotic systems equipped with multiple sensors. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 5076–5082

Frieze AM, Galbiati G, Maffioli F (1982) On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. Networks 12(1):23–39

Gabriely Y, Rimon E (2001) Spanning-tree based coverage of continuous areas by a mobile robot. In: IEEE International Conference on Robotics and Automation, pp 1927–1933

Galicki M (1998) The planning of robotic optimal motions in the presence of obstacles. The International Journal of Robotics Research 17(3):248–259

Galicki M (2000) Time-optimal controls of kinematically redundant manipulators with geometric constraints. IEEE Transactions on Robotics 16(1):89–93

Gonzalez-Banos HH, Latombe JC (2002) Navigation strategies for exploring indoor environments. International Journal of Robotics Research 21(10-11):829–848

Gregory J, Olivares A, Staffetti E (2012) Energy-optimal trajectory planning for robot manipulators with holonomic constraints. Systems & Control Letters 61(2):279–291

Haimes YY, Lasdon LS, Wismer DA (1971) On a bicriterion formulation of the problems of integrated system identification and system optimization. IEEE Transactions on Systems, Man, and Cybernetics pp 296–297

Haq A, Aoustin Y, Chevallereau C (2012) Effects of knee locking and passive joint stiffness on energy consumption of a seven-link planar biped. In: IEEE International Conference on Robotics and Automation, pp 870–876

Hess J, Tipaldi GD, Burgard W (2012) Null space optimization for effective coverage of 3d surfaces using redundant manipulators. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 1923–1928

Holz D, Basilico N, Amigoni F, Behnke S (2010) Evaluating the efficiency of frontier-based exploration strategies. In: International Symposium on Robotics and German Conference on Robotics, pp 36–43

Hover FS, Eustice RM, Kim A, Englot B, Johannsson H, Kaess M, Leonard JJ (2012) Advanced perception, navigation and planning for autonomous in-water ship hull inspection. The International Journal of Robotics Research 31(1):1445–1464

Jaillet L, Cortes J, J, Simeon T (2010) Sampling-based path planning on configuration-space costmaps. IEEE Transactions on Robotics 26(4):635–646

Karaman S (2010) Optimal kinodynamic motion planning using incremental sampling-based methods. In: 49th IEEE Conference on Decision and Control, pp 7681–7687

Karaman S, Frazzoli E (2010) Incremental sampling-based algorithms for optimal motion planning. In: Robotics: Science and Systems

Karaman S, Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. International Journal of Robotics Research 30(7):846–894

Kavraki LE, Svestka P, Latombe JC, Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation 12(4):566–580

Kruse E, Gutsche R, Wahl F (1996) Effective iterative sensor based 3-d map building using rating functions in configuration space. In: IEEE International Conference on Robotics and Automation, pp 1067–1072

Kuffner JJ, LaValle SM (2000) RRT-connect: an efficient approach to single-query path planning. In: IEEE International Conference on Robotics and Automation, pp 995–1001

Kuipers B, Byun YT (1991) A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. Robotics and Autonomous Systems 8(1-2):47–63

LaValle SM (1998) Rapidly-exploring random trees: a new tool for path planning. Tech. rep., Computer Science Department, Iowa State University

LaValle SM (2006) Planning algorithms. Cambridge University Press, UK

LaValle SM, Kuffner JJ (1999) Randomized kinodynamic planning. In: IEEE International Conference on Robotics and Automation, pp 473–479

Lee SH, Kim J, Park FC, Kim M, Bobrow JE (2005) Newton-type algorithms for dynamics-based robot movement optimization. IEEE Transactions on Robotics 21(4):657–667

Makarenko AA, Wiliams SB, Bourgault F (2002) An experiment in integrated exploration. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 534–539

Marler RT, Arora JS (2004) Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization 26(6):369–395

Mei Y, Lu YH, Hu YC, Lee CSG (2004) Energy-efficient motion planning for mobile robots. In: IEEE International Conference on Robotics and Automation, vol 5, pp 4344–4349

Mei Y, Lu YH, Lee CSG, Hu YC (2006) Energy-efficient mobile robot exploration. In: IEEE International Conference on Robotics and Automation, pp 505–511

Nagatani K, Okada Y, Tokunaga N, Kiribayashi S, Yoshida K, Ohno K, Takeuchi E, Tadokoro S, Akiyama H, Noda I, Yoshida T, Koyanagi E (2011) Multirobot exploration for search and rescue missions: a report on map building in robocuprescue 2009. Journal of Field Robotics 28(3):373–387

NASA (2013) Mars exploration rover mission. http://marsrover.nasa.gov/overview/, accessed: 2013-06-15

Perez A, Platt R, Konidaris G, Kaelbling LP, Lozano-Pérez T (2012) LQR-RRT*: optimal sampling-based motion planning with automatically derived extension heuristics. In: IEEE International Conference on Robotics and Automation, pp 2537–2542

Reif JH (1979) Complexity of the mover's problem and generalizations. In: 20th Annual Symposium on Foundations of Computer Science, pp 421–427

Renton P, Greenspan M, Elmaraghy H, Zghal H (1999) Plan-n-scan: a robotic system for collision free autonomous exploration and workspace mapping. Journal of Intelligent and Robotic Systems 24(3)

Rusu RB, Sucan IA, Gerkey B, Chitta S, Beetz M, , Kavraki LE (2009) Real-time perception guided motion planning for a personal robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 4245–4252

Scott WR, Roth G, Rivest JF (2001) View planning as a set covering problem. Tech. rep., National Research Council of Canada

Shiller Z (1994) Time-energy optimal control of articulated systems with geometric path constraints. In: IEEE International Conference on Robotics and Automation, pp 2680–2685

Siciliano B, Sciavicco L, Villani L, Oriolo G (2009) Robotics: Modeling, Planning and Control. Springer, London, UK

Spong MW, Hutchinson S, Vidyasagar M (2006) Robot Modeling and Control. John Wiley and Sons, New York

von Stryk O, Schlemmer M (1994) Optimal control of the industrial robot manutec r3. In: Computational Optimal Control, Birkhauser Verlag Basel, Switzerland, pp 367–382

Sucan IA, Kalakrishnan M, Chitta S (2010) Combining planning techniques for manipulation using realtime perception. In: IEEE International Conference on Robotics and Automation, pp 2895–2901

Thrun S (2003) Robotic mapping: a survey. In: Exploring Artificial Intelligence in the New Millenium, Morgan Kaufmann, San Francisco, CA

Thrun S, Burgard W, Fox D (2005) Probabilistic Robotics. The MIT Press, Cambridge, MA

Torabi L, Gupta K (2012) An autonomous six-dof eye-in-hand system for in situ 3d object modeling. International Journal of Robotics Research 31(1):82–100

Urmson CP (2003) Approaches for heuristically biasing RRT growth. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 1178–1183

Vazirani V (2001) Approximation Algorithms. Springer-Verlag, Berlin, Germany

Verscheure D, Demeulenaere B, Swevers J, Schutter JD, Diehl M (2008) Time-energy optimal path tracking for robots: a numerically efficient optimization approach. In: 10th IEEE International Workshop on Advanced Motion Control, pp 727–732

Verscheure D, Diehl M, Schutter JD, Swevers J (2009) On-line time-optimal path tracking for robots. In: IEEE International Conference onRobotics and Automation, pp 599–605

Wallgrün JO (2010) Hierarchical Voronoi Graphs - Spatial Representation and Reasoning for Mobile Robots. Springer

Wang G, Irwin MJ, Berman P, Fu H, Porta TFL (2005) Optimizing sensor movement planning for energy efficiency. In: International Symposium on Low Power Electronics and Design, pp 215–220

Wang P, Gupta K (2007) View planning for exploration via maximal expected entropy reduction for robot mounted range sensors. Advanced Robotics 21(7):771–792

Wang P, Krishnamurti R, Gupta K (2007) View planning problem with combined view and traveling cost. In: IEEE International Conference on Robotics and Automation, pp 711–716

Washington R, Golden K, Bresina J, Smith DE, Anderson C, Smith T (1999) Autonomous rovers for mars exploration. In: Aerospace Conference, 1999. Proceedings. 1999 IEEE, vol 1, pp 237–251

Whaite P, Ferrie FP (1997) Autonomous exploration: driven by uncertainty. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(3):193–205

Xu A, Viriyasuthee C, Rekleitis I (2011) Optimal complete terrain coverage using an unmanned aerial vehicle. In: IEEE International Conference on Robotics and Automation, pp 2513–2519

Xu L, Stentz A (2011) An efficient algorithm for environmental coverage with multiple robots. In: IEEE International Conference on Robotics and Automation, pp 4950–4955

Yamauchi B (1997) A frontier-based approach for autonomous exploration. In: IEEE International Symposium on Computational Intelligence in Robotics and Automation, pp 146–151

Yu Y, Gupta K (2000) Sensor-based probabilistic roadmaps: experiments with an eye-in-hand system. Advanced Robotics 14(6):515–536

Yu Y, Gupta K (2004) C-space entropy: a measure for view planning and exploration for general robot-sensor systems in unknown environments. International Journal of Robotics Research 23(12):1197–1223