Helsinki University of Technology
Industrial Information Technology Laboratory Publications
Teknillisen korkeakoulun teollisuuden tietotekniikan laboratorion julkaisuja
Espoo 2003 TKK-INIT-1

ADAPTIVE PROBABILISTIC ROADMAP CONSTRUCTION WITH MULTI-HEURISTIC LOCAL PLANNING

Pekka Isto

Dissertation for the degree of Doctor of Philosophy to be presented with due permission of the Department of Computer Science and Engineering for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 26th of September, 2003, at 12 noon.

Helsinki University of Technology
Department of Computer Science and Engineering
Industrial IT Laboratory

Teknillinen korkeakoulu
Tietotekniikan osasto
Teollisuuden tietotekniikan laboratorio

Distribution:
Helsinki University of Technology
Industrial IT Laboratory
P.O. Box 9204
FIN-02015 HUT
Tel: +358-9-451 6231
Fax: +358-9-451 5351
Email: mpg-info@cs.hut.fi

ISBN: 951-22-6722-5 (printed version)
ISBN: 951-22-6723-3 (electronic version)
ISSN: 1459-6458

Otamedia Oy
Espoo 2003

ABSTRACT

The motion planning problem means the computation of a collision-free motion for a movable object among obstacles from the given initial placement to the given end placement. Efficient motion planning methods have many applications in many fields, such as robotics, computer aided design, and pharmacology. The problem is known to be PSPACE-hard. Because of the computational complexity, practical applications often use heuristic or incomplete algorithms. Probabilistic roadmap is a probabilistically complete motion planning method that has been an object of intensive study over the past years. The method is known to be susceptible to the problem of "narrow passages": Finding a motion that passes a narrow, winding tunnel can be very expensive.

This thesis presents a probabilistic roadmap method that addresses the narrow passage problem with a local planner based on heuristic search. The algorithm is suitable for planning motions for rigid bodies and articulated robots including multi-robot systems with many degrees-of-freedom. Variants of the algorithm are described for single query planning, single query planning on a distributed memory parallel computer, and a preprocessing type of learning algorithm for multiple query planning.

An empirical study of the effect of balance between local and global planning reveals that no universal optimal balance is likely to exist. Furthermore, it appears that many traditional simple local planners are too weak for efficient solving of problems with a narrow passage. The empirical results show that local planners based on backtracking search are more efficient than the more traditional local planners when a motion through a narrow passage is to be planned. The parallel variant has acceptable scalability on a parallel computer built from commodity components. It is also observed that run-time adjustment of the parameters of the search can reduce the variance of the run-cost. The run-cost variance is a known, but little studied deficiency of randomized motion planning methods. It is suggested that the future research in randomized motion planning algorithms should address run-cost variance as an important performance characteristic of the algorithm.

The results are obtained with empirical methods and established procedures from design and analysis of experiments. The algorithms are assessed with a number of test problems including known benchmark problems from the literature.

Keywords: Motion planning, robotics, heuristic algorithms, empirical algorithmics

PREFACE

The beginnings of this thesis are in the research projects of the Robotics Group at the Laboratory of Information Processing Science and TAI Research Centre of Helsinki University of Technology. The group has since evolved to be the Industrial Information Technology Laboratory, and this thesis was completed as a part of the research activities of the laboratory. I am greatly indebted to my supervisor and the head of the laboratory, Professor Juha Tuominen for introducing me to robotics and the motion planning problem, and setting me on the path that has lead to this thesis. Professor Martti Mäntylä provided early supervision of this work, and his support and guidance during all phases has been instrumental in making this thesis possible.

Many members of the Robotics Group influenced and supported this work. Mr. Antti Autere inspired the search-based approach to motion planning problem taken in this study. His constructive criticism and insistence on proofs and evidence as hallmarks of scientific research have been very valuable in developing the skills needed for completing this thesis. Mr. Johannes Lehtinen, Mr. Pasi Eronen and the IMPACT group of Tik-76.115 Software Project under the guidance of Mr. Andy Nurminen provided software that has supported this research. Mr. Lehtinen also provided the geometric model for the Hwang and Ahuja benchmark problem. The Parasol laboratory at Texas A&M University provided the Alpha Puzzle benchmark problem.

This research is computational in nature and would not have been possible without the support of Mr. Janne Ravantti who provided access to the cluster computer at University of Helsinki. Mr. Olli Seppälä and Mr. Jan Nyman provided local computational resources at the laboratory. Laboratory secretaries Kirsi Huhn and Mirva Piispa provided valuable help in arranging the printing and the defence.

I am grateful to my pre-examiners Professor Kamal K. Gupta and Dr. Thierry Siméon for their time and expertise. Their comments improved the manuscript by helping me to be more explicit and precise on several issues and pointing out omissions.

The funding for this work has come from numerous sources. Laboratory of Information Processing Science, European Commission and the Academy of Finland provided the funding during the Robotics Group years. The thesis research has been funded by Helsinki Graduate School in Computer Science and Engineering, Helsinki University of Technology, Finnish Cultural Foundation, and the Industrial Information Technology Laboratory. All financial support is gratefully acknowledged.

I would like to thank my family and friends for supporting and believing in me during all phases of my life including the ups and downs of this thesis project.

Espoo, September 2003.

This thesis consists of an overview and the following publications:

Paper I: P. Isto, Path Planning by Multiheuristic Search via Subgoals, Proceedings of the 27th International Symposium on Industrial Robots, CEU, Milan, 712-726.

Paper II: P. Isto, A Two-level Search Algorithm for Motion Planning, Proceedings of the 1997 IEEE International Conference on Robotics and Automation, IEEE Press, 2025-2031.

Paper III: P. Isto, A Parallel Motion Planner for Systems with Many Degrees of Freedom, Proceedings of the 2001 International Conference on Advanced Robotics, 339-344.

Paper IV: P. Isto, Constructing Probabilistic Roadmaps with Powerful Local Planning and Path Optimization, Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE Press, 2323-2328.

Paper V: P. Isto, J. Tuominen, and M. Mäntylä, Adaptive Strategies for Probabilistic Roadmap Construction, Proceedings of the 2003 International Conference on Advanced Robotics, 682-687.

Paper VI: P. Isto, M. Mäntylä, and J. Tuominen, On Addressing the Run-Cost Variance in Randomized Motion Planners, Proceedings of the 2003 IEEE International Conference on Robotics and Automation, IEEE Press, 2934-2939.

TABLE OF CONTENTS

1. INTRODUCTION

Computation of collision-free motions for a movable object among obstacles is an important but computationally hard problem. An example of an application for collision-free motion generation is computer animation, where realism requires that moving objects do not pass through any solid obstacles. It is of course possible to produce the motions manually and check them for collisions. If collisions are detected, motions must be modified and rechecked. The process becomes iterative and can be very tedious and time-consuming. It would often be desirable to be able to describe the required motions at a higher level and let the animation system to insert collision avoidance maneuvers when necessary. Other relevant applications for motion planning algorithms come from numerous fields, such as robotics, mechanical engineering, and pharmacology. The problems from these fields are somewhat different in nature, and thus, many different variations of the motion planning problem exist. The fully defined problems must specify details, such as the kinematic structure of the movable object, location and shape of the obstacles, and the certainty of the obstacle information available during the planning stage. Several such variations are described in more detail in the chapter 3.

The most basic variation of the motion planning problem considers a single free-flying object among completely known and static obstacles. The problem is to plan a collision-free motion for the movable object from a given initial position and orientation to a given goal position and orientation. A more precise definition is presented later in this thesis. The motivation for this thesis is the importance of practical motion planning methods and their potential benefit in many fields. A particular inspiration and motivation has been model-based gross motion planning in robotics. A typical robotic system considered here is a manipulator arm. Such arm consists of multiple rigid links connected with joints. A solution for the motion planning problem for an arm is a sequence of joint values from the start position to the goal position. A full control system for a robotic arm must address many other issues in addition to ensuring collision-free motions. Topics such as force control, trajectory planning and task planning are important for a robotic system, but not considered here.

The objective of this research is to construct a method for solving model-based gross motion planning problems. In order to achieve the objective, several algorithms are designed and evaluated empirically. The algorithms are suitable for planning motions for systems with arbitrary number and types of degrees-of-freedom in a completely and accurately known static environment. Only kinematic constrains are considered. The motion planning problem is addressed as a search problem, and the presented algorithm searches the solution in the joint space of the robot or other movable object. The search is performed via point subgoals in the joint space with an A* based local planner. The search algorithm has two components, a global planner and a local planner. A global planner places the subgoals, controls the capability of the local planner, calls the local planner to attempt local planning between the start position, subgoal positions and the goal position, and stores the successful path segments. Once a sequence of path segments from start to goal is available, they are concatenated to a complete solution by the global planner. Thus, the algorithm has two-levels: a local level that tries to produce path segments in the joint space, and a global level that generates prospective path segments, controls the local planner and forms the solution to the motion planning task. Since the planner uses randomly generated point subgoals or samples of the search space and uses a graph to represent the connectivity between these samples, it belongs to a class of motion planners that has become to be known as probabilistic roadmap planners. This class of planners is known to be probabilistically complete.

This thesis makes several contributions:

- An efficient and effective motion planning algorithm is presented. Variations for single query planning and multiple query planning are presented. Testing with well known benchmark problems shows that the presented algorithms make improvements over previous results. Novel features of the algorithm are the use of powerful local planner in conjunction with the probabilistic roadmap method, run-time adaptation of the local planner, and a set of efficient search heuristics for the local planner. The algorithm can be parallelized easily, and it demonstrates acceptable speed-up on commodity parallel hardware. These results were published in papers I, II, III, and IV.
- An empirical study of the effect of balance between local and global planning reveals that no universal optimal balance is likely to exist. Furthermore, it appears that many traditional simple local planners are too weak for efficient solving of problems with a "narrow passage". These results were published in papers II, IV, and V.
- A contribution to the methodology of motion planning research is made by introducing the application of rigorous statistical techniques for performance assessment. The methodology was used to obtain the results of paper IV.
- Finally, this thesis proposes that the future improvements in randomized motion planning algorithms should address run-cost variance as an important performance characteristic of the algorithm. Additionally, it is shown empirically that run-time adaptation of search parameters can be used as a variance reduction technique. These results were published in paper VI.

This work is primarily constructive in nature: the aim is to provide a practicable solution method to a hard algorithmic problem with important real-life applications. Since the method is heuristic, its properties are studied empirically. The method has been developed iteratively. A number of design decisions have been made along the way to the last tested version. These decisions have been made based on various experiments. While the generation of the alternatives is largely an intuitive and creative process, their properties and relative merits can be studied with established empirical methods. This process yields a body of knowledge about the role and importance of the various components of the method.

This research will produce both normative and descriptive knowledge and will yield recommendations for successful methods of motion planning. The recommendations are justified by describing probable mechanisms for the success. Although it is arguable whether this body of knowledge forms a systematic theory, it certainly increases understanding of the problem and the techniques available for solving it. The methods and instantiations described in this thesis are intended for general solving of the motion planning problem, not for solving a particular class of problems emerging from a specific application. Thus, this research is basic research for principles rather than applied research for developing applications. However, the research has been carried out in the context of robotics, and this thesis presents the results in that context. For the sake of motivation, a number of other applications for motion planning methods are presented in this thesis.

The next chapter describes the philosophical and methodological positioning of the research presented in this thesis. Out of the three paradigms of computer science, theory, abstraction, and design, this research employs design and abstraction. Chapters 3 and 4 introduce the subject of this research. Chapter 3 describes the research problem addressed in this thesis: the motion planning problem. The chapter covers variations of the motion planning problem, various complexity results, and known applications. Chapter 4 reviews the previous research in motion planning. Since the amount of related literature is considerable, the chapter presents only an overview of the major algorithmic approaches to motion planning and the background to the motion planning algorithm presented in this thesis.

Chapters 5-8 present the contribution made in this thesis. Chapter 5 presents the design part of this research. The chapter covers the requirements set for the motion

planner and describes the motion planning algorithm variants studied in this thesis. Chapters 6 and 7 present the abstraction part of the research. The chapters describe the experimental set-up for this research and the empirical results. Some of the observations are abstracted into focused hypotheses and tested for significance to identify essential properties of the motion planning algorithm. Chapter 8 discusses the findings, and the last chapter presents the conclusions.

It should be noted that according to the nature of an article dissertation, the presentation in chapters 5-9 is a compendium of the appended publications summarizing the goals, techniques and discoveries of the research rather than a full reproduction of the publications. The reading of the appended publications is necessary for the complete understanding of the research presented in this dissertation.

2. ON SCIENCE, RESEARCH AND METHOD

A doctoral thesis is expected to make a scientific contribution. It is therefore necessary to discuss how this thesis fulfills the criteria set forth for scientific knowledge. Science is systematic and rational inquiry for new knowledge. (Haaparanta and Niiniluoto 1986, p. 7) This statement presents science as a process that has a particular goal and that must fulfill certain requirements. The goal is to expand the knowledge about the World. The process becomes systematic, when the inquiry is organized through particular social institutions, such as universities and research centers and the results of the inquiry are compiled into broad knowledge systems. The requirement for rationality constraints the thinking and the methods that are used to obtain new knowledge: The validation of knowledge cannot rely on intuition or authority, but on a method that has been approved by the scientific community.

Extremely simply put, science is the activity that is performed by the members of the scientific community. But as Haaparanta and Niiniluoto (1986, p. 8) point out, this definition leads easily to a circular definition. It is necessary to analyze what are the characteristics of a method that could be a candidate for becoming approved by the scientific community, and thus, a method that can be scientific. Haaparanta and Niiniluoto (1986, p. 13-17) list characteristics of a scientific method. It must be objective, critical, autonomic and progressive. Objectivity requires that the results of the research correspond with the properties of the research object and are independent of the opinions of the scientist. Objectivity can be increased by public presentation of the results and their justification, and critique of the presentation by the scientific community. A scientific method is autonomic, when the critique of the results is based solely on the truthfulness of the results, not on political, religious, or moral grounds. Progressive implies not only that the amount of knowledge increases in time, but also that untruthful hypotheses and theories are replaced by more truthful ones. A scientific method must be self-repairing as not to mislead scientists irreversibly, but causing untruthful propositions to be replaced with truthful ones.

Multiple motivations for inquiring for new knowledge exist. Niiniluoto (1980) lists several knowledge interests with accompanying goals of research and functions of knowledge. One may have a veristic interest, search for truth, and try to explain the World. Technical interest is related to the will to control nature through prediction. Hermeneutic interest seeks to communicate and interpret tradition through understanding. Emancipatory interest seeks liberation from false cognizance by critique of ideology. Inquiry for explanation and understanding yields descriptions of the world while prediction and critique enable and motivate the control of world. The goal of science can be said to be statements that describe the states of affairs in the world. Such view of science is known as cognovist (Haaparanta and Niiniluoto 1986, p. 9). The resulting knowledge is often said to be **descriptive** (e.g. March and Smith 1995). A different view is to see science as problems to be solved and the goal of science to produce prescriptions for solving those problems rather than knowledge as statements. This view is known as behaviorism (Haaparanta and Niiniluoto 1986, p. 10). The results here are also knowledge, namely knowledge about successful (and unsuccessful) prescriptions for solving the problems. Such knowledge is sometimes called prescriptive or **normative** knowledge.

Another often made distinction within science is to divide it to basic science and applied science. Much controversy and science politics can emerge from this divide. The goal of **basic science** is the knowledge itself, without any immediate application, while **applied science** usually has also some other useful goal (Järvenpää and Kosonen 1997). March and Smith (1995) point out that the essential difference between basic and applied science is the research intent: Is the primary objective to produce new knowledge, or is it to construct some practical application. The *"two species of scientific activity"* have also significant interactions (March and Smith 1995). Basic science

produces theories and other forms of knowledge for applied science to consume. The applications provide tests for the underlying theories, and as the theories prove to be unsatisfactory, challenges for basic science.

In addition to defining science to be a particular type of human activity, it can also be taken as a reference to the body of knowledge accumulated by scientific study (Haaparanta and Niiniluoto 1986, p. 8). This body of knowledge is available in various types of scientific publications and databases. Science can also refer to the social institutions that are the organizational setting for scientific work. In this meaning science refers to scientists, scientific institutions, and all the resources, administrative and political systems that are necessary in order to produce, disseminate and utilize scientific knowledge.

A scientific discipline can be characterized by the object of research and the methods of research. Newell, Perlis, and Simon (1967) define computer science as the study of computers and phenomena surrounding computers. Their object of research is "*living computer*", which means "*the hardware, their programs or algorithms, and all that goes with them*". Newell, Perlis, and Simon state that the phenomena of computers are not subsumed under any one existing science; therefore, a distinct discipline is needed to describe and explain those phenomena. Even though computers are artificial objects, Newell, Perlis, and Simon model computer science after natural sciences. They acknowledge that computers belong also to engineering, but leave open the professional specialization between analysis and synthesis, and between pure study of computers and their application.

This specialization has never occurred, but computer science and computer engineering have remained inseparable. The Association for Computing Machinery Task Force on the Core of Computer Science concluded that no fundamental difference exists between these two fields in the core material (Denning et al. 1989). The task force uses the phrase *discipline of computing* to embrace all of computer science and engineering. Their short definition of the discipline states: "*The discipline of computing is the systematic study of algorithmic processes that describe and transform information: their theory, analysis, design, efficiency, implementation, and application. The fundamental question underlying all of computing is 'What can be (efficiently) automated?'* ".

The task force recognizes three major paradigms within the discipline: theory, abstraction, and design. Each has a distinct process for operation and a particular outcome from that process. The paradigm of theory is rooted in mathematics and iterates the following steps to develop a coherent, valid **theory**: "*(1) characterize objects of study (definition); (2) hypothesize possible relationships among them (theorem); (3) determine whether the relationships are true (proof); (4) interpret results.*" The process of **abstraction** is rooted in the experimental scientific method and it iterates the following steps in the investigation of a phenomenon in order to build a model: "*(1) form a hypothesis; (2) construct a model and make a prediction; (3) design an experiment and collect data; (4) analyze results.*" The third paradigm is **design**. It is rooted in engineering and iterates the following steps to construct a system or a device in order to solve a problem: "*(1) state requirements; (2) state specifications; (3) design and implement the system; (4) test the system.*"

These three paradigms and associated processes are intertwined in the discipline of computing, and the discipline "*sits at the crossroads among the central processes of applied mathematics, science, and engineering*" (Denning et al. 1989). However, each of these paradigms represents separate areas of competence (Denning et al. 1989). Individual researchers tend to develop their skills predominantly in one of these areas. This is a source of much controversy: It is not uncommon for individual computer scientists to argue for the superiority of one of these areas, even if only for themselves. One just has to study ACM Turing Award lectures to see how prominent computer

scientists position themselves in theory (Cook 1983), abstraction (Newell and Simon 1976) or design (Brooks Jr. 1996[1]).

March and Smith present an integrated framework for research in information technology (1995). The main difference between information technology (IT) research and computer science is that unlike computer science, IT research is not restricted to algorithmic processes, but takes account of the people and organizations involved in the information processes. Thus, their research framework should be equally valid for computer science. If we accept that mathematics is subsumed by natural sciences (an idiosyncrasy, according to Newell, Perlis, and Simon 1967), the framework is indeed highly compatible to paradigms and processes laid out by the ACM Task Force on the Core of Computer Science. The theory paradigm of ACM task force can be seen to be subsumed by the natural science of the framework with mathematical proof as a specific form of justification for the research output.

The framework divides research efforts into sixteen subtypes by research outputs and the research activities. (March and Smith 1995) The outputs or artifacts are constructs, models, methods, and instantiations. The activities are building, evaluating, theorizing and justifying. Constructs or concepts constitute a conceptualization for a domain. A model is a set of statements expressing relationships among the constructs. A method is a set of steps or an algorithm used to perform a task, and an instantiation is the realization of an artifact in its environment. The build activity demonstrates an artifact for feasibility, while evaluation can reveal if any progress over the previous artifacts have been made. Theorizing attempts to explicate the characteristics of the artifacts and its interactions with the environment. Theories are justified with evidence and testing. According to March and Smith, the build and evaluate activities belong to design science, and theorize and justify activities to natural science.

In IT and computer science research, building the first artifact of any type in the framework is considered a contribution provided that the artifact has utility for an important task. (March and Smith 1995) Building the subsequent artifacts must be accompanied with evaluation activity, since the significance of the contribution comes from the ability to show some improvement. It is necessary to define what is being achieved with the artifact and how to measure that achievement. Thus, the development of metrics and measurement techniques becomes essential. Successful artifacts call for a theory explaining the reasons behind the success; the same is true also for unsuccessful artifacts. Formal theories can be justified with mathematical proofs, while non-mathematical theories are usually justified with data collection and analysis methods.

According to Haaparanta and Niiniluoto (1986, p. 11) inquiry becomes systematic and rational when it is done using some research method. Although it is debatable whether research should follow some defined methodology, it can be stated that doing so will help fulfill the criteria of scientific research. Järvenpää and Kosonen (1997, p. 6) define a research method to mean the practices and norms involved in acquiring and analyzing the research materials. A research method may also mean a larger whole, an approach that can consist of multiple measurement techniques (Järvenpää and Kosonen 1997, p. 6). A pragmatic view of methodology is that it provides researchers with a set of established procedures and tools for producing new knowledge. These procedures are at least tentatively accepted by the scientific community for use within a discipline. Using a known and accepted method will help in producing results which will pass the review and critique by the scientific community.

A large number of research methods exist (e.g. Järvenpää and Kosonen 1997; Järvinen 1999). The research objectives and questions guide the selection of appropriate research methods. The objective of this research is to construct a method for solving motion planning problems. This is obviously design science in the March

[1] This is his Newell Award address. Brooks received the 2000 Turing Award, see http://terra.cs.nps.navy.mil/DistanceEducation/online.siggraph.org/2001/SpecialSessions/2000TuringLecture-DesignOfDesign/session.html

and Smith framework with build as the primary activity. As will be discussed below, there exists a large amount of literature presenting numerous motion planning methods. Since the method presented in this thesis is not the first one, evaluation for improvement becomes essential. The objective of constructing a practical method implies that it is necessary not only to consider the method as research output, but also an instantiation of the method in its environment. The ACM task force conceptualization also suggests that the design paradigm is appropriate. This implies the need to state requirements and specifications prior to implementation and testing. If the instantiation of the method proves successful in testing and evaluation, it becomes desirable to consider natural science activities of theorizing and justifying in order to explain the observed benefits. Thus, concepts and models will become research outputs of this stage of research. The ACM task force abstraction process suggests that forming hypotheses and testing them experimentally is the appropriate course of action.

Implementation and experimentation, and taking programs as experiments have long roots in computer science (e.g. Newell and Simon 1976). Despite efforts to promote experimental research in computer science (e.g. Denning 1980, 1981), the use of empirical methods in research has remained relatively underdeveloped. There have been explicit efforts to improve the experimental methodology for example in operations research (e.g. Hooker 1994, 1995), and mathematical sciences (e.g. Crowder et al. 1978; Jackson et al. 1991). Within computer science, calls for better methodology have been made in algorithm research (e.g. Moret 2002), software engineering research, and artificial intelligence (Cohen 1995). There has been little use of empirical methodology in motion planning research to take the best advantage of the experimental data. Motion planning research is essentially research in algorithms, and therefore, lessons learned in experimental algorithm research are directly applicable.

The research methods for this study are borrowed from empirical artificial intelligence research and experimental algorithmics as well as related fields of operations research and mathematical programming. Cohen divides empirical methods into exploratory techniques and confirmatory procedures. (1995) Exploratory methods include visualization techniques and descriptive statistics among others. Confirmatory procedures are statistical methods for hypothesis testing and prediction. In performance assessment, exploratory methods can be used to demonstrate performance. Statistical testing may be used to estimate the significance of the observed performance difference between constructs. If performance is to be explained empirically, one must indulge into model building in order to obtain an empirical theory. Performing empirical research involves selection of the appropriate techniques and procedures. The discussion of the selections made in this research is postponed to the relevant chapters of this thesis.

Having discussed the types of research and modes of justification for the results that are relevant for this thesis, it is possible to define some terminology for the coming presentation. In the following discussion, if results or knowledge is obtained by constructing an artifact and demonstrating by experiments and exploratory methods that it possesses some stated desirable properties, it is said to have been **demonstrated** to be true. If some results are justified by a formal proof or a confirmatory procedure, they are said to have been **shown** to be true.

This chapter presented the research approach and methods selected for this thesis research. The selections were made based on the research objectives within the framework of ACM Task Force on the Core of Computer Science and that presented by March and Smith. The experimental method is influenced by the methodological developments in algorithmics and artificial intelligence research. It is acknowledged here that the same research problem can be as legimitively be addressed with other approaches and methods, as well. One may put emphasis on the evaluation of the method rather than its instantiation and ask for a formal analysis of the properties of the presented algorithm. At the other end of the spectrum, one may define the

environment of the instantiation to include an actual robot system within its work environment and ask for a field experiment.

It may be appropriate to conclude this chapter by presenting a concept that Gallos (1996) calls research identity. While it is difficult or perhaps impossible to give universal answers regarding science, research and methodology, each individual researcher has to demarcate an area of operation that fulfills his or her expectations of the scientific. Research identity covers scientist's answers and attitudes to the difficult questions regarding the precise nature of knowledge and truth, important research questions and preferred methods. It explains the goals and motivation for doing research. Of course, research identity is not a static or always very explicitly defined entity, but it is evolving and appears often multifaceted. Nevertheless, it helps to explain the choices made by individual researchers and the ways they position themselves in the wider scientific community. Perhaps one of the most important functions of a doctoral thesis is that it helps the writer to find and establish a research identity. This thesis describes the author's research identity or at least parts of it at one point in time.

3. MOTION PLANNING PROBLEMS AND APPLICATIONS

This chapter describes the scope of this thesis and presents definitions for the core terminology needed to describe the motion planning problem. A definition of the basic motion planning problem is presented and some extensions of it are described. The complexity of the problem is characterized. At the end of the chapter, some reported applications employing motion planners and experiences with them are presented. That section provides the motivation for research in a problem that is known to be computationally hard.

One of the most elementary capabilities of an autonomous robot is the ability to generate the motions needed to obtain some high levels goals (Latombe 1991, p. ix-x). This observation has motivated much research in robot motion planning. The approaches to robot motion planning can be roughly divided into two categories. (Gupta and del Pobil 1998) The classical motion planning, or model-based motion planning, assumes that the robot system has an explicit representation of the robot's environment. In sensor-based planning the environment is unknown and the robot is guided directly from the sensory input without constructing internal representation for the environment. In real robotic systems these approaches can be combined, and often are combined, since the tasks may involve contact with the environment. In such a situation, fine tolerances and force-feedback have to be considered. Those are issues that are not easy to incorporate into model-based approaches. This thesis considers solely the model-based gross motion planning problem.

Some definition of terminology is necessary to facilitate the fore-coming presentation. (Hwang and Ahuja 1992, p.221) Although the movable object can take many forms, it is often called a **robot** for brevity. A robot that consists of rigid links and revolute or prismatic joints connecting the links is called a **manipulator**. The robot operates in some physical space called the **work space** of the robot. Perhaps the one most important concept is the **configuration** of a robot. A configuration of a robot is a specification of the position of every point of the robot in its work space. The set of all configurations is the **configuration space** of the robot, also called the *cspace* for short. There exist alternative parameterizations for a configuration. The minimal number of parameters required to fully specify a configuration is the **degrees-of-freedom** (*dof*) of the robot. The **free space** refers to the parts of the work space not occupied by obstacles or parts of the *cspace* in which the robot does not collide with the obstacles. The latter is also sometimes called more precisely free *cspace*. A **path** is a curve in the *cspace*. It is used to represent a motion of the robot either as a mathematical expression or more often as a sequence of points along the curve. A configuration or a path is collision-free or **feasible** if it does not involve collisions with the obstacles for the robot. If time is assigned to the points along the curve, it is called a **trajectory**. The times signify the instants at which the robot assumes the configuration associated with that point. Each of these concepts can be given precise mathematical definitions (e.g. Latombe 1991), but such a formal development is not necessary for this thesis, and therefore, is omitted.

It should be noted that there is some terminological ambiguity in the literature. At times, one uses the term motion planning to refer to the process or task of generating trajectories rather than paths. More specific terms of path planning and trajectory planning can be used to make that distinction, if necessary. (Hwang and Ahuja 1992, p.221, 225) Path planning refers to planning the geometric and kinematic specifications of the motion for the robot, where as trajectory planning includes also the planning of velocities. A robot system must have capability for both path planning and trajectory planning along with a number of other capabilities, but these are probably best implemented separately and executed concurrently with interactions among the modules implementing the capabilities (Latombe 1991, p. 45-50).

The above concepts come together to form the **basic motion planning problem** (Latombe 1991, p. 5-7):

"Let A be a single rigid object - the robot - moving in a Euclidean space W, called workspace, represented as $\mathbf{R}^N$, with $N = 2$ or 3.

Let $B_1,...,B_e$ be fixed rigid objects distributed in W. The B_i's are called the obstacles.

Assume that both the geometry of A, $B_1,...,B_q$ and the locations of the B_i's in W are accurately known. Assume further that no kinematic constraints limit the motions of A (we say that A is a free-flying object).

The problem is: Given an initial position and orientation and a goal position and orientation of A in W, generate a path p specifying a continuous sequence of positions and orientations of A avoiding contact with the B_i's, starting at the initial position and orientation, and terminating at the goal position and orientation. Report failure if no such path exists."

The above problem is purely geometric planning problem for a rigid body. It is simplified, but nonetheless a hard problem. Many extensions exist, and many of the methods for solving the basic problem can be modified for solving the extensions. Several of the well-known extensions are listed here. (Latombe 1991, p. 22-32; Hwang and Ahuja 1992, p. 225) The above problem is **static**: all the obstacle information is available right at the beginning of the planning. In a **dynamic** variation, an increasing amount of obstacle information becomes available during the planning, for example because of interaction with sensing. When the problem has a changing environment or moving obstacles, it is a **time-varying** problem. When there are multiple robots, the problem is called **multimovers problem**. If objects can change shape, the problem is **conformable**. An important subclass of conformable problems is a robot consisting of multiple rigid objects connected with joints, typically sliding (prismatic) joints or hinges (revolute joints). Such robots are called articulated robots. A manipulator arm is perhaps the best known articulated robot. The robotic system may have inherent restrictions on its motions, which cause the planned motions to be **constrained**. Kinematic constraints restrict the motion of the robot or its parts. Articulated robots have the relative motions of the parts restricted by the joints between the parts. If the constraint can be removed by reparameterization of the configuration, they are called holonomic constraints. **Holonomic constraints** reduce the number of parameters needed to specify a configuration. Thus, they do not fundamentally change the nature of the problem, they just reduce the dimensionality of the *cspace*. A constraint that is a non-integrable equation involving the configuration parameters and their derivates (velocity parameters) is called a **nonholonomic constraint**. They require explicit handling, and thus, motion planning techniques developed for holonomic systems are not sufficient. If the obstacles are not known accurately, the **uncertainty** must be accounted for. Besides obstacle information, there may be uncertainty with respect to the robot's path following accuracy and sensing capabilities among others. The obstacles may be movable, and the robot may manipulate the obstacles to open passages. **Movable objects** complicate the problem with issues such as planning separate transit and transfer motions and determining stable grasps.

The computational complexity of the motion planning problem has hindered the development of practical motion planning algorithms (Hwang and Ahuja 1992, p. 219). In order to get some understanding of the complexity of the problem and motivation for the development of heuristic methods, it is necessary to present some results from the theoretical study of the motion planning problem. In theoretical analyses, the fact that the robot must be located in the free work space is represented by a collection of equalities and inequalities, which express that no feature of the robot touches or intersects with the features of the obstacles. (Schwartz and Sharir 1990) Typically these constraints are algebraic and of some maximal degree. The free *cspace* of a robot with k degrees-of-freedom is then represented as the subset of $\mathbf{R}^k$ defined by a

Boolean combination of these constrains. Such a set is called a semialgebraic set. The size of this representation is defined to be the number of the equalities and inequalities, and it is called the geometric or combinatorial complexity of the given instance of the motion planning problem. It is known that the combinatorial complexity of the free *cspace* of a robot with k degrees-of-freedom and constrained by n geometric constraints is $\Omega(n^k)$. But, one needs to compute only the connected component of free *cspace* that contains the initial placement of the robot. A recent result by Basu (2003) shows that under some natural geometric assumptions, the combinatorial complexity of single connected component is $O(n^{k1})$. However, as Schwartz and Sharir (1990) point out, one is interested in computing the path, not a representation of the free *cspace*. It is currently not known if this can be used to reduce the upper bound.

The theoretical analysis of the computational complexity of the motion planning problem can be characterized by presenting the theorem 40.1.3 from Sharir's survey article (1997):

> "Theorem 40.1.3 *Lower bounds*
>
> The motion planning problem, with arbitrary many degrees-of-freedom, is PSPACE-hard for the instances of: (a) coordinated motion of many rectangular boxes along a rectangular floor; (b) motion planning of a planar mechanical linkage with many links; and (c) motion planning for a multi-arm robot in a 3-dimensional polyhedral environment."

This theorem is a composite of earlier results from several authors. The first part is proved by Hopcroft, Schwartz and Sharir (1983). Joseph and Plantiga proved the second part (1985), and Reif (1979) proved the last part. A doubly exponential upper bound to the motion planning problem was obtained from the algorithm presented by Schwartz and Sharir (1983). An algorithm with a singly exponential upper bound was later presented by Canny (1988). Among the algorithms, which construct a representation for the whole free *cspace*, Canny's algorithm is near-optimal in the worst-case, since the free *cspace* can have exponentially many connected components (Sharir 1997).

Many of the extensions of the motion planning problem have also been analyzed, but extensions tend to further increase the complexity of the problem (Schwartz and Sharir 1990). There are a large number of exact and non-heuristic algorithms with provable worst-case complexities for various versions of the motion planning problem, but they are usually not suitable for practical implementations (Latombe 1999). For systems with more than a few degrees-of-freedom, the exact algorithms become very inefficient in practice (Sharir 1997, p. 749). These results suggest that, generally, approximate or heuristic methods must be used if practical implementations are needed. However, some of the best known theoretical algorithms are briefly described in the next chapter.

Despite the complexity of the problem, many important applications of motion planning methods motivate the study and development of practicable algorithms. In the introduction, this thesis motivated the research in motion planning with problems from graphical animation and robotics. Those are not at all the only possible applications for motion planning methods, but a number of application fields have been identified and demonstrated. Practical motion planning methods have a number of important applications in diverse fields such as robotics, mechanical engineering, computer graphics, and computational pharmacology. Nevertheless, robotics remains the most important motivation at this time. An autonomous or semi-autonomous robot system must be capable of generating collision-free motions for itself. Therefore, it must possess motion planning capability. Given the crucial importance of motion planning capability in the robotics, and consequently, the interest of the robotics research community toward developing motion planning techniques, it is unsurprising

that a large number of applications have been reported. Handey is an experimental robotic system with motion planning capability provided by one of the early *cspace* planners (Lozano-Pérez et al. 1987; Lozano-Pérez et al. 1992). Graux et al. (1992) describe experiences using the well-known Randomized Path Planner (RPP) in a real industrial environment for planning motions for robots riveting Airbus panels. An early version of the author's motion planner was used in an experimental car disassembly system called Neurobot (Tuominen et al. 1995). One of the industrial success stories is the use of AMROSE motion planning system at the Odense Steel Shipyard Ltd (Overgaard et al. 1998). The system is capable of automatically processing one-of-a-kind ship blocks. The SANDROS planner together with a fast geometry calculation library has been integrated with TELEGRIP robot off-line programming system (Watterberg et al. 1997). The system is said to be in daily use at Sandia National Laboratories.

The use of motion planning as a virtual prototyping tool was reported by Chang and Li (1995). They used RPP to study whether maintenance operations could be performed on complex mechanical assemblies. Traditionally, such studies are done manually with the help of physical rapid prototypes. Expensive prototypes can be replaced with the use of motion planning techniques and the whole design process can be speeded up, since the lead-time for the rapid prototypes is several days. Chang and Li reported that the users found the inconsistent performance of the used randomized motion planning algorithm disturbing. Such a virtual prototyping system can be augmented with haptic and visual interfaces to allow a human operator to guide the planner for more efficient operation (Amato et al. 1998b).

Siméon et al. (2001) describe the use of motion planning techniques for planning logistics and operations in large industrial installations, such as power plants. The system interfaces with commercial CAD systems to import geometric data and provides several randomized motion planning algorithms for generation of motions for holonomic and non-holonomic devices. The system is also commercially available from a spin-off company (Kineo 2001, 2003).

Generating motions for computer-animated figures can be very burdensome. Again, automatic generation of motions for the digital actors requires some form of motion planning capability. Koga et al. (1994) presented an animation system with automatic generation of manipulation motions for 7 *dof* approximation of the human hand. This system also used RPP.

Recently, new promising applications for motion planning have been found in chemistry and pharmacology. Singh et al. (1999) use motion planning techniques in screening promising molecules for drug development. Once ligand molecules with suitable 3D structure have been screened from a database of known molecules, ligand-protein binding motions are generated to identify possible receptor affinity. Song and Amato (2000) use motion planning to study how one-dimensional amino acid chain folds into three-dimensional protein structure. Once the genome of an organism is decoded, the next steps of research involve study of the structure of the coded proteins (folding) and their role in cell processes (binding). Due to importance of drug development, it may well be that the future driving applications of motion planning research will be from this new promising field.

It should be noted that also some commercial robot off-line programming systems and other software advertise motion planning capability (e.g. Delmia 2003a, 2003b). However, the algorithms are usually proprietary, and it would be difficult to assess them without access to the software.

4. PREVIOUS WORK AND RECENT DEVELOPMENTS

This chapter presents an overview of the research in motion planning algorithms. There is a large and growing body of knowledge about motion planning available in predominantly robotics related scientific literature accumulated over the past decades. Several excellent books (Schwartz et al. 1987, Latombe 1991, Gupta and del Pobil 1998) and survey articles (Schwartz and Sharir 1990, Hwang and Ahuja 1992, Gupta 1998, Latombe 1999) have been published that integrate and condense the research results from different points of view. It would be futile to attempt to duplicate such an integrative presentation here. Instead, this section presents a more modest, two-phase overview of the previous research. First, a broad overview of the algorithmic approaches to motion planning is presented to set in place the larger framework of the contributions made in this thesis. Second, a deeper survey is done in the literature published on the probabilistic roadmap methods, an approach of motion planning that has been very popular during the recent years and that is the approach taken also for the planner described in this thesis. The presentation in this chapter is complemented by the sections on previous related work in the papers I-VI. Those sections present more focused overviews of the previous work relevant to the particular issues addressed in each of the papers. A newcomer to motion planning research would benefit from reading the introductory chapter of Latombe's text book (1991), especially, because of the excellent illustrations presented there.

Hwang and Ahuja (1992, p. 226) classify motion planning algorithms into several categories. The distinctions are made according to the completeness and scope of the algorithms. Hwang and Ahuja separate three types of completeness. **Exact** algorithms guarantee a solution if one is possible or report the problem as unsolvable. **Resolution complete** algorithms discretize some continuous quantities such as object dimensions or configuration parameters, but become exact in the limit as the discretization approaches a continuum. For **probabilistically complete** algorithms the probability of finding a solution can be made to approach one if the problem is indeed solvable. Most such algorithms use a randomized search procedure, which is guaranteed to find a solution if it is allowed to run long enough. **Heuristic** algorithms are often non-complete as they may fail to find a solution even when one exists. The attractiveness of heuristic algorithms comes from the property that they usually succeed or fail fast. The scope of an algorithm is **global** when it uses all the information in the environment and it plans the complete motion from start to goal configuration. **Local** algorithms use information only from the nearby obstacles and they are used to plan only short motions. Local algorithms are used as components in global planners or as safeguards when all the obstacles are not known precisely. Latombe (1991, p. 21) points out that although the distinction between local and global methods is intuitive and practical, it has no solid theoretical basis.

There is a rather good consensus about the main algorithmic approaches to motion planning (Latombe 1991; Hwang and Ahuja 1992). The approaches are cell decomposition, roadmap, potential field and optimization. The first two approaches have roots in the development of exact algorithms based on principles from computational geometry. The potential field method was introduced as an on-line control procedure for local collision avoidance, but it has also been developed into global motion planners. Although optimization procedures have been rarely used to attack the motion planning problem directly, optimization techniques are often used in combination with the others, e.g. gradient descent with potential fields. The same observation is valid for most motion planners: They usually combine techniques for representing the *cspace* and searching the representation for possible a solution. Roadmaps and cell decompositions are used to represent the *cspace* and a search or other optimization-like procedure is used to find a solution from the representation.

Cell decomposition methods represent the free *cspace* as a collection of simple cells and adjacency relationships between those cells. (Latombe, 1991, p. 200; Hwang and Ahuja 1992, p. 234-235) Once the cell decomposition is computed, the motion planning task can be solved by searching a sequence of adjacent cells from the cell containing the start configuration to the cell containing the goal configuration. Such a sequence is called the channel. Cell decomposition methods can be further divided into methods for exact and approximate decomposition. Exact decomposition methods produce cells that correspond precisely to the free *cspace* boundaries and the union of the cells is exactly the free *cspace*. The exact cell decomposition algorithm by Schwartz and Sharir (1983) can solve the basic motion planning problem and several extensions including problems for articulated robots. The method is mainly interesting as a proof of existence of a general motion planning algorithm. It has double exponential complexity in the number of degrees-of-freedom, and thus not suitable for practical use.

Approximate cell decomposition methods do not compute the cell precisely, but use some predefined simple shape such as rectangloid to conservatively approximate the free *cspace*. (Latombe, 1991, p. 249; Hwang and Ahuja 1992, p. 234-235) The rationale of approximate methods is that they are much easier to implement than exact methods. The trade-in is that these methods are typically only resolution complete. The resolution can be in principle made arbitrary high, up to the resolution of the robot's actuators if needed. However, computational complexity restricts these methods to small-dimensional problems (*dof*<5) (Latombe, 1991, p. 249). Lozano-Pérez and Brooks introduced the method for 2-dimensional (Lozano-Pérez 1981) and 3-dimensional *cspaces* (Brooks and Lozano-Pérez 1983). For 3-dimensional problem they use octree with leaf nodes labeled to be completely in the free *cspace*, completely in the non-free *cspace* or partially in the free *cspace*. If no channel can be found from the octree, mixed nodes are further divided and the new labeled leaves added to the octree and the search is repeated. This continues until the search succeeds or some predefined resolution limit is met. Lozano-Pérez (1987) presents an algorithm that represents the free *cspace* as a tree of feasible ranges for the joints of a manipulator. The joint values are quantized at some predefined resolution. This algorithm is said to be the first resolution-complete planner for general manipulators that has been implemented (Hwang and Ahuja 1992, p. 260, 263). However, the algorithm has a weak point that it is exhaustive (Hwang and Ahuja 1992, p. 264).

Donald presents a motion planning algorithm for 6 degrees-of-freedom robots in 3-dimensional workspaces. (1987) The algorithm searches a six-dimensional lattice of points cast over the *cspace*. Each lattice point represents a neighborhood in the *cspace*. The algorithm computes the free *cspace* boundaries and uses heuristic "*local experts*" tuned to different types of free *cspace* points to improve the search efficiency. Donald states that this algorithm is the first practical algorithm in its class. This algorithm is interesting as a transitional algorithm from algorithms deriving from computational geometry and emphasizing the *cspace* representation to algorithms relying on efficient collision detection at discrete configurations and emphasizing efficient search procedures (Gupta 1998a). An example of the other side of this transition can be seen in the randomized motion planner presented by Glavina (1990; 1991).

Kondo develops a very efficient grid search method for motion planning. The first version of the method is essentially a bi-directional best-first search algorithm that expands a rectangular grid representation of the *cspace* (Kondo and Kimura 1989). His next motion planner uses A* search with weighted Euclidean distance as the heuristics. (Kondo 1991a) Kondo experiments with different weights on the degrees-of-freedom of the robot, but concludes that no recommendation can be made other than that the heuristics should be greedy. As a remedy to the difficulty of selecting the best heuristics *a priori*, he presents a very efficient multi-heuristic grid search algorithm. (Kondo 1991b) The algorithm uses several randomized heuristics in a round-robin fashion to guide A* search on the grid representation of the *cspace*. The efficiency of each

heuristics is evaluated and more efficient heuristics guide the search more. However, the algorithm can have excessive memory consumption, and it is only practical for problems that do not require large backtracking motions. This algorithm was the starting point for the research presented in this thesis.

The **potential field** approach can be best described as using a scalar function spanned over the *cspace* as the guiding heuristic for grid search (Latombe 1991, p. 19). The function is called the potential, and it is a combination of repulsive potential from the obstacles and attractive potential from the goal configuration. The robot is then driven by the potential until it stops at some stable point. This approach is best known from the work by Khatib (1985), although Loeff and Soni (1975) presented similar idea. There is a problem with this approach, though. Koditschek (1987) proves that all general potential fields have saddle points in addition to the local minimum at the goal configuration. This means that simple gradient procedures can terminate at points other than the intended goal configuration. This has been long known as the local minimum problem. In order to deal with the problem, potential field motion planners can try to construct a potential field with no or few local minima, devise some techniques for escaping the local minima, or both. Connolly et al. (1990) propose a potential function derived from the solution to Laplace's equation. Such a solution is a harmonic function and does not have local minima other than the one at the goal configuration. The problem with this approach is that since the potential is computed numerically on a grid representation of the complete *cspace*, it is only suitable for low dimensional problems. Souccar et al. (1998) describe application of harmonic functions for various motion planning problems and reactive motion control, but only for systems with up to 4 degrees-of-freedom. Bohlin (2002) introduces a harmonic potential function based motion planner for free-flying rigid bodies (6 degrees-of-freedom). The planner demonstrates promising performance, but it is not complete.

Barraquand and Latombe introduced one of the first randomized motion planners. (Barraquand and Latombe 1990, 1991; Latombe 1991, p. 340-350) Very appropriately, their planner is called Randomized Path Planner (RPP). It combines gradient descent on the potential with a random walk procedure to escape spurious local minima. RPP leaves the start configuration with gradient descent, and if it terminates at a spurious local minima rather than the intended goal configuration, a random walk of some length is started from the local minimum. Once a lower potential value is found or the length is attained, a new gradient descent towards the goal is attempted. If no lower potential can be found after a given number of descent and random walk iterations, a backtracking move to some previous configuration on a random walk segment of the current solution candidate is executed. The process is iterated from that configuration. RPP does not require any particular type of potential, or any potential at all, but can be guided by the distance to goal if the distance metric is defined to be infinite at configurations belonging to the non-free *cspace*.

RPP is one of the most important motion planning algorithms. While Schwartz and Sharir (1983) and Canny (1988) showed the existence of general motion planning algorithms, those algorithms were mainly of theoretical interest. They have not been implemented for use in any practical application (Schwartz and Sharir 1990). Although practical algorithms existed before (e.g. Lozano-Pérez 1987, Faverjon and Tournasoud 1987), RPP was instrumental in demonstrating that difficult high-dimensional problems from relevant applications can be solved in practical time (Graux et al. 1992, Koga et al. 1994, Chang and Li 1995). RPP has a well-known deficiency, though. If RPP has been trapped in a local minimum that can be only escaped against the potential through a narrow passage, RPP has to find the passage with a random walk, and that can take a very long time. This phenomenon was demonstrated by Zhu and Gupta (1993) in a very interesting experimental study.

Caselli et al. (2002) add line search heuristics for escaping "*simple*" local minima to a potential field planner, which is very similar to RPP. Their experiments show improvement in average run-time performance and variance over pure random walk

escaping. They keep random walks in order to maintain probabilistic completeness of the planner. They also show superior performance of their potential field planner over the probabilistic roadmap planner proposed by Kavraki and Latombe (1994). It is not clear from the paper what kind of potential they use and whether the construction time of the potential is included in the planning times. Nevertheless, the results demonstrate that the search techniques in RPP are still relevant over a decade after it was introduced.

Hwang and Ahuja note that the best way to use potential field approach is to use it as a local planner with some global method (1992, p. 234, 236). Faverjon and Tournasoud present such a planner for manipulators with many degrees-of-freedom. (1987) The local planner combines the usual attractive goal potential with constraints on the velocity at which the manipulator can approach the obstacles. The global planner decomposes the *cspace* into cells and updates the probabilities that the local planner will succeed inside a given cell after each invocation of the local planner inside that cell. A* search is used to find the most probable sequences of cells from the start containing cell to the goal containing cell, and the local planner is called to generated path segments between cells. This planner introduced the two-level approach of combining local and global planners.

Gupta and Zhu use numerical potential fields for solving subproblems in their motion planner for manipulators with many degrees-of-freedom. (Gupta and Zhu 1994; Gupta 1998b) The algorithm considers the links of the manipulator sequentially as two-dimensional subproblems. A backtracking mechanism is used to recover if no solution can be found for the selected subproblem. Artificial *"virtual"* obstacles are placed to force the planner to unexamined portions of the *cspace*. Due to arbitrary nature of the virtual obstacles, the planner is not complete. However, the planner has the advantage of being deterministic.

Motion planning can be taken as an **optimization** problem. The most obvious approach is to formulate the problem as a variational problem and use some optimization procedure to optimize a functional that includes goal attracting and obstacle repulsing potential over the entire path (Latombe 1991, p. 317). However, such an optimization problem becomes highly non-linear due to the constraints defining the *cspace* obstacle boundaries. Barraquand and Ferbach (1994) use dynamic programming rather than gradient search to improve the candidate path. They restrict the dynamic programming to up to 4-dimensional subspaces of the *cspace* in order to keep it tractable. They conclude that the method can be fast, but RPP outperforms it on more difficult problems. The variational planner is, however, more suitable for constrained motion planning problems, such as manipulation planning.

If the motion planning problem is formulated as search of the global minima of the potential on a discrete representation of the *cspace*, any optimization procedure that yields a search trajectory from the starting point at the start configuration to the optimal point at the goal configuration can be used to produce a solution path. Many neighborhood based optimization procedures are relevant, see e.g. (Blum and Roli 2001).

The **roadmap** method is a global approach that represents the connectivity of the free *cspace* as a network of one-dimensional curves. (Latombe 1991, p. 153; Hwang and Ahuja 1992, p. 232) Once the roadmap is constructed, a motion planning query can be answered by connecting the start and goal configurations to the roadmap, and searching the roadmap for a solution. The crux of the approach is of course the construction of the roadmap. Several methods have been introduced. The visibility graph method (Nilsson 1969; Lozano-Pérez and Wesley 1979) builds the roadmap by connecting every pair of obstacle vertices with a straight line edge. Edges that cross into obstacles are pruned and the remaining graph captures the connectivity of the free *cspace*. If the robot is a dimensional object, Minkowski set difference or other method (see e.g. Latombe 1991, p. 105-149; Hwang and Ahuja 1992, p. 228-230) can be used to compute the *cspace* obstacles and the visibility graph is then built from them.

Visibility graph methods as such cannot handle rotational degrees-of-freedom, and it has rarely been used for systems with more than 2 degrees-of-freedom (Latombe 1991, p. 169; Hwang and Ahuja 1992, p. 233).

Besides the difficulty of handling rotations, the visibility graph approach has the deficiency that it yields solutions that pass points at the obstacle vertices. It would be preferable to keep some distance to the obstacles. A generalized Voronoi diagram is a set of points that maximizes the distance between the robot and the obstacles. There exist numerous methods for constructing Voronoi graph for a particular type of robot and environment, but mostly for two-dimensional spaces, see (Latombe 1991, 169-176). Canny and Donald (1988) present a form of Voronoi diagram that is easy to extend to higher dimensional cases. As described below, the Voronoi diagram has reappeared as guide for sampling the free *cspace*. Freeway method (Brooks 1983) is quite similar to Voronoi diagram as it tries to capture the connectivity with a graph that keeps distance to the obstacles. Freeway method builds the graph from the spines of generalized cylinders extracted from the work space. The method is conservative and thus may fail to find a solution even if it exists.

Canny's roadmap algorithm (1988) is a general method that established a single exponential lower bound in the number of degrees-of-freedom for the generalized mover's problem. The algorithm sweeps a hyperplane across the free *cspace* and traces the extremal points of the intersection between the plane and free *cspace* boundary in some arbitrary direction to get one-dimensional silhouette curves. Separate silhouette curves are connected by running similar sweep across the subspace of intersection between the plane and free *cspace* at locations of the plane where the connectivity of the curves changes. The sweeping is continued recursively until there is no change of connectivity at some subspace or the dimensionality of the subspace is two. The combined set of silhouette curves is the roadmap.

Hwang and Ahuja define a class of methods that they call the subgoal network (1992, p. 233). This approach intends to build an approximation of the free *cspace* in the form of a graph of reachable configurations. A motion planning task is solved when the start and goal configurations are connected to the same connected component of the graph. Motion planners in this class need components that generate prospective intermediate configurations, sequences of intermediate configurations for solution candidates and local motion planning algorithm that can be used to test the reachability of the intermediate configurations in a solution candidate. This approach was introduced in the Faverjon's and Tournasoud's planner (1987), and it has become to be the dominant approach to practical motion planning.

A subgoal network motion planner was presented by Glavina (1990). Glavina's motion planner (1990, 1991) was a novel one in many respects. Together with RPP, it was one of the first randomized motion planners. The planner generates random subgoal configurations in the *cspace* and a local planner is used to connect the subgoals in order to form a graph that approximates the connectivity of the *cspace*. The local planner is derived from the local experts of Donald (1987). It proceeds greedily in the *cspace* towards the goal configuration sliding along the *cspace* obstacle surfaces until it reaches the goal or gets trapped in a local minimum. If the local planner fails to connect the start and goal configurations directly, subgoals are generated and the local planner tries to connect them to start, goal and other subgoals until a graph containing a solution is found. Glavina points out that his planner conserves memory, since it stores only one-dimensional subspaces of the free *cspace*.

Glavina's algorithm introduced techniques that have become widely used in numerous planners since then. The basic techniques of point subgoals, local planner and graph approximation of the *cspace* have been established in the current state-of-the-art motion planners. A version of the planner (Baginski 1996) had a restart loop, a known technique for variance reduction that has been recently proposed for use in the current randomized planners (Geraerts and Overmars 2002). The algorithm can also be seen as a representative of the shift of focus from computational geometry to search

in discrete *cspace*, since it is in several respects a discrete approximation of Donald's algorithm. There is also a link to mathematical optimization procedures, since the sliding local planner is very similar to the gradient projection method for convex optimization. Because of the simple local planner, Glavina's algorithm must rely heavily on the randomization to produce easy subproblems. A similar subgoal utilizing algorithm, but with a simple straight-line interpolation as the local planner was presented as an example of a randomized search procedure by Hwang and Ahuja in their survey paper (Hwang and Ahuja 1992, p. 238-239).

Overmars introduced another similar algorithm and studied its properties experimentally. (1992) His work is the first to identify and address the issues that have been established as the standard problems for research in this class of planners: sampling strategy, distance metric, connection strategy, and local planner. A **sampling strategy** (*"adding strategy"*) determines where to place the subgoal configurations. **Distance metric** is useful for selecting prospective pairs of subgoals for the **local planner** (*"simple motion"*) to connect. **Connection strategy** (*"select neighbors"*) determines how to deal with subgoals from different connected components of the subgoal graph when connecting the newly generated subgoal to the graph. Overmars also recognizes the difficulty of finding paths through narrow passages with random sampling. He presents ideas such as pushing samples in the non-free *cspace* to the boundary of the free *cspace*, adding new subgoals in the vicinity of the existing configurations, and using the number of different components near the new sample to determine the probability it will be added to the graph. He also presented ideas for improving the quality of the solution paths, using the approach for learning, and extending the approach for non-holonomic devices. Overmars notes that the planner is easy to parallelize by computing *"simple motions"* concurrently and that variance can be reduced with restarting.

Chen presented a learning algorithm for motion planning. (1992) The algorithm stores the learned knowledge as an *"experience graph"*, which is a subgoal network. The experience graph is constructed while solving planning tasks by augmenting it with paths generated by a powerful planner, but *"abstracted"* into a sequence of subgoals that can be connected by a fast planner to solve the same task.

Chen and Hwang presented motion planners that are based on a search strategy they call SANDROS. (Chen and Hwang 1992; Hwang and Chen 1995; Chen and Hwang 1998) The SANDROS strategy is a two-level, non-uniform search technique that uses a simple local planner to connect heuristically generated subgoal sequences. They claim that a significant improvement over their algorithm is only possible with radically different approaches, like parallel processing or knowledge-based algorithms. However, they also note that the local planner they use may fail to find paths through a winding tunnel. A version of the SANDROS planner was available as an option to the TELEGRIP (Deneb 1994) robot simulation software. That version failed to solve the test problem in figure 4 of paper II.

Bessiére et al. presented a motion planner that is based on genetic algorithms. (1993) They call the method Adriane's claw algorithm. One genetic algorithm called EXPLORE is used to place point subgoals or *"landmarks"* evenly in the *C-space*, and the other algorithm called SEARCH is used to connect these landmarks to the goal configuration. EXPLORE distributes the landmarks at the end of a Manhattan path from the start configuration so the path segment to the landmark is immediately known. An interesting feature of the algorithm is that it optimizes the distribution of the landmarks or samples in the free *cspace* rather than uses some random process to place them. More recently, similar effect has been obtained with quasi-random numbers that have a proven property of "good" distribution over the *cspace*, see below for more.

Research in this family of motion planners got a major boost in 1994 when several groups presented independently planners based on these ideas at the IEEE International Conference on Robotics and Automation and elsewhere (Kavraki and

Latombe 1994; Horsch et al. 1994; Overmars and Švestka 1994). These planners had a novel focus of executing a preliminary preprocessing or learning stage to construct the roadmap and amortizing this preprocessing cost over multiple motion planning tasks in the same *cspace* during a subsequent motion planning query stage. Kavraki and Latombe contributed node enhancement technique to add nodes in difficult regions of the *cspace*. They used RPP in a component reduction stage to connect components of the roadmap that the simple local planner failed to connect. Horsch et al. presented a reflecting local planner. Overmars and Švestka introduced structured sampling and extension to non-holonomic robots. Many of these results were integrated into comprehensive presentation by Kavraki et al. (1996).

While the subgoal based algorithms introduced before 1994 produce a roadmap with probabilistic techniques, and thus, are definitely probabilistic roadmap algorithms, they are often omitted in presentations of the developments, e.g. (Overmars 2002, Ladd and Kavraki 2002). In this sense, they can be called pre-historic probabilistic roadmap algorithms. There are some distinctive properties in the probabilistic roadmap algorithms as the narrower class that sets them apart from the larger class of subgoal network based algorithms. Hwang and Ahuja (1992, p. 234) describe the subgoal network approach as a task decomposition technique that is intended to decompose the original problem into a set of easier subproblems. Probabilistic roadmap was introduced as a technique to capture the connectivity of the whole *cspace* of the robot in the workspace independently of some particular motion planning query. The single query subgoal network approach has an obvious stopping criterion in the finding of the solution to the query. Probabilistic roadmap construction has to be stopped at some arbitrary limit on run-cost, roadmap size, coverage, or some such metric. A probabilistic roadmap planner usually builds a graph representation of the *cspace* while subgoal network planners more often build trees of reachable subgoals rooted at the start and goal configurations. In the recent literature, the term probabilistic roadmap (PRM) has been taken to signify all planners that rely on sampling of the *cspace* and building a representation of the *cspace* from those samples. Most relevant algorithmic techniques are equally applicable to both types of motion planners, and in the following these techniques are surveyed without paying particular attention to the type of the planner they are embedded in.

The most obvious sampling strategy is to sample the *cspace* uniformly, and this was the strategy used in the earliest PRM planners. The uniform distribution was usually implemented with pseudo-random number generators. Branicky et al. (2001) proposed the use of quasi-random numbers, since they can be proven to have a good coverage of the *cspace* unlike an arbitrary segment of a pseudo-random numbers. Quasi-random numbers can also be said to be "deterministic", but this is a rather technical advantage, since pseudo-random numbers are certainly deterministic when generated with a digital computer. Lindemann and LaValle (2003a) present several desirable formal qualities for a uniform sampling sequence and conjecture that they are useful for motion planning. They define a sequence that fulfils the desired properties and present an algorithm for generating it. In experiments they demonstrate that the sequence has the best performance for one out of four test problems. Interestingly, for 5 and 6 degrees-of-freedom problems, random order grid sampling demonstrates the best average performance.

The problem with uniform randomization is that some areas of the *cspace* are often more "critical" than other. As an instance of this general problem, the susceptivity of RPP to the narrow passage problem was demonstrated by Zhu and Gupta (1993). It was known from the very beginning that the same problem plagues also motion planners that use randomized uniform sampling of the *cspace* (Overmars 1992). If the solution requires passing some small but critical portion of the free *cspace*, the solution cost can become dominated by the low probability of obtaining a sample in that portion with uniform sampling. This observation has motivated the development of various structured sampling techniques to increase the probability of obtaining samples in

"difficult" areas. The general idea is to use the workspace or *cspace* properties to bias the sampling towards some areas of *cspace*, typically to increase the probability of obtaining samples in the narrow corridors. Overmars and Švestka (1994) used geometric features of the workspace to place the samples.

One possibility to increase samples in narrow passages is to "push" samples inside *cspace* obstacles out to the free *cspace* with the intention that some of them end up in the passage (Overmars 1992; Hsu et al. 1999). Amato et al. (1996) use the same idea in their Obstacle-Based PRM (OBPRM), but push the samples from the free *cspace* towards the obstacles. Boor et al. (1999) generate the samples in close pairs and keep only those samples that lie in the free *cspace* and have their pair inside a *cspace* obstacle. Sampling near the obstacles has demonstrated good performance in several experiments, but like all heuristics, it has limitations. Laumond and Siméon (2001) note that sampling near obstacles may actually require more samples to obtain good coverage of the free *cspace*.

Sampling at the generalized Voronoi diagram or medial axis of the free *cspace* avoids the explicit construction of the Voronoi diagram, but takes advantage of its properties. A probabilistic roadmap planner sampling at the medial axis has been shown to increase sampling in the narrow passages for a point robot in 2D environment (Wilmarth et al. 1999a). The development of medial axis sampling planners for more general problems has been hindered by the lack of efficient algorithms for computing the penetration depth of non-convex objects (Wilmarth et al. 1999b). However, approximation methods for non-convex free-flying objects and articulated robots have been recently proposed and improvement over naive uniform sampling has been demonstrated (Lien et al. 2003). Holleman and Kavraki sample at the workspace medial axis (2000). It is unclear, however, how to generalize workspace medial axis to articulated robots.

The structured sampling strategies appear to provide a performance improvement when demonstrated with low-dimensional problems. The major problem with structured sampling strategies is that many of them sample a space that has a dimensionality of up to *dof*-1. With high-dimensional problems they can reduce to sampling a high-dimensional subspace of the *cspace* with out finding the narrow passage in the workspace. This is a direct consequence of the fact that the methods rely on locating a *cspace* surface, but the surface can be highly dimensional, and thus, can have a large cardinality with only a subset of the points interesting in terms of the narrow passage property in the workspace. Simply put, the subspace that structured sampling strategies sample may also contain a narrow passage that the solution has to pass. Several such problems are included in the test set of this thesis.

Devices with closed kinematic chains require a specific sampling method that does not violate the closure constraints of the device. Generally, the set of configurations satisfying the closure constraints is lower dimensional that the *cspace* of the device. (LaValle et al. 1999) Therefore, the probability of random configuration satisfying the closure constraints is zero. LaValle et al. (1999) address the problem by maintaining the closure constraints only to within a specified tolerance. Their sampling strategy is to generate random configurations and use randomized gradient descent to minimized the violation of the closure constraints to within the tolerance. Furthermore, they use similar randomized gradient descent to generate path segments between the samples. It is unclear whether the method is efficient when the tolerance is made small enough for the paths to be executable by physical systems.

Han and Amato (2001) present two-stage strategy for generating samples for closed chain devices. At the first stage, they construct a structure they call a kinematic roadmap that contains a set of self-collision-free closure configurations. The configurations are generated by breaking the closed loop into an active part and a passive part. The joint values of the active part are generated randomly, and the joint values for the passive part are computed with the inverse kinematics. If no joint values for the passive part satisfying the closure constraints can be found, the configuration for

the active part is rejected and a new one is generated. Valid configurations in the kinematic roadmap are connected by making the selected local planner to drive the active joints and using the inverse kinematics to compute such joint values for the passive part that the closure constraints are maintained. At the second stage, they populate the environment with copies of the kinematic roadmap and use a rigid body motion planner to connect the configurations of the same closure type from the copies of the kinematic roadmaps. The final roadmap consists of a connected set of collision-free portions of the kinematic roadmaps distributed into the environment. Cortés et al. (2002) present an algorithm that improves the efficiency of generation of valid random configurations for a closed chain device. The algorithm generates the joint values for the active chain sequentially so that the end-frame of the active part is guided towards the workspace reachable by the passive part.

Once the samples have been generated, there are a quadratic number of possible path segments between the samples to be tested for connectivity with local planner. Several graph building strategies have been proposed to prioritize or restrict the segments that are attempted. The growth of sample adding cost with the size of the graph can be limited by setting some constant limit on the number of nodes that the new sample is tried to connect (Kavraki and Latombe 1994). It is not necessary to connect the new sample to more than one sample in each connected component of the roadmap (Overmars 1992).

Siméon et al. (2000) use visibility criterion to decide which nodes to add to the graph. A newly generated node is added to the roadmap only if it can be connected to several connected components of the roadmap or it can not be connected to any. This strategy keeps only nodes that contribute to the coverage of the roadmap. Visibility strategy demonstrates performance improvement over trying to connect the new node to every node of the roadmap. Laumond and Siméon (2001) discuss the combinatorial topology induced by different local planners ("*steering methods*"). The selection is known to be very important for obtaining good performance from a PRM-type motion planner, but there is little theory to guide the selection.

Lazy PRM generates the samples, but postpones the path segment generation by local planning to the query stage (Bohlin and Kavraki 2000). Thus, the planner is very similar to early planners by Glavina (1990; 1991) and Overmars (1992). Their local planner is very simple, but it tries to speed up the identification of a failure. The most important benefit appears to be the opportunity to attempt path segments between promising samples, since all the samples are available from the beginning. It was demonstrated by Amato et al. that near samples are more likely to be connected by the local planner (2000). Nielsen and Kavraki (2000) present a fuzzy PRM planner that keeps account of the probability that a particular path segment will be collision-free. The probability is used during query stage to find and verify segments that are more likely to be collision-free.

A number of planners build the probabilistic roadmap as trees rooted at start and goal configurations. Hsu et al. (1997) present a planner that expands the trees away from the existing samples in the roadmap. Rapidly-Exploring Random Tree (RRT) is an algorithm originally intended for kinodynamic planning (LaValle and Kuffner 1999; 2001). RRT uses sampling with Voronoi bias to extend the trees toward unexplored *cspace*. Versions of RRT have been used in numerous planners, either alone or as a local planner in the PRM framework. RRT implementations have demonstrated impressive performance as a standalone motion planner (Kuffner 2002) and as the local planner in a PRM motion planner (Siméon 2001). The recent effort to design derandomized RRT variants (Lindemann and LaValle 2003b) is very interesting development, since it holds the promise of an efficient deterministic motion planner.

The complementary nature of the local planner and the global randomized sampling procedure has been noted by several researchers (e.g. Overmars 1992; Chen and Hwang 1992; Kavraki and Latombe 1994). On the other hand one can use very powerful local planner to connect start, goal, and few subgoals to form the solution. At

the extreme the local planner would be a complete motion planning algorithm and no subgoals are necessary. On the other end of the spectrum the local planner is very weak and many subgoals are necessary. At this extreme the local planner could be a small change in one of the degrees-of-freedom and the global planner needs to be a full search algorithm such as A* search (Hart et al. 1968). Over the years, many local planning algorithms have been introduced and used. They have tended to be relatively simple and fast and motivated by the use of the PRM approach as learning method. The path segments between samples are not to be stored with the roadmap but to be reconstructed with the local planner during the query stage (Kavraki and Latombe 1994; Horsch et al. 1994).

First local planners were greedy "sliding" type of local search procedures (Glavina 1990; Overmars 1992; Chen and Hwang 1992). Horsch et al. (1994) used "bouncing" local planner that reflects to a new random direction if it makes a contact with a *cspace* obstacle on its way towards the target sample. Qin's and Heinrich's randomized parallel planner uses rule-based local planner (1996). Amato et al. test several interpolating and *"A* -like"* local planners (2000). Vallejo et al. present a single query PRM planner that uses multiple local planners including some simple ones and some that have been derived from Adriane's claw and RRT (2000).

A large number of variant PRM planners can be constructed by combining the algorithmic components presented above. Amato et al. compare experimentally several distance metrics and local planners in their capability of producing roadmaps with large number of path segments with OBPRM (2000). Euclidean distance is their recommended distance metric. Relatively costly *"A* -like"* local planners produce the most path segments, but this is unsurprising since they do not restrict the running cost of the (local) planners. Vallejo et al. find out in their experiments with several subgoal generation techniques and local planners that the most problems were solved by combining the techniques and local planners (2000). Combining local planners or complete planners have been suggested many times (e.g. Hwang 1996; Chen and Hwang 1998; Amato et al. 1999), but, generally, criteria for selecting motion planners to combine and techniques for selecting which one to execute at a given time are largely unaddressed research problems. Since realistic work spaces are usually geometrically much more complex (Chang and Li 1995; Hsu et al., 1997; Bohlin 2002) than the typical "blocks worlds" used to demonstrate motion planners, developing such selection techniques will be a nontrivial endeavor.

Geraerts and Overmars (2002) have made an experimental comparison of various local planning, sampling, and node connection strategies for constructing the roadmap for free-flying robot. The experiments consider linear and binary searching for collisions on the path segment between samples. The tested sampling strategies include the usual pseudo-random sampling, grid sampling, quasi-random Halton sequence sampling, workspace cell decomposition sampling, a novel random Halton sequence sampling, Gaussian sampling, and two obstacle based sampling methods. Tested node connection strategies are the basic strategy of connecting to k nearest nodes, connecting to the nearest node in each connected component, connecting to the k nearest nodes in each connected component, and visibility based connection. Probably the most important insight from the experiments is that the proposed techniques do not always produce the expected benefits and sometimes the original claims of benefit were contradicted. This is to be expected due to the heuristic nature of most of the tested techniques. Further, combinations of techniques that individually demonstrate good performance can result in inferior performance. Among the sampling strategies, the best performance is demonstrated by Halton sequence with an additional random value added to each point from the sequence. This "noisy" sampling sequence is very interesting, since it is "re-randomized" and indicates that some amount of randomness improves the performance of the planner. The optimal amount of randomness is not addressed in the study. The overall best connection strategy is to connect to k closest nodes in each connected component of the roadmap.

Most of the above research is empirical in nature. However, a number of theoretical results form the theoretical foundation of PRM planners. In brief, the probability of failing to find a path from a probabilistic roadmap decreases exponentially as the number of samples in the roadmap is increased (Kavraki et al. 1995; Ladd and Kavraki 2002). Additional results can be found in the literature (Hsu et al. 1999a).

5. METHOD

As described in the chapter 2, the first two steps of the design process are the statements of the requirements and the specifications. Most of the specifications have been defined in the introduction, where the objective of this research was stated: The objective of this research is to construct a method for solving model-based gross motion planning problems. Many details of the requirements were implicitly defined by selecting the part of the previous work that was described in the previous section. However, some details are made explicit here. The types of problems to be solved should include the basic motion planning problem and planning for articulated robots, more specifically manipulator arms. The planner is intended only for generating collision-free motions. Issues, such as smoothness or other measures of path quality, and trajectory planning, are supposed to be handled separately. Most randomized planners produce paths that can be much longer than necessary and have first-order discontinuities. Such solutions cannot be accepted for execution by physical systems, but they are accepted as solutions from the planner. The planner should be independent of the kinematic structure of the robot and be capable of solving non-trivial problems for many degrees-of-freedom robots. Due to the complexity of the problem, an algorithm that can be parallelized easily on distributed memory parallel computers is preferred, since such computers can be build at relatively low cost from commodity hardware components. The planner is intended for solving single instances of motion planning problems, but a learning capability would be a benefit.

There are some suggestions for design ideas that emerge from the research literature reviewed above. One should use greedy search, since it will get the solution fast if the problem is easy. Greedy search will often lead to dead ends (local minima), so a backtracking mechanism is necessary. Since backtracking from deep local minima wells can be very expensive, there must be a mechanism to measure and limit the amount of backtracking performed. Combining several complementary techniques are commonly recommended. Since it is difficult to design a single heuristics that is widely applicable and efficient, multiple heuristics should be considered. Randomization has proven very effective in taking advantage of the property that usually many solutions are available for the motion planning problem at hand. Randomization will break the structure of the problem and make it difficult to design problems that exhibit the worst-case behavior. Randomization will introduce some variance to the planner, either as run-cost variability, solution quality variability, or both. The variance is disliked by the users; therefore, some technique should be used to deal with the inevitable variation in performance caused by randomization.

The beginnings of the motion planning method presented in this thesis can be read in the Hwang and Ahuja survey article. When describing Kondo's (1991b) algorithm, they *"...note that this algorithm is very fast if the solution does not require a large backtracking motion"* (Hwang and Ahuja 1992, p. 266). It is obvious then that an explicit backtracking mechanism must be added to Kondo's algorithm to get one without this deficiency. This thesis is largely the story of that backtracking mechanism.

Kondo's algorithm uses several heuristics to guide A* searching in a grid representation of the *C-space*. The search algorithm generates a portion of the grid and stores the configurations as nodes and the adjacency relationships between those configurations as links between the nodes. If the start and the goal configurations become connected in the collision-free part of the representation, a solution to the problem is found. As the heuristics are executed, they are also evaluated for efficiency and the more efficient heuristics guide the search more than the less efficient ones.

Kondo's algorithm is capable of solving many problems, but there is room for several improvements. Because a collision check is a computationally expensive operation, an improved search algorithm has been designed to use lazy evaluation principle to decrease the number of collision checks performed during the search. This

means that not all of the nodes that are generated are tested, but only those nodes that are further expanded by generating the neighboring configurations. For details, see paper I. The collision status of a node is stored in its representation. This allows saving the redundant checking of the revisited nodes in the A* search. It was experimentally found out that up to a third of the *open* operations in A* can be reopenings, so the slight increase in the memory usage pays off with a decrease in the number of performed collision checks. Similarly, according to lazy evaluation principle, the reopened nodes are just re-evaluated and inserted back to the OPEN set of A*. An alternative is to propagate the better path to the successor nodes as described by Rich and Knight (1991). However, all this effort is wasted if the successors were not selected for expansion later in the search.

The A* search is guided by one of the available heuristics at a time, but the newly generated nodes are evaluated by all of the heuristics. After a selected number of node expansions has been guided by the currently active heuristics, the active heuristics is changed and the execution of the A* search is continued with the guidance of the newly selected heuristics. The heuristics are selected for guiding the A* search sequentially in a round-robin fashion. The execution of some particular heuristics for a number of node expansions is called a stage and the execution of all available heuristics for one stage is called a round.

The efficiency of a heuristics is evaluated with the formulas introduced by Kondo. After the j th round, an evaluation value

$$P_t(j) = \frac{\sum_{(for\ \ last\ \ Q\ \ nodes)} p_t(C)}{Q}$$

is calculated for each heuristics $t = 1,...,T$, where T is the number of heuristics and a constant Q has the value of 20. The value of $p_t(C)$ is calculated for each opened node C by the following equation:

$$p_t(C) = \frac{g(C)^{dof}}{F_t(C)},$$

where $g(C)$ is the distance from the start node to the current node C in grid steps (Manhattan distance) and $F_t(C)$ is the total number of nodes opened by the tth heuristics until the expansion of node C. The exponent for $g(C)$ is intended to adjust for the dimensionality difference between the one-dimensional path length measured by $g(C)$ and the *dof*-dimensional search space volume measured by $F_t(C)$. Thus, the efficiency of the heuristics t at the current node C is estimated by the ratio between the measures of the path volume and the search space volume, and the efficiency of the heuristics at the end of a round is estimated by averaging the node estimate over the last 20 nodes.

For the first round, each heuristics is allocated $E_{init} = 25$ node expansions to be performed during its stage. For the subsequent rounds each heuristics is allocated

$$E_t(j+1) = \max\left[E_{init} \frac{P_t(j)}{\max\left[P_1(j),\ldots,P_T(j)\right]}, 1 \right]$$

node expansions for its stage. The above formula allocates node expansions according to the relative efficiencies of the heuristics. It is very similar to the one used by Kondo, but the outer maximum operation is added to guarantee that at least one expansion is always allocated.

A second evaluation value is calculated for each opened node according to the equation:

$$O_t(C) = \frac{\sum_t F_t(C)}{g(C)}$$

If the evaluation value $O_t(C)$ for the tth heuristics increases above a threshold value O_{th}, the execution of the heuristics is discontinued for that round. However, the discontinued heuristics may become active again in some later round of the search if one of the other heuristics expands nodes that move the discontinued heuristics to a better area in the search space. If all heuristics are discontinued, the local planner fails.

The search is bi-directional. In accordance to Pohl's cardinality comparison principle (Pohl 1971), after each round, the search direction with the smaller examined grid point set is selected for the next round. The effect is that the search proceeds from the cluttered space to the open space.

The heuristic evaluation of a node is performed with the function $f(C) = g(C) + h(C)$, where $g(C)$ was given above and $h(C)$ is weighted Manhattan distance between the evaluated configuration and the goal configuration. The following four weight sets are used to define four heuristics.

Manipulator heuristics:

$$a_i = \left\lceil 9\frac{dof + 1 - i}{dof} \right\rceil, \quad i = 1,\ldots, dof.$$

Position heuristics:

$$a_i = \begin{cases} 9, & i = 1,\ldots, d \\ 1, & i = d+1,\ldots, dof \end{cases}$$

Rotation heuristics:

$$a_i = \begin{cases} 1, & i = 1,\ldots, d \\ 9, & i = d+1,\ldots, dof \end{cases}$$

Even heuristics:

$$a_i = 5, \quad i = 1,\ldots, dof.$$

Above $d = \lfloor (dof + 0.5)/2 \rfloor$.

As the names of the weight sets imply, they are intended to prefer some particular type of motion. Manipulator heuristics prefers to move the joints to the goal configuration in their order in the kinematic chain starting with the base joint and progressing toward the wrist joints. Position heuristics and rotation heuristics prefer to move predominantly the one or the other half of the joints. Even heuristics is a fallback heuristics that does not have any particular preference. Although the weight sets were selected with the implied kinematic structures in mind, the essential property is that they have different preferred search directions in the *cspace*. When some direction of motion is blocked by an obstacle, some other direction may be unobstructed and let the robot to move on towards the goal.

An additional "greedy" multiplier term A=3 and a tie-breaking term ρ are added to the full expression of $h(C)$:

$$h(C) = A\left(\sum_{i=1}^{dof} a_i D_i(C,G) - \rho a_j \right)$$

$D_i(C,G)$ is the distance between the current node and the goal node G in grid steps along axis i. The tie-breaking term has a value of 0.5 if the evaluated node was expanded in the same direction as the parent node along the axis j, or 0 otherwise.

All of the heuristics share the search space representation, and therefore progress made by any of the heuristics will immediately and continuously benefit all of them. The evaluation values control the search so that the relatively more efficient heuristics are used more and the individual heuristics and eventually the local planner are discontinued if sufficient progress is not made. Combining multiple heuristics and dynamically scheduling between them avoids defining explicit criteria for selecting between distinct local planners and has synergetic effect not easily available for dissimilar distinct local planners.

The memory consumption of the local planner grows exponentially as the number of degrees-of-freedom is increased. The threshold parameter O_{th} can be used to control the memory consumption, but an additional limit on the maximum number of configuration nodes for each subtask is needed. This is a machine specific parameter that depends on the amount of main memory in the machine.

The motion planning algorithm in paper I took inspiration from RPP and added an explicit backtracking mechanism that is based on randomization. Since the search is deterministic, RPP-like backtracking along the current solution path does not lead to new portions of the *cspace*. Instead, a random configuration is generated to be a point subgoal (sample), and the search is then continued for the path segments from the original start to the sample and from the sample to the original goal. Often, the detour avoids deep local-minimum wells, but if one is encountered again, the sample generation is continued recursively until a path avoiding deep local-minimum wells is found. The mechanism provides a dramatic performance improvement, but at the cost of the completeness. It requires that every sample would eventually become a part of the solution path. Unfortunately, if any of the samples were placed in a portion of the free *C-space* that was unconnected to the start and goal configurations, the planner could never produce a solution, even if one could be found without the stray sample.

The relaxation of the requirement that all generated samples will become a part of the solution path leads immediately to the subgoal network or probabilistic roadmap approach in the second iteration of the planner in paper II. This regained the completeness and delivered an additional increase in the efficiency, since path segments to the most difficult samples are only generated if they are needed in the solution. This version uses somewhat *ad hoc* connection strategy to limit the combinatorial explosion of possible path segments. The connections between sample configurations are searched only for pairs of start connected and goal connected samples. This means that there can be up to two samples in the solution. The limit on the number of samples on the solution path was removed in the third version of the global planner in paper III. That version builds two trees of samples and successful path segments, one rooted at the start configuration and the other at the goal configuration. A graph based probabilistic roadmap type version of the planner in paper IV uses connection strategy that produces candidate sample pairs for the local planner by selecting for the new sample up to k closest nodes from each connected component of the roadmap at the sample generation time. This was observed to be the best connection strategy by Geraerts and Overmars (2002). In the experiments $k=10$. Euclidean distance is used as the distance metric as it has been shown to have good performance with minimal computational cost (Amato et al. 2000).

In addition to managing the roadmap, the global component of the motion planner controls the threshold parameter O_{th} of the local planner. At minimum the threshold has to be set at some value to make the local planner fail and let the global generate new prospective path segments. The majority of PRM planners have static local planners, which can be obtained for this study by setting a fixed upper limit for $O(C)$.

While it is difficult to select an *a priori* value for O_{th}, it is possible to set a schedule for it. Several schedules were experimented with in papers III and V. A simple strategy involves increasing the $O(C)$ limit with the size of the roadmap. The intuition behind this strategy is that more difficult problems require larger roadmaps and a more capable local planner to adequately describe the connectivity of the *cspace*. Furthermore, as more samples are added to the roadmap, the failure probability of the local planner decreases (Kavraki et al. 1995; Ladd and Kavraki 2002). With lower failure probability, failures that are more expensive can be tolerated. This strategy has a global character in the sense that it determines a single increasing $O(C)$ limit for all the samples in the roadmap.

The sample tree building global planners in papers II and III use a global exponential schedule such that the threshold parameter is updated according to the

formula $O_{th} = O'_{th} R^S$, where S is the current number of samples, O'_{th} and R are constants.

Schedules for the graph based probabilistic roadmap type version of the planner are slightly different. The following formula is used to determine the linear threshold O_{th} for $O(C)$ during roadmap construction at a particular roadmap size of S:

$$O_{th}(s) = \frac{S}{s} \times 32 \quad (1)$$

Any number of such strategies can be defined with various values for the constant s. The constant of 32 is the largest static threshold used in the experiments described below.

A local strategy is defined by setting the O_{th} threshold separately for each sample in the roadmap. The strategy uses a measure of difficultness of the *cspace* around a particular sample. For each sample v the fraction of successful calls of local planner is computed:

$$r_s(v) = \frac{N_s(v)+1}{N(v)+1},$$

where $N(v)$ is the total number of local planner calls with the configuration space sample v either as start or target and $N_s(v)$ is the number of calls that succeeded in producing a path segment to or from the sample v. This measure is very similar to failure ratio (Kavraki et al. 1996).

A value of O_{th} is computed for both start and target samples with the parameterized formula:

$$O_{th}(v,n) = 1 + \frac{n}{r_s(v)}, \quad (2)$$

and the maximum is used as the current threshold value. Again, parameter n determines the exact strategy.

Parameterized heuristics present a problem of selecting the values for the heuristic parameters. If the properties of the expected motion planning problems are known, then the parameters should of course be tuned for those problems using preliminary experiments. When tuning is not possible or desirable, then some on-line procedure can be used to select the value.

In this thesis, a metaplanner is used to select the values for parameters s and n. Since it is difficult to determine an optimal value for the parameters, the selection is done randomly from a set of reasonable values for each parameter. The metaplanner selects a parameter value uniformly from the set at the start of the execution of the PRM planner. A motion planner called PRM-C uses a static local planner with a constant value of parameter O_{th} selected by the metaplanner at the start of the run. PRM-G uses the global adaptation of the local planner according to the equation (1) with parameter value s selected by the metaplanner. Similarly, PRM-L uses the local adaptation strategy defined by the equation (2) and parameter n. Based on the preliminary experiments (see paper V), the reasonable ranges of parameter values were determined and the sets defined to be $\{2, 4, 8, 16, 32\}$ for o, $\{300, 1000, 3000, 9000\}$ for s and $\{0.01, 0.03, 0.1, 0.3\}$ for n.

Using a powerful search based local planner in the preprocessing stage of a learning PRM planner is hindered by the fact that the path segments must be reconstructed during the query stage. Keeping the queries very fast motivated the use of simple but very fast local planners. However, there is no obligation for the use of the local planner during query stage if the connections between nodes can be reconstructed by other means. Paper V introduced the use of path optimization to transform the path segments into a form that can be rapidly reconstructed during the query stage with a simple local operator.

Once the local planner generates a path segment, it is transformed with a polygonal optimizer (Berchtold and Glavina 1994). The optimizer deletes a configuration from the solution path if a collision-free straight-line connection can be made from the

preceding configuration to the successor configuration. The remaining configuration triplets are considered as triangles and the corners of the triangles are "cut off" as much as possible to further reduce the length of the path segment. Finally, retracting the remaining configurations toward the straight line connecting the preceding and successor configurations smoothes the path.

The optimizer returns the solution as a sequence of configurations that can be connected by straight-line interpolation. These configurations are stored with the roadmap edge. If needed at query time, the path segment is reconstructed by interpolation in the local operator without performing any collision checks. Thus, rapid query processing is maintained.

6. EVALUATION FRAMEWORK

This chapter describes the evaluation framework used to assess the performance of the motion planner variants described in the previous chapter. This is part of the fourth activity in design process and required here since an improvement over previous results should be shown. Most of the performance assessment in motion planning research is experimental. Sometimes theoretical results are presented, but they are usually related to convergence or other completeness properties. Due to the complexity of the problem, many motion planners use heuristic techniques, which are often difficult to formulate precisely except in some idealized model. Even if formal approach is possible, it is difficult to select appropriate input distributions for the analysis, since it is not known how to condense complex 3-dimensional geometric scenes into a form amenable to theoretical analysis of e.g. average run-cost. Because of these reasons, it is very difficult to prove performance results beyond restating the known worst-case complexity.

The standard in motion planning research is that when a novel algorithm or technique is being introduced, its performance is demonstrated with a set of a few test problems selected by the authors to be suitable for their case. When describing the experimental results and assessing the performance improvement provided by the novel properties of the algorithm, designed experiments are very rarely used and specific hypotheses are seldom formulated and tested statistically. Especially now, when the most successful motion planners are based on randomized algorithms, it would very important to verify that the observed benefits from novel techniques are not just some chance variation and not significant in the statistical sense. The very same data that is needed for reliably estimating the usual average run-cost can provide estimate for the variability of the run-cost, and thus, an estimate of the statistical significance of the differences.

In this thesis, the assessment of the algorithm and its components are done with well-established empirical methods and lessons learnt in other related empirical fields of research are used. Crowder et al. (1978) identify three types of experimental studies and set different qualifications for each of them: feasibility studies, assessments, and performance comparisons. Clearly, this thesis does not present an entirely novel approach to motion planning, so some improvement in performance over the existing motion planning methods should be shown. But as explained by Hooker (1994, 1995), it is more interesting to understand why an algorithm performs better (or worse) than some other algorithm than to just observe the fact that it does.

The most important performance measure in this study is the number of collision checks performed by the algorithm in solving the test problems. The collision check is usually the most expensive operation in the motion planner (Latombe 1999). This is especially the case when the planning is done for realistic industrial applications, which involve complex geometry (Bohlin and Kavraki 2000). The number of performed collision checks is often good measure of the amount of *cspace* that has to be examined during planning. The number of collision checks as a measure has an advantage that it is resistant to changes in computer technology and implementation details. An implementation should give similar results on any computer and different implementations of an algorithm should give qualitatively similar results. The number of performed collision checks belongs to a class of measures often called structural measures (e.g. Moret 2002).

The user of an algorithm is, however, most often interested about the running time of the algorithm on the problem instances he or she is likely to want solved. The number of performed collision checks is indicative of the running time, but it does not account for the computational cost of the actual motion planning algorithm. This point is especially salient when comparing different algorithms and implementations against each others. Some algorithms may be efficient in terms of performed collision

checks, but require expensive maintenance of a supporting data structure that consumes the running time saved in reduced collision checking. Data structure choices and implementation issues may have significant effect on the running time of an implementation, but these are not visible in the number of performed collision checks. This issue is well known among the researchers of search algorithms. Experimentation with parallelized algorithms is an important special case, since parallelization is intended to reduce the wall-clock running time of the motion planner. For parallel motion planner the running cost in terms of CPU time or performed collision checks may even increase over the serial version, especially if a speculative execution strategy is used to keep all computational resources busy during the planning. For these reasons, the measures in performed collision checks are complemented with actual running times when appropriate or necessary. Furthermore, run-time is the most common performance measure reported in the literature. Despite the difficulties in using the running time in comparing between algorithms and implementations, it is an important simple measure for assessing the algorithm.

The amount of collision checking and running time are measures of the run-cost of a motion planner. It is often recommended to measure also the quality of the solution (e.g. Crowder et al. 1978; Moret 2002). This is important issue for many applications where the solution is to be used by some physical system. Solution quality has received attention in robotics motion planning research (e.g. Overmars 2002), but majority of the effort has been put into development of algorithms that accept any solution. Properties, such as path length, and smoothness, have been used to characterize solution quality. Many of the current motion planners are randomized, and they can produce solutions that can have very poor quality in these terms. The solution quality is often not addressed at all or left to be handled in a postprocessing or optimization stage after a collision-free motion is produced by the motion planner (e.g. Latombe 1991, p. 348; Berchtold and Glavina 1994). The situation is somewhat different with preprocessing type of PRM planners, which provide the solution as graph that approximates the connectivity of the free *cspace*. An obvious measure for the quality of the roadmap is that the roadmap should have exactly one component for each connected component of the free *cspace* (Hsu et al. 1999a). Other qualitative goals have been proposed e.g. minimizing the number of nodes in the roadmap that is necessary to obtain some coverage (Nissoux et al.1999).

The measures of solution quality and running cost can be combined by studying what is the solution quality attained within a given running cost limit. This was the approach taken in paper IV. For probabilistically complete algorithms this question takes the extreme form of what is the probability of attaining any solution within a given running cost. This question is important in robotics, when real-time or near real-time planning is necessary, but the path quality can be sacrificed. The relevant performance quality of an algorithm here is its reliability. Crowder et al. (1978) define reliability as the size of the class of problems the algorithm can solve. Randomized algorithms often have a property that their ability to solve problems can vary within a domain and even from run to run. This can be manifested in running cost, solution quality, or both. The reliability of a randomized algorithm is best characterized by its run-cost or solution quality distributions, but those are often difficult or expensive to obtain, even for just one or few test problems. The distribution can be sampled experimentally by running the implemented algorithm. The empirical distribution can then be presented as a histogram or statistical properties of the distribution can be estimated to characterize the reliability. See paper VI for further discussion of this matter.

The selection of test problems plays an important role in the experimental performance evaluation of algorithms. Goldberg (1999) presents guidelines for selecting test problems for determining, explaining, and predicting the performance of general-purpose algorithm implementations. His emphasis is on graph algorithms, but the general principles are more widely applicable. For testing algorithms, real-life

problems are highly desirable, but often difficult to find. Even when a library of real-life problems is available, synthetic problems can be useful in exploring the algorithm's strengths and weaknesses in detail, and in testing for anticipated future applications. Synthetic test problems can be generated to have particular problem or solution structure. Both easy and hard problem families should be used to obtain both upper and lower bounds for algorithm's performance. Of special interest are problem instances, which are hard for one algorithm but not for another. Goldberg calls such problems **separators**. Such instances are important not only in establishing relative algorithm performance, but also in predicting and explaining the difference. Widely accepted problem sets make it possible to compare published results and evaluate new algorithms. However, they also bring the risk of specializing the algorithms for the problem set while compromising the performance on other relevant problem types. The problem sets need to evolve to match bigger and faster computers and new algorithms.

The evaluation problems used here include benchmark problems from the literature. These are problems intended to present interesting problem categories and proposed for general testing, not just for demonstrating the properties of a particular motion planner. Using benchmark problems makes it possible to compare the performances of different algorithms and variations. To avoid getting the algorithm tuned for optimal performance on the benchmarks, they are complemented with additional problems with some interesting properties. These complementary problems have been designed for testing this particular algorithm, but some of them should prove interesting for the wider research community. They have been designed to have properties that are difficult to deal with previous techniques. Easy test problems are needed to be able to verify that the heuristics introduced in the chapter 5 improve performance over the previous heuristics. The easy test problems are presented in figures 1-3. They are two pick-and-place type of tasks for 6 *dof* and 5 *dof* manipulators and a problem for a 6 *dof* free-flying robot.

Among the test problems are two well-know benchmark problems proposed in the literature. The Hwang and Ahuja benchmark problem is a 5 degrees-of-freedom robotics motion planning problem for a SCARA-type robot (Hwang and Ahuja 1992). The task was designed to represent a realistic but non-pathological problem for a manipulator. The task involves removing a hook from a wicket and a subsequent backtracking motion to avoid a large obstacle (see figure 4). No generally available geometric model for the task exists, but several versions of the problem were made for this thesis. The versions differ in the depth of the notch made in the L-shaped large obstacle. The depth controls the amount of free-space available in between of the high obstacle and the robot body. The narrower the free-space the more difficult the problem is, since the motion planner must find a path through the bottleneck. The problem versions are named with *Adept* prefix followed by a numerical value for the depth of the notch in millimeters. The problem becomes unsolvable around the depth of 75mm.

The second benchmark problem is the Alpha Puzzle benchmark problem proposed by Amato et al. (1999). The problem is intended to represent 6 *dof* disassembly problems and it is designed to have a narrow passage. The original Alpha Puzzle problem involves separating the two intertwined loops. The loops can be intertwined in two different ways with the prongs of the loops either in symmetric (figure 5a) or anti-symmetric (figure 5c) orientations. Several versions of the Alpha Puzzle exist with varying difficultness. A smaller version number indicates a more difficult problem. Versions 1.5 and 1.2 can be solved with a "sliding" motion[2] and they are considerably easier than versions 1.1 and 1.0, which must be solved with a "twisting" motion[3]. These

[2] http://www.laas.fr/~nic/Move3D/amato15.mpg
[3] http://www.kuffner.org/james/plan/movies/alpha1solution.mpg

harder Alpha Puzzles are separator problems: only a few algorithms have been able to solve them.

Several many degrees-of-freedom problems are constructed by combining multiple robots into a single system. A more difficult task is obtained from Hwang and Ahuja benchmark problem by requesting the same motion but simultaneously for two robots making it a considerably more difficult 10 *dof* problem (no figure shown). The same technique is used to construct 12 *dof* and 18 *dof* problems from several 6 *dof* Puma type problems. These problems have multiple kinematic chains and they are narrow passage in a narrow passage type problems: one has to find a "narrow passage" in subspace that is itself caused by a narrow passage. Furthermore, the Pumas have to avoid collisions among themselves, so the geometric properties such as medial axes of the work space change frequently. These problems should be difficult for structured sampling techniques. These problems also test the feasibility of the motion planners for centralized multi-robot planning. The last test problem in the set is a Puma with large payload inside a cage like construction. The free space around the robot is rather constrained and there are multiple long winding tunnels in the *cspace* of the problem. Motion planners with simple local planners would have difficulties in escaping the tunnels.

The inherent limitation of experimental assessment of algorithms must be acknowledged here: it is difficult to generalize beyond the specific test problems used in the experiments. No generalization is claimed or implied here beyond that obtained by using known benchmark problems which their designers introduced to be representative of certain classes of problems. The test problems introduced for this research are designed to be difficult for other similar motion planners rather than represent any particular problem class. It is interesting to note that benchmark problems are usually designed to be difficult for the dominant technique in the motion planners of the time. Hwang and Ahuja benchmark (1992) requires a large backtracking motion, which means that the planner has to escape a large local minimum. As described in the chapter 4, local minimum problem was an important issue in designing potential field planners. Alpha Puzzle benchmark has a narrow passage, a known difficulty for sampling based planners. The test problems designed specifically for the research presented in this thesis have been designed to be difficult for structured sampling techniques that have been proposed for overcoming the narrow passage problem and for planners that rely on simple local planners.

It is rather easy to design a problem that would be difficult for all sampling based motion planners. These planners assume that sample generation cost is small. The assumption can be violated by designing a test problem with an arbitrarily small fraction of free *cspace*, so that the probability of obtaining a collision-free sample configuration is very small. A simple example of such a problem is to plan the motion of a robot through tight winding tunnel inside a huge obstacle. Since collision-free configurations are found only in the tunnel, the probability of finding a free sample with uniform sampling can be made arbitrarily small by increasing the relative size of the obstacle with respect to the tunnel. It is a known fact that problems, which are difficult for sampling based motion planners, are found in applications. One such example comes from assembly planning (Amato et al. 1998c).

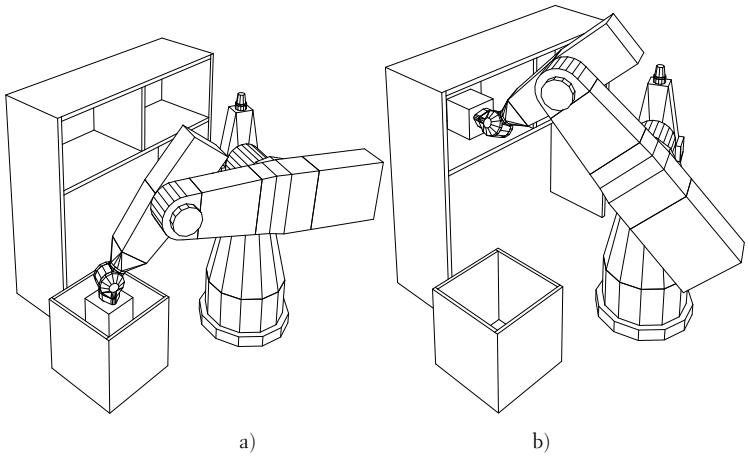

Figure 1: The start and goal configurations for a 6 dof Puma robot.

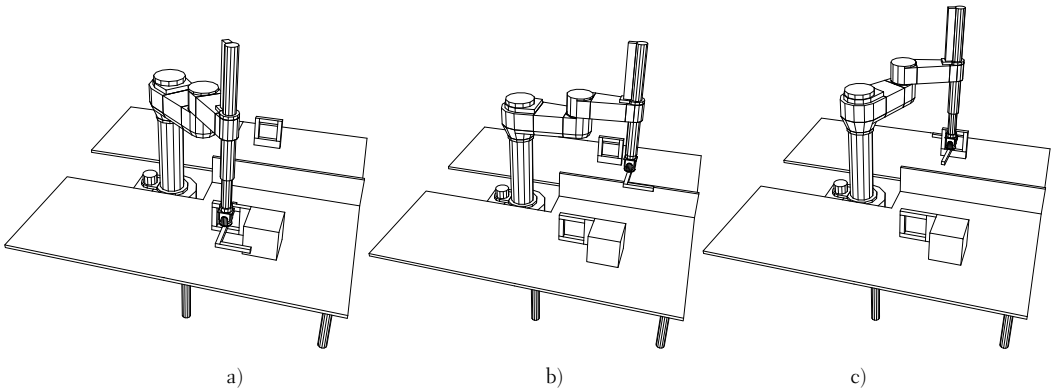

Figure 2: The start configuration and two goal configurations for a 5 dof SCARA robot.

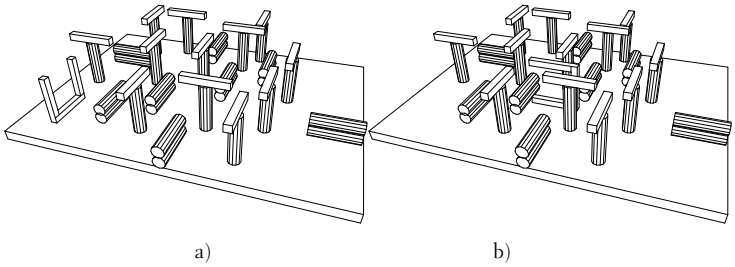

Figure 3: The start configuration and the goal configuration for a 6 dof U-shaped rigid body. The workspace has a height limitation that forbids the rigid body from moving above the obstacles.

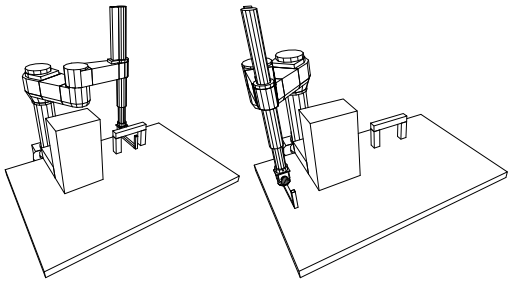

Figure 4: The start and goal configurations for a version of the benchmark problem proposed by Hwang and Ahuja. This figure presents the easiest version with passage depth of 400 mm.

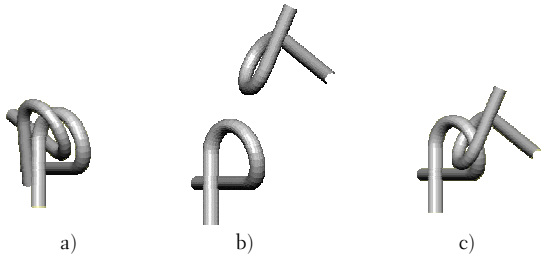

Figure 5: The configurations for the Alpha Puzzle benchmark task.

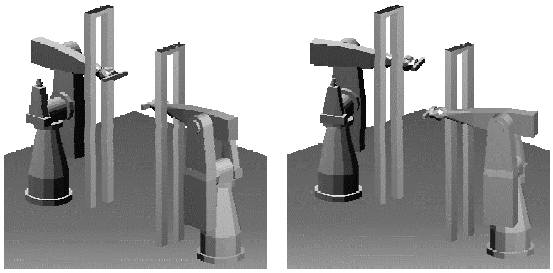

Figure 6: The start and goal configurations for the 12DOF task "dogs with bones".

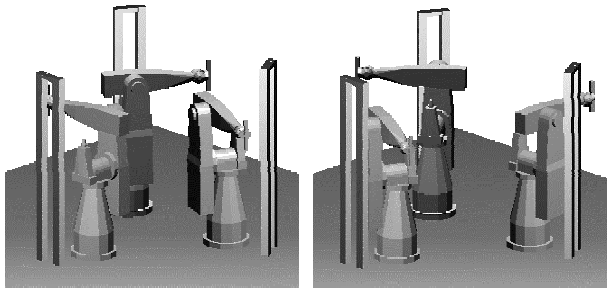

Figure 7: The start and goal configurations for the 18DOF task. The robots must avoid the gates and each others while performing a right hand rotation.

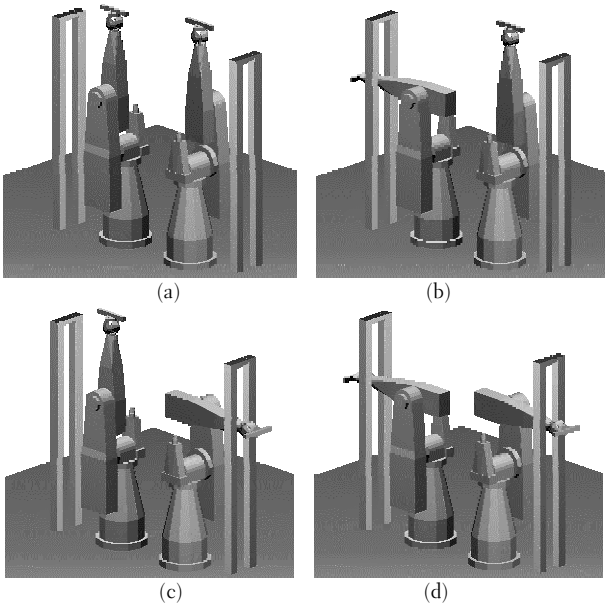

Figure 8: The seed configurations for the 12 degrees-of-freedom task with two kinematic chains.

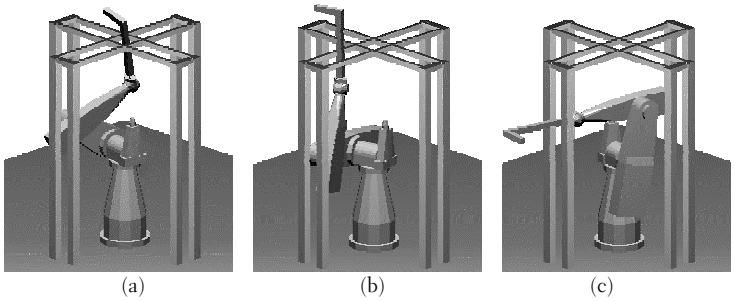

Figure 9: Some of the configurations for the 6 degrees-of-freedom task. The total number of seed configurations is 9, since all the configurations symmetrical to (b) and (c) by rotation of the base joint are used as seeds.

7. EMPIRICAL RESULTS

This chapter describes the experiments and data used to assess the motion planning variants in this thesis. First, relative straightforward experiment is performed to show empirically that the set of multiple-heuristics improves the performance of the planner in comparison to single fixed heuristics and the randomized multiple-heuristics used by Kondo (1991b). These results were published in paper I. An exploratory study is used to demonstrate that the parallel version of the planner can solve difficult many degrees-of-freedom problems at high resolutions and that it demonstrates an acceptable speed-up on a Pile-of-PCs type hardware. These results were published in paper III. The capability of powerful local planning in addressing the narrow passage problem is assessed by comparing it against a number of traditional simple local planners. These results were published in paper IV. The effect of the threshold parameter controlling the competence of the local planner is studied experimentally to observe if there is a systematic response. Experiments are also performed to test if the presented scheduling techniques improve performance. These results were published in papers V and VI, similar experiments are reported in paper II, but not presented here for the sake of brevity. Finally, the possible effect of the scheduling on the run-cost variance is tested. These results come from paper VI.

Table 1 shows the data for the first set of experiments[4]. For each problem, every degree of freedom of the robots was quantized into 100 discrete positions. The statistical data was calculated for at least 500 runs. None of the single heuristics above could solve every task in the table 1 in less than 100000 collision checks, while search with the multiple fixed heuristics found a solution to every task in less than 3000 collision checks.

The parallel motion planner was tested on a Linux PC cluster comprised of 11 processors with clock speeds between 450 MHz and 550 MHz and memory sizes of 128 MB or 512 MB. The computing nodes were connected with 100 Mbit Ethernet. For the scalability experiments the computing nodes were grouped so that each group has an average clock speed of 500 MHz. The implementation uses RAPID collision detection library (Gottschalk et al. 1996) and MPICH message passing library (Gropp et al. 1996). These experiments were performed with the exponential scheduling constants O'_{th} and R having values of 3 and 1.05.

The parallel algorithm is implemented by running the global planner in one of the processors and running copies of the local planner in the rest of the processors. While in the serial version of the algorithm the global planner launches one local planner at a time, the parallel version launches one local planner in each processor for solving different subproblems concurrently. The local planners remain based on the serial A* search algorithm. Thus, the parallel speed-up comes from computing the path segments in parallel in the processors running the copies of the local planners. As in the serial planner, the global planner combines the path segments from the successful runs of the local planners until a solution is found. Additional parallelism may be obtained by parallelizing the A* algorithm itself as done by Henrich et al. (1998), but this study investigates only the parallelization by running multiple copies of the local planner concurrently. Additional opportunities for increasing the parallelization of the algorithm are discussed in paper III.

As seen in table 2, the planner can solve the Hwang and Ahuja benchmark task (*Adept400*, figure 4) in seconds with a 296×171×42×191×105 grid representation of the *C-space* (label: 5DOF). The search resolution is the same as that of the similar "*AdeptOne*" task (Chen and Hwang 1998). A 2960×1710×420×1910×1050 high-resolution version of the task (HR5DOF) can be solved in few minutes using the whole cluster. The parallel planner can solve the Alpha Puzzle 1.2 task (figure 5) in minutes

[4] Note that the manipulator heuristics is slightly different in this experiment. See paper I.

with a resolution of 128 positions for each *dof* (6DOF). A high-resolution version with 1280 positions can be solved in tens of minutes on the whole cluster (HR6DOF). The 12DOF and 18DOF tasks are planned with 100 discrete positions for each degree-of-freedom. The data in table 2 shows that also these problems can be solved in minutes with the cluster.

The data in table 2 shows that the planner can have superlinear expected speed-up for some problems[5]. This can be explained by the fact that the serial version of the planner has to complete the local planning for all path segments serially, but the parallel version can run local planning for several segments concurrently. When the serial version is hampered by computing costly but unnecessary path segments, the parallel version can plan for other path segments in the additional processors and obtain the solution for the complete problem without finishing all the unnecessary segments. Not all of the problems have such behavior, thought. The scalability of the planner for one such problem is shown in figure 10. Even in such a case, the tested worst-case parallelization strategy provides an acceptable speed-up. Additional reduction in planning time could be attained by adding more processors to the cluster. However, the graph shows diminishing returns and eventually a larger granularity strategy must be used. See paper III for further discussion.

The preprocessing experiments are designed to compare the quality of the roadmaps produced by several local planners. The multi-heuristic A* local planner with path optimization is compared with more conventional local planners. Linear *straight-line* (*SL*) and *rotate-at-½* (*RAS*) local planners are selected as base line, since they were the recommended fast local planners in the Amato et al. study (2000). A simple greedy search local planner (*G*) is obtained from the multi-heuristic local planner described in the chapter 5 by setting $A=0$, $O_{th}=1$ and executing only the "even" heuristics. These planners have distinctively different power as the ability to proceed after they make contact with a *cspace* obstacle surface. *SL* and *RAS* fail as soon as contact is made. The greedy local planner can proceed by "sliding" along the *cspace* obstacle surface as long as any of the neighboring configurations improves the heuristic estimate and no backtracking is required. The multi-heuristic local planner can escape local minima by backtracking, but the extent of backtracking is limited by the threshold value O_{th}. Two constant values for the threshold are used in the experiments: 2 (M_2) and 32 (M_{32}). *SL* is arbitrary set to be more powerful than *RAS* in the analysis below.

The test problems with predetermined configurations are those in figures 4, 5bc, 8, and 9. Note that for the test problem in figure 9, the total number of seed configurations is 9, since all the configurations symmetrical to (b) and (c) by rotation of the base joint are used as seeds. This experiment uses Alpha Puzzle 1.0, which is the most difficult version in the family, but the seed configuration is the antisymmetric intertwined configuration rather than the symmetric in the original Alpha Puzzle problem. The grid sizes for the multi-heuristic local planner are 512^{dof} for the Alpha Puzzle 1.0 and 128^{dof} for the other tasks. The step size for the *SL* and *RAS* was selected so that the discretations are comparable.

The performance metrics for this experiment are the number of components in the final roadmap and the frequency of roadmaps that have paths between all of the seed configurations. The task for the planner is to construct a roadmap for a work cell within a roadmap size limit of 1000 nodes and a construction time limit of 60 minutes for the Hwang and Ahuja task and 360 minutes for the other tasks starting with the predetermined seed configurations. Fifteen replicates are produced, each with a different pseudo-random sampling sequence. When applicable in the analysis, the sampling sequences are used as a blocking factor to eliminate the variability among the sequences. The experimental design is a randomized complete block design (Montgomery 1997, p. 171-191) with work cell, local planner and sampling sequence as fixed factors

[5] This observation was made by Professor Seth Hutchinson.

The average number of components in the final roadmap and the number of successfully connecting the seed configurations are presented in the table 3 for both size and time constrained preprocessing runs. Analysis of variance (ANOVA) was performed to the data with the number of final components as the response. The statistical model is significant ($p<0.0001$) and it explains 99% of the variance. The analysis reveals that the local planner is a highly significant factor ($p<0.0001$) with a highly significant interaction ($p<0.0001$) with the task in both size and time constrained experiments.

ANOVA test can detect that there are significant differences among the local planners, but other statistical techniques are needed to reveal which local planners differ. A set of contrasts is used to test several focused hypotheses. The fast local planners as a group are compared to the powerful local planners as a group (*SL* and *RAS* vs. *G*, M_2, and M_{32}). Similarly, the backtracking local planners are compared to the non-backtracking local planners (M_2, M_{32} vs. *SL*, *RAS*, *G*). In order to study the significance of "sliding" performed by *G*, it is compared to *SL* and *RAS* as a group. In each case, the null hypotheses of no difference between group means can be rejected in favor of an alternative hypothesis of group means differing significantly ($p<0.0001$) for both sets of experiments.

The Ryan-Einot-Gabriel-Welsch multiple comparison procedure (Einot et al. 1975) can declare a very significant ($\alpha=0.01$) difference between the means of each pair of local planners in the size constrained runs, but fails to declare the difference between the means of M_2 and M_{32} in the time constrained runs statistically significant at $\alpha=0.05$ (experimentwise). Jonckheere-Terprstra test (Pirie 1983) provides strong support ($p<0.0001$) for the significance of the decreasing trend in the number of final components as the local planner gets more powerful. The significance of the differences between average numbers of final components are tested separately for each task with the least significant difference test at $\alpha=0.05$ (comparisonwise) and indicated in the table 1 of paper IV.

Cochran Q test (Siegel 1956) detects highly significant ($p<0.0001$) differences among the success rates of the local planners for both sets of experiments. In particular, McNemar's test (Siegel 1956) detects significant differences between M_2 and M_{32} in both the size ($p<0.0001$) and time ($p=0.0004$) constrained experiments. Furthermore, Jonckheere-Terprstra test declares the increasing trends in success rate highly significant ($p<0.0001$) for both sets.

Table 4 presents descriptive statistics of the number of configurations stored with the roadmap edges after a path segment produced with a powerful local planner has been optimized. The data shows that on the average only a small number of configurations have to be stored in each edge.

The effect of the scheduling techniques is investigated in two phases. First, the effect of the threshold O_{th} is investigated experimentally to determine if any systematic response can be observed. In order to not confound the effect of the scheduling with that of multiple-heuristics, the experiment is done with a uni-heuristic local planner that executes only a greedy heuristics based on Manhattan distance (see paper V). Roadmaps are constructed for the test problems with the uni-heuristic PRM planner until a roadmap connecting the seed configurations is obtained. The number of collision checks performed during the roadmap construction is used as a cost measure. Motion planner variants that have median run-cost higher than 10^8 collision checks are disqualified. This amounts to setting a lower limit on the accepted reliability of the planner. The median estimates the 50% point of the run-cost distribution. Thus, only algorithms that have 50% chance of yielding a solution in 10^8 collision checks are accepted.

Tables 5 and 6 present the average construction costs for the test problems at various fixed levels for the threshold O_{th}. Additionally, table 5 presents construction costs for a naive PRM planner, which tries to connect the nodes with the *straight-line* local planner (labeled *SL*). The data from table 5 for the versions of Hwang and Ahuja

problem is visualized as a surface in figure 11. It can be seen that the performance of the PRM planner depends strongly on the threshold value, especially for the more difficult versions of the problem. Furthermore, there is a minimum cost "valley" on the surface across the versions of the problem. The interpretation of the data suggests that the best value for the threshold depends on the difficulty of the problem with higher values preferred for more difficult versions. The data in table 6 presents average roadmap construction costs for the Alpha Puzzle benchmarks. The Alpha Puzzle data indicates behavior similar to the one observed for the Hwang and Ahuja problem, but here the valley seems to locate at higher levels of O_{th}. The planner with O_{th} value of 1 was disqualified for both Alpha Puzzle versions 1.0 and 1.1, and the planner with O_{th} value of 2 was disqualified for Alpha Puzzle 1.0.

The finding of a possible minimum cost valley in the cost surface motivates development of techniques, which can take advantage of the information gathered during the roadmap construction and utilize that information to reduce the construction cost. The adaptive strategies described above are steps to this direction, as they use measures of the difficultness of the problem to adjust heuristically the local planner.

Using a metaplanner to select the heuristic parameters is tested next. Due to computational cost, the experiment is restricted to *Adept80* and Alpha Puzzle 1.2 problems. Table 7 gives descriptive statistics for the experiments with the PRM variants. The table gives run-cost mean, standard deviation and the coefficient of variation for a sample of 240 runs. The coefficient of variation expresses standard deviation as a percentage of mean, so it can reveal if the standard deviation changes together with the mean.

As can be seen in table, PRM-G and PRM-L have considerably better performance than PRM-C both in terms of mean run-cost and the standard deviation of the run-cost for the version of the Hwang and Ahuja benchmark problem. PRM-L not only has the absolute standard deviation improved but also the coefficient of variation is smaller. This indicates that not only has the scale of the run-cost distribution changed but also its shape. The existence of a statistically significant difference in the standard deviations is confirmed by Levene's test of homogeneity of variances (Glaser 1983), which detects a very significant difference ($p<0.001$). The Ryan-Einot-Gabriel-Welsch multiple comparison procedure assumes equal group variances. It is therefore not suitable for use now that the variances are known to be different. Dunnet's C test does not make this assumption (Dunnet 1980). When comparing the means, it can declare the difference between static (PRM-C) and adaptive (PRM-G, PRM-L) planner variants statistically significant at $\alpha=0.01$ (experimentwise), but fails to detect statistically significant difference between the two proposed scheduling heuristics. For the Alpha Puzzle 1.2 problem the results are not as good. There is an improvement in standard deviation when using the proposed scheduling heuristics, but that difference is not statistically significant. Neither are the differences in the means.

Task	Manipulator	Position	Rotation	Even	Multiple fixed	Multiple randomized			
						ave	min	max	Std
1a→1b	202355	27104	710	1764	909	7092	276	262664	20599
2a→2b	26190	>376384	418	7920	880	2255	312	42066	3456
2b→2c	3149	>330338	4463	3822	2640	4489	643	129605	8074
3a→3b	>403244	109306	100180	>460623	539	101800	302	397498	116410

Table 1: Results for the various heuristics. Labels in the first column refer to figures above. The sample size is at least 500 runs.

Task	CPU's	Min.	25%	50%	75%	Max.	Ave.	Std.
5DOF	1	2.7	14.7	23.1	34.6	158.0	28.5	22.4
	11	0.6	1.6	2.3	3.1	11.7	2.6	1.6
HR5DOF	1	75.7	216.0	390.5	623.0	3609	500.0	470.0
	11	21.8	41.3	58.0	82.5	312	68.6	42.1
6DOF	1	14.1	209.5	450.0	909.5	5636	764.4	956.9
	11	0.8	15.2	32.2	66.6	417	54.7	69.6
HR6DOF	1	23.5	1302	4171	7407	21377	5045	4666
	11	12.1	120	331	655	1855	444.4	409.1
10DOF	1	45.9	498.5	1117.5	2579	10648	1981	2272
	11	9.3	42.0	96.6	183.5	987	149.1	181.0
12DOF	1	10.9	286.0	940.5	5553	34974	4112	6300
	11	4.5	17.5	58.1	305.0	1945	225.6	345.6
18DOF	1	31.2	476.5	1297	3143	25584	3330	5026
	11	10.8	38.0	82.0	186.5	1661	214.8	325.3

Table 2: Run times in wall clock seconds for the various tasks on a single 500 MHz CPU and on the whole cluster of 11 CPUs. Sample size is 100 runs. Percentiles are rounded up. The task labels are explained in the text.

	Size Constrained									
	Fig. 4		Fig.5bc		Fig. 8		Fig. 9		Ave.	Σ
RAS	4.9	2	3.1	0	16.0	0	73.7	0	24.5	2
SL	3.5	10	3.2	0	14.3	0	56.1	0	19.3	10
G	1.5	15	2.5	0	4.7	0	19.7	0	7.1	15
M_2	1.1	15	2.3	1	2.9	12	8.3	0	3.6	28
M_{32}	1.1	15	1.6	7	2.3	15	2.2	15	1.8	52
	Time Constrained									
	Fig. 4		Fig.5bc		Fig. 8		Fig. 9		Ave.	Σ
RAS	7.7	15	41.2	0	28.7	0	127.3	0	51.3	15
SL	3.1	15	20.9	0	28.6	0	102.7	0	38.8	15
G	1.2	15	6.1	0	5.1	0	27.1	0	9.9	15
M_2	1.3	15	2.9	1	2.5	12	7.2	0	3.5	28
M_{32}	1.7	15	2.0	6	1.9	10	2.2	15	2.0	46

Table 3: Average numbers of final components and the numbers of successful merges of the seed configurations into the roadmap. Grand averages and total sums over all tasks are presented for the final number of components and the number of successes, respectively. Sample size is 15 runs.

	Fig. 4		Fig.5bc		Fig. 8		Fig. 9	
	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.
G	2.11	0.05	2.18	0.03	2.51	0.03	2.38	0.04
M_2	2.27	0.12	2.35	0.04	2.94	0.05	3.04	0.10
M_{32}	2.72	0.10	2.53	0.12	3.38	0.26	4.74	0.72

Table 4: Averages and standard deviations of the numbers of configurations stored with a roadmap edge after the optimization of roadmap path segments. The data is the combination of size constrained and time constrained runs. Sample size is 30 runs.

Task	O_{th}						SL
	32	16	8	4	2	1	
Adept80	5,446,407	2,986,472	1,706,875	904,616	548,688	3,389,542	6,642,256
Adept100	1742,697	1170,693	640,290	517,955	279,917	830,774	635,328
Adept200	542,358	301,870	167,786	120,968	95,044	123,068	292,904
Adept300	261,297	150,627	129,283	86,150	50,768	58,584	272,631
Adept400	203,713	112,787	61,117	38,961	25,500	23,974	259,681

Table 5: The average numbers of collision checks performed by the uni-heuristic PRM planners for the versions of Hwang and Ahuja benchmark problem. Each data point represents an average of at least 15 runs.

Task	O_{th}					
	32	16	8	4	2	1
AP 1.0	226,692,577	138,243,470	94,844,153	76,840,334	—	—
AP 1.1	69,032,080	41,777,409	32,771,501	65,773,256	72,028,029	—
AP 1.2	1,729,588	1,224,763	1,127,312	1,166,626	1,908,043	156,882,356
AP 1.5	343,581	220,483	144,205	110,641	85,539	192,598

Table 6: Average construction costs (collision checks) for the versions of Alpha Puzzle problem. The sample size is at least 15 runs.

	Hwang and Ahuja problem			Alpha Puzzle 1.2		
	Mean	Std. Dev.	Coeff. Var.	Mean	Std. Dev.	Coeff. Var.
PRM-C	2,103,084	3,563,392	169	1,418,781	1,023,970	72
PRM-G	804,865	1,357,438	169	1,473,386	924,775	63
PRM-L	816,762	800,089	98	1,357,358	921,015	68

Table 7: The mean and standard deviation of the run-cost in collision checks and the coefficient of variation for the problems of figure 4 and figure 5. The sample size is 240 runs.

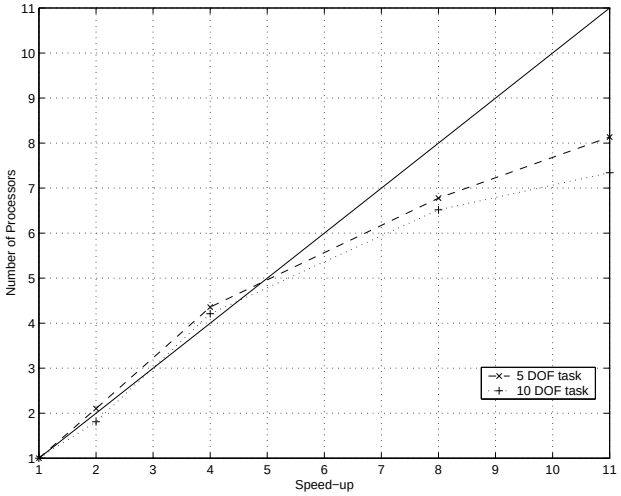

Figure 10: Speed-up for tasks 5 dof Hwang and Ahuja benchmark task and on the 10 dof doubled Hwang and Ahuja. The speed-up is calculated from the average run time for 5 runs of the planner.

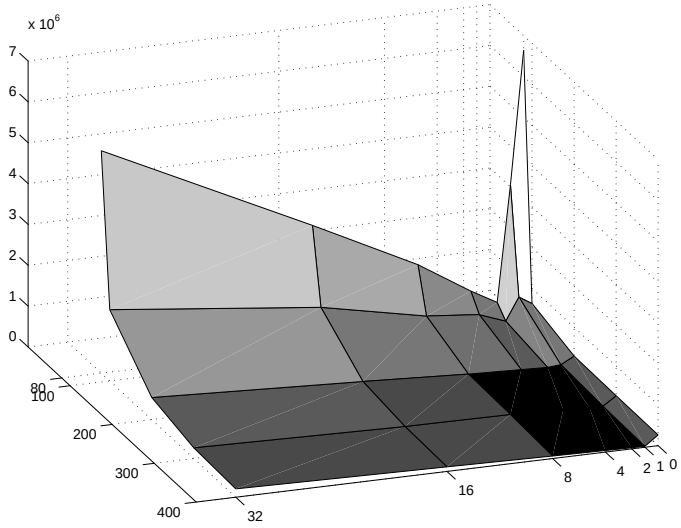

Figure 11: Cost surface for various versions of the Hwang and Ahuja benchmark problem.

8. DISCUSSION

After presenting the experimental data in the previous chapter, it is now possible to assess the performance of the various components investigated in this thesis. First, the descriptive statistics for the various heuristics indicate that combining the presented fixed heuristics makes an efficient planner with a reliable performance. Both the rotation heuristics and multiple fixed heuristics are the best in two test tasks, but the multiple fixed heuristics has a good performance over all the test problems unlike any of the single heuristics. It should be noted that the performance of the fixed heuristics is better than the average performance of the randomized heuristics. This demonstrates that the multiple fixed heuristics is indeed capable of utilizing the information about typical kinematic structures of the robots that have been coded into the weight sets.

The experiments with the parallel implementation demonstrate a moderately good scalability of the algorithm on an inexpensive parallel computer built form commodity hardware and free software. The superlinear expected speed-up observed for some test problems demonstrates that, at times, the parallel version can better take advantage of the task decomposition provided in the subgoal network. While the algorithm shows diminishing speed-up as the number of processors is increased, alternative parallelization strategies may be used to provide improvement in the behavior of the planner, see paper III for a discussion. Even in the current form, the parallel implementation can be used to solve very difficult motion planning problems within near real-time and practicable off-line time limits. The test problems included cases with many degrees-of-freedom demonstrating that A* search-based approach can be used to solve such problems. Additionally, easier but non-trivial problems can be solved with exceptionally high-resolution representation of the *cspace*.

In the preprocessing experiments, as expected, more powerful local planners generate fewer components and have a higher success rate than weaker local planners when applied to the same samples without a run-time limit. The main result of the size-constrained experiments is an empirical confirmation for the assignment of relative power to each local planner, especially for *SL* and *RAS*.

On the other hand, the time-constrained experiments provide relevant information about the time efficiency of the local planners in producing roadmaps with few components. According to the statistical analysis, the local planner should be able to "slide". Backtracking capability further improves the time efficiency of the local planner, at least when combined with efficient heuristics to guide the search. As a group, the powerful local planners are more efficient than the traditional fast local planners. Furthermore, there is a statistically significant trend of increasing efficiency, as the local planner gets more powerful.

The success rates of the planners increase together with their power. Since connecting the seed configurations requires paths through narrow space, these results indicate the effectiveness of the planners in addressing the "narrow passage" problem. The ceiling effect observed with the Hwang and Ahuja benchmark problem demonstrates that also the simple local planners can deal with relatively easy problems. The difficult problems, however, are solved only with backtracking search.

The results on the relative effectiveness of *SL* and *RAS* observed here complement the earlier results of Amato et al. (2000). They constrain roadmap size and use the number of connections as the performance metric. When comparing the average number of connections for size-constrained Alpha Puzzle rigid body task, the results agree, but in the experiments presented here the difference is not statistically significant ($p=0.74$). For the robot tasks *SL* is significantly better than *RAS*, except for the time constrained 2 pumas task. In that work cell, *RAS* behaves like *SL* for each robot separately. The general observations of Amato et al. are similar to those presented here: powerful local planners work best for difficult problems. However, the path transformation technique presented here allows the use of local planners sufficiently

powerful to solve even the most difficult problems in this study, while still maintaining rapid query processing.

Storing the configurations needed for the reconstruction of the path segment increases the number of configurations in the final roadmap. However, since there is on the average only a few configurations needed for each edge and the used connection strategy produces approximately one edge per node in the roadmap, the increase in the roadmap size is moderate. The increase in memory consumption is quite insignificant with the amount of memory available in contemporary computers. If the roadmap size becomes an issue, it can be effectively dealt with by the visibility roadmap approach.

When analyzing the results from experiments with the scheduling metaplanners, it can be seen in the data for the Hwang and Ahuja benchmark problem that an improvement in average performance and reduction in run-cost variance is possible when using dynamic scheduling. The collection of test problems is rather small, but unless contradicting evidence emerges, PRM-L can be recommended.

The failure of the scheduling heuristics to yield statistically significant differences for the Alpha Puzzle 1.2 may be explained by the fact that it is very difficult to generate "good" subgoal samples for this task. Finding critical samples for this problem from a pseudo-random sampling sequence is a rare event and the increase in the capability of the local planner fails to make it sufficiently more frequent. It can be stated that the behavior of the planners on this test problem is determined by the "narrow passage" nature of the problem, as samples are required in a small bottleneck area in the *cspace*.

The ability of the A* based local planner to solve the Alpha Puzzle 1.0 even with very straightforward heuristics, and simplistic sampling and connection strategies is quite significant. Traditional local planners require the use of a sophisticated sampling strategy or the visibility roadmap approach to solve the Alpha Puzzle family of benchmarks.

An understanding of the success of PRM construction with a powerful local planner can be gained by noting that because of the sliding and backtracking capability, a powerful local planner increases the set of configurations reachable from a given start configuration. This means that the visibility sets of configurations are larger and the apparent expansiveness of the *cspace* is increased. A full theoretical analysis is beyond the scope of this thesis, but for a relevant discussion of the theoretical concepts, see (Hsu et al. 1999).

Although empirical comparison of algorithms is a non-trivial matter, it is irresistible to do some "quick-and-dirty" comparisons to the results available in the literature. With respect to the Hwang and Ahuja benchmark task the situation is simple: there are no other results to compare against. Chen and Hwang use a very similar, but easier AdeptOne task to demonstrate the performance of their SANDROS planner. SANDROS performs 10,079 collision checks in solving the demonstration task (Chen and Hwang 1998). That result is in the same order of magnitude as the proposed algorithms perform for the easy versions of the Hwang and Ahuja benchmark task.

More comparable results have been published for the Alpha Puzzle benchmarks. Amato's group has published running times for solving Alpha Puzzle 1.5 and 1.2 on HP V2200 computer (Vallejo et al. 2001). Solving Alpha Puzzle 1.5 with OBPRM takes 3495 s and with their version of Adriane's Claw algorithm 1699 s. Solving Alpha Puzzle 1.2 with their version of RRT takes 49455 s. In their experiments only their composite planner SS can solve both versions with running times 1699 s and 68934 s. Caselli et al. (2002) publish running times for solving Alpha Puzzle 1.5 with their potential field planner (PFP) and Kavraki's PRM on Pentium II 450 MHz processor. The average running times are 81.26 s and 281.46 s, respectively. The data in table 2 gives the average running time for solving Alpha Puzzle 1.2 with the parallel planner running on one 500 MHz processor as 764 s.

The results in paper IV and paper V seem to be the only published results for Alpha Puzzle 1.1 and Alpha Puzzle 1.0. However, on their WWW pages, Amato's research group states to have solved Alpha Puzzle 1.1 (Amato 2003), and Alpha Puzzle 1.0 has

been solved using RRT (Kuffner 2002) and PRM with RRT as the local planner (Siméon 2001). Kuffner (2002) gives 8 minutes as the fastest planning time on a 1.7 GHz Pentium III PC, but does not present average running time.

Some aspects of the algorithm proposed here are left unexplored. A* search algorithm has exponential memory consumption, which is controlled in the proposed algorithm indirectly by setting the O_{th} limit on the efficiency and directly by setting a hard-limit on the memory consumption for each call of the local planner. While non-trivial multimovers problems with multiple 6 *dof* manipulators can be solved with the current algorithm, a large enough number of degrees-of-freedom will eventually make the approach impractical. No such problems were constructed to specifically find out the combination of largest practical number of degrees-of-freedom and task difficulty that can be solved with the available hardware. An algorithmic solution to the memory consumption is to use a restricted memory search algorithm such as one presented by Russell (1992). A restricted memory search algorithm will find the solution to the problem within a given amount of memory if the amount is large enough to store the solution path. However, it is not clear how to implement a restricted memory algorithm with multiple heuristics, and preliminary experiments indicate the benefit from multiple heuristics could be reduced significantly (Isto 1996).

A related issue is that the grid representation of the *cspace* in the current algorithm is uniform. The resolution of the grid must be high enough to represent the smallest relevant details of the *cspace*, but high resolution is superfluous in areas of large amount of free-space. Although the results in paper III indicate that the current algorithm is capable of solving problems with unprecedented search resolution, a multi-resolution search algorithm such as one presented by Autere and Lehtinen (1997) would be more efficient. The necessary resolution at each point in the *cspace* can be computed with the robot's Jacobian from the distance to the nearest obstacle (Paden and Mees 1989). It should be noted that the same method (or some approximation) should also be used when performing the local planning with an interpolation-based local planner unless the swept volume is computed.

The relative merits and the interplay of a powerful local planner and a structured sampling strategy is an interesting open question. However, some inferences about the relative merits can be drawn from the fact that all the motion planners that have succeeded in solving the Alpha Puzzle 1.0 have used some powerful search procedure (A* or RRT). Geraerts and Overmars (2002) found out that combining techniques that are good individually can result in inferior performance. It is therefore difficult to estimate the possible effect of combining a powerful local planner and a structured sampling strategy without performing a theoretical or empirical study.

The cost surface of a problem over the set of interesting parameters was used to demonstrate that the performance of the motion planner depends on the power of the local planner and to motivate the study of scheduling methods. The cost surface can be used to tune the parameter values for a particular class of problems, but it is non-trivial to use it during solving the motion planning problem, since the information is only available after a number of problems have been solved. The scheduling methods presented here show some benefits, but they are only first steps toward inventing methods for run-time adjustment of the parameters. More research in this issue is certainly needed.

Many of the extensions of the basic motion planning problem were left out of the scope of this thesis. This is largely because of the programming effort necessary for implementing the algorithms, test problems and support software. The search based local planner can be extended to handle many of the proposed extensions of the basic motion planning problem by applying known ideas from the literature. A time-varying problem with moving obstacles can be solved by replacing the A* search in configuration space with search in configuration-time space as described by Latombe (1991, p. 22-23, 357-373). Problems with conformable objects other than the manipulators studied here can be solved provided that the possible increase in the

number of degrees-of-freedom is handled as described above and a suitable heuristic function can be developed. This is particularly true for planning for flexible objects, since the computation of feasible bends for the objects is computationally expensive (Lamiraux and Kavraki 2001), and the number of different bends should be kept small by the heuristics. Problems with danger zones (Sent and Overmars 2001) can be handled most easily by augmenting the heuristics with a penalty function that keeps the A* search away from the danger zones as much as possible. Problems with nonholonomic constraints are relatively well understood (Laumond et al. 1998). A possible approach to use A* as a local planner for non-holonomic robots is to use it to search the space of possible inputs rather than directly the space of possible configurations. Motion planning for devices with closed kinematic chains can be done by applying the idea proposed by Han and Amato (2001) and breaking the chain into an active part and a passive part and letting the A* local planner drive the active part of the chain. The same idea has been utilized also in manipulation planning (Siméon et al. 2002) which is a motion planning problem with a movable object. However, all the above ideas remain unverified, and it can be expected that filling in the details will require a non-insignificant effort.

9. CONCLUSIONS

This thesis constructed a probabilistic roadmap motion planner for free-flying robots and manipulators. The planner can take advantage of the kinematic structure of the typical robots considered in this thesis, and improve the performance of the algorithm with presented heuristics. Despite using a potentially memory exhausting A* search based local planner, the motion planner can solve difficult problems with many degrees-of-freedom. Additional improvement in running time can be obtained by running the motion planner on an inexpensive and readily available parallel computer.

A new approach for two-stage probabilistic roadmap planning was presented that allows the use of powerful local planners during the roadmap construction. The approach separates the local planner used during the roadmap construction stage and the local operator used to reconstruct the roadmap connections during the query stage. Experiments with a multi-heuristic local planner and difficult problems demonstrated the effectiveness of the method in producing roadmaps that capture the connectivity of the *cspace* even in the absence of a sophisticated sampling strategy. The results indicate that the use of powerful, backtracking local planners should play a role in solving the narrow passage problem.

A heuristic technique was developed to control the capability of the local planner. The technique comes largely out of necessity, but it has an advantage that it can reduce the run-cost variance of the planner. Due to the limited success of the proposed variance reduction heuristics, their value is more in setting the stage for future improvements than in solving the variance problem.

The results on Alpha Puzzle 1.2 presented here are an improvement over previous published results. To the best knowledge of the author, the results for Alpha Puzzle 1.1 and 1.0 are the only published results.

The contribution of this thesis within the third paradigm of the computer science, theory, is minimal. It would be very interesting and desirable to have formal, theoretical explanations for the phenomena observed here. The benefit of a powerful local planner may be explainable within the established theoretical framework of probabilistic roadmap planners using concepts such as visibility and expansiveness. Explaining the variance reduction effect of the dynamic scheduling may require additional theoretical constructs. Developing an appropriate model and an analysis for explaining the empirical observations would probably be a thesis size undertaking in itself, so it is left outside of the scope of this thesis.

In addition to developing formal theories to explain the observed characteristics of various motion planners, and the open questions discussed in the previous chapter, two other issues for future research have emerged during this thesis project. In the future research in randomized motion planning, the full run-cost distribution of the planner should be considered. The end-user of the motion planning system will typically experience just a single run of the algorithm for each motion planning task. If the run happens to come from the far-right tail of the run-cost distribution, she would take little comfort from the explanation that averaging over repeated motion planning tasks would indicate better (expected) performance. The variance tends to be proportional to the expected run-cost. Robotic motion planning tasks, like the Hwang and Ahuja benchmark problem, can be solved near real-time with contemporary motion planning systems. However, the variance is a problem mainly because of the lack of well-understood performance guarantees required for real-time operation. Assembly tasks, such as those that inspired the Alpha Puzzle benchmark problem, tend to be more difficult, and therefore, the variance becomes a true usability issue. Restarting and heuristic techniques can provide improvements, but eventually derandomized and other deterministic methods should be developed.

As described in the chapter 3, applications from rational drug design may become increasingly important motivation for research in motion planning techniques. The

problems from biology can have degrees-of-freedom in thousands, and it is likely that the present algorithmic techniques are not scalable to such problems. Novel approaches, such as more knowledge intensive planning, may be necessary to tackle problems with very large number of degrees-of-freedom.

BIBLIOGRAPHY

Amato N.M. (2003) http://parasol-www.cs.tamu.edu/dsmft/benchmarks/alpha/, 19.5.2003.

Amato N. M., Bayazit O. B., Dale L. K., Jones C., Vallejo D. (1999) OBPRM: An Obstacle-Based PRM for 3D Workspaces, Workshop on Algorithmic Foundations of Robotics, A K Peters, Wellesley, 1999, 155-168.

Amato N. M., Bayazit O. B., Dale L. K., Jones C., Vallejo D. (2000) Choosing Good Distance Metrics and Local Planners for Probabilistic Roadmap Methods, IEEE Transactions on Robotics and Automation, Vol. 16, No. 4, August, 442 –447.

Amato N. M., Bayazit O. B., Kim K., Son W., Song G. (1998b) Providing Haptic 'Hints' to Automatic Motion Planners, Technical Report 98-026, Department of Computer Science, Texas A&M University, November 24.

Amato N. M., Jones C., Vallejo D. (1998c) An Adaptive Framework for 'Single Shot' Motion Planning, Technical Report 98-025, Department of Computer Science, Texas A&M University, November 24.

Amato N.M., Wu Y. (1996) A Randomized Roadmap Method for Path and Manipulation Planning, Proceedings of the 1994 IEEE International Conference on Robotics and Automation, IEEE, 113-120.

Autere A., Lehtinen J. (1997) Robot Motion Planning by a Hierarchical Search on a Modified Discretized Configuration Space, Proceedings of the 1997 IEEE/RSJ International Conference, Intelligent Robots and Systems, IEEE, 1208-1213.

Baginski, B. (1996) Fast Motion Planning in Dynamic Environments with the Parallelized Z^3-Method, Proceedings of the Sixth International Symposium on Robotics and Manufacturing ISRAM, ASME Press, 81-86.

Barraquand J., Ferbach P. (1994) Path Planning through Variational Dynamic Programming, Proceedings of the 1994 IEEE International Conference on Robotics and Automation, IEEE, 1839-1846.

Barraquand J., Latombe J.C. (1990) A Monte-Carlo Algorithm for Path Planning with Many Degrees of Freedom, Proceedings of the 1990 IEEE International Conference on Robotics and Automation, IEEE, 1712-1717.

Barraquand J., Latombe J.C. (1991) Robot Motion Planning: A Distributed Representation Approach, The International Journal of Robotics Research, vol. 10, no. 6, December, 628-649.

Basu S. (2003) On the Combinatorial and Topological Complexity of a Single Cell, Discrete and Computational Geometry, Vol. 29, No. 1, November, 41-59.

Berchtold S., Glavina B. (1994) Scalable Optimizer for Automatically Generated Manipulator Motions, Proceedings of the 1994 IEEE/RSJ/GI International Conference on Intelligent Robots and Systems, IEEE, 1796-1802.

Bessiére P., Ahuactzin J.M., Talbi E.G., Mazer E., (1993) The "Adriane's Clew" Algorithm: Global Planning with Local Methods, Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 1373-1380.

Blum C., Roli A. (2001), Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison, Technical Report No. TR/IRIDIA/2001-13 October 2001, Institut de Recherches Interdiciplinaires et de Développements en Intelligence Artificielle, Universite Libre de Bruxelles.

Bohlin R. (2002) Robot Path Planning, Thesis for the degree of doctor of philosophy, Department of Mathematics, Chalmers University of Technology.

Bohlin R., Kavraki L.E. (2000) Path Planning Using Lazy PRM, Proceedings of the 2000 IEEE

International Conference on Robotics and Automation, IEEE, 521-528.

Boor V., Overmars M.H., van der Stappen F. (1999) The Gaussian Sampling Strategy for Probabilistic Roadmap Planners, Proceedings of the 1999 IEEE International Conference on Robotics and Automation, IEEE, 1018-1023.

Branicky M., LaValle S. M., Olsen K., Yang L. (2001) Quasi-Randomized Path Planning. Proceedings of the 2001 IEEE International Conference on Robotics and Automation, IEEE, 1481-1487.

Brooks, R. (1983) Solving the Find-Path Problem by Good Representation of the Free Space, IEEE Transactions on System, Man and Cybernetics, Vol. 13, No. 4, 190-197.

Brooks R., Lozano-Pérez T. (1983) A Subdivision Algorithm in Configuration Space for Findpath with Rotation, Proceedings of the 8th International Conference on Artificial Intelligence, 799-806.

Brooks Jr, F.P. (1996) The Computer Scientist as Toolsmith II, Communications of the ACM, March, Vol. 39, No. 3, 61-68.

Canny J.F. (1988) The Complexity of Robot Motion Planning, Cambridge, MIT Press.

Canny J.F., Donald B.R. (1988) Simplified Voronoi Diagrams, Discrete and Computational Geometry, Vol. 3, 219-236.

Caselli S., Reggiani M., Sbravati R. (2002) Parallel Path Planning with Multiple Evasion Strategies, Proceedings of the 2002 IEEE International Conference on Robotics and Automation, IEEE, 260-266.

Chang H., Li T.-Y. (1995) Assembly Maintainability Study with Motion Planning, Proceedings of the 1995 IEEE International Conference on Robotics and Automation, IEEE, 1012-1019.

Chen P. C. (1992) Improving Path Planning with Learning, Machine Learning: Proceedings of the 9th International Workshop, Morgan Kaufmann Publishers, San Mateo, 55-61.

Chen P.C., Hwang Y.K. (1992) SANDROS: A Motion Planner with Performance Proportional to Task Difficulty, Proceedings of the 1992 IEEE International Conference on Robotics and Automation, IEEE, 2346-2353.

Chen P.C., Hwang Y.K. (1998) SANDROS: A Dynamic Graph Search Algorithm for Motion Planning, IEEE Transactions on Robotics and Automation, Vol. 14, No. 3, June, 390-403.

Cohen P. R. (1995) Empirical Methods for Artificial Intelligence, The MIT Press, Cambridge, 1995.

Connolly C.I., Burns J.B., Weiss R. (1990) Path Planning Using Laplace's Equation, Proceedings of the 1990 IEEE International Conference on Robotics and Automation, IEEE, 2102-2106.

Cook S.A. (1983) An Overview of Computational Complexity, Communications of the ACM, Vol. 26, No. 6, June, 400-408.

Cortés J., Siméon T., Laumond J.P. (2002) A Random Loop Generator for Planning the Motions of Closed Kinematic Chains Using PRM Methods, Proceedings of the 2002 IEEE International Conference on Robotics and Automation, IEEE, 2141-2146.

Crowder H., Demo R.S., Mulvey J.H. (1978) Reporting Computational Experiments in Mathematical Programming, Mathematical Programming, Volume 15, 316-329.

Delmia (2003a) IGRIP, http://www.delmia.com/gallery/pdf/igrip.pdf, 19.5.2003

Delmia (2003b) ENVISION/ASSEMBLY, http://www.delmia.com/gallery/pdf/envision_assembly.pdf, 19.5.2003

Deneb (1994) TELEGRIP User Manual, Deneb Robotics Inc.

Denning P.J. (1980) What is Experimental Computer Science, Communications of the ACM, Vol. 23, No. 10, October, 543-544.

Denning P.J. (1981) Performance Analysis: Experimental Computer Science at Its Best, Communications of the ACM, Volume 24, Number 11, November, 725-727.

Denning P. J., Comer D. E., Gries D., Mulder M. C., Tucker A. B., Turner A. J., Young P.R. (1989) Computing as a Discipline, Communications of the ACM, Vol. 32, No. 1, January, 9-23.

Donald B.R. (1987) A Search Algorithm for Motion Planning with Six Degrees of Freedom, Artificial Intelligence, Vol. 31, No. 3, March, 295-353.

Dunnett C.W. (1980) Pairwise Multiple Comparisons in the Unequal Variance Case, Journal of American Statistical Association, Vol. 75 No. 372, 796-800.

Einot I., Gabriel K. R. (1975) A Study of the Powers of Several Methods of Multiple Comparisons, Journal of the American Statistical Association, Vol. 70, No. 351, September, 574-583.

Faverjon B., Tournassoud P. (1987) A Local Based Approach for Path Planning of Manipulators with a High Number of Degrees of Freedom. Proceedings of the 1987 IEEE International Conference on Robotics and Automation, IEEE, 1152-1159.

Gallos J.V. (1996) On Becoming a Scholar: One Woman's Journey, In Frost P.J. and Taylor M.S.(eds.), Rhythms of Academic Life: Personal Accounts of Careers in Academia, SAGE Publications, Thousand Oaks, California, USA, 11-18.

Geraerts R., Overmars M. H. (2002) A Comparative Study of Probabilistic Roadmap Planners, Technical Report 2002-041, Department of Computer Science, Utrecht University.

Glaser R. E. (1983) Levene's Robust Test of Homogeneity of Variances, In Kotz & Johnson (eds.), Encyclopedia of Statistical Sciences, Vol 4, John Wiley & Sons, 608-610.

Glavina B. (1990) Solving Findpath by Combination of Goal-Directed and Randomized Search, Proceedings of the 1990 IEEE International Conference on Robotics and Automation, IEEE, 1176-1181.

Glavina B. (1991) A Fast Motion Planner for 6-DOF Manipulators in 3-D Environments, Proceedings of the 1991 International Conference on Advanced Robotics, 1176-1181.

Goldberg A.V. (1999), Selecting Problems for Algorithm Evaluation, In J.S. Vitter, C.D. Zaroliagis (eds.), Workshop on Algorithm Engineering 1999, Lecture Notes on Computer Science 1668, 1-11.

Gottschalk S., Lin M. C., Manocha D. (1996) OBB-Tree: A Hierarchical Structure for Rapid Interference Detection, Technical report TR96-013, Department of Computer Science, University of N. Carolina, Chapel Hill.

Graux L., Millies P., Kociemba P.L., Langlois B. (1992) Integration of a Path Generation Algorithm into Off-line Programming of AIRBUS Panels, Proceedings of Aerofast'92 (Bellevue, Washington), SAE Technical Papers Series, October.

Gropp W., Lusk E., Doss N., Skjellum A. (1996) A High-Performance, Portable Implementation of the MPI Message Passing Interface Standard, Parallel Computing, Vol. 22, No. 6, September, 789-828.

Gupta K.K. (1998.), Motion Planning for "Flexible" Shapes (Systems with Many Degrees of Freedom): A Survey, The Visual Computer, Vol. 14, No. 5/6, 288-302.

Gupta K.K. (1998a) Overview and State of Art, In (Gupta and Pobil 1998, p. 3-8).

Gupta K.K. (1998b) The Sequential Framework for Practical Motion Planning for Manipulator Arms: Algorithm and Experiments, In (Gupta and Pobil 1998, p. 7-31).

Gupta K., del Pobil A. P. (eds.) (1998) Practical Motion Planning in Robotics: Current Approaches and Future Directions, John Wiley & Sons, West Sussex.

Gupta K., Zhu X. (1994) A Novel Approach to Motion Planning for Many Degrees of Freedom Manipulators: Numerical Potential Fields within Sequential Framework, Proceedings of the 1994 IEEE International Conference on Robotics and Automation, IEEE, 2038-2043.

Haaparanta L. and Niiniluoto I. (1986) Johdatus tieteelliseen ajatteluun, Reports from the Department of Philosophy, University of Helsinki, No 3, 1986, Hakapaino Oy, Helsinki, 1995. (In Finnish)

Han L., Amato N.M. (2001) A Kinematics-Based Probabilistic Roadmap Method for Closed Chain Systems, Workshop on Algorithmic Foundations of Robotics, A K Peters, Wellesley, 2001, 233-246.

Hart P., Nilsson N., Raphael B. (1968) A Formal Basis for the Heuristic Determination of Minimum Cost Paths, IEEE Transactions on Systems Science and Cybernetics, Vol. 4, No. 2, July, 100-107.

Henrich, D., Wurll, C., Worn, H. (1998) 6 DOF Path Planning in Dynamic Environments - A Parallel On-line Approach, Proceedings of the 1998 IEEE International Robotics and Automation, IEEE, 330-335.

Holleman C., Kavraki L. (2000) A Framework for Using the Workspace Medial Axis in PRM Planners, Proceedings of the 2000 IEEE International Conference on Robotics and Automation, IEEE, 1408-1413.

Hooker J.N. (1994), Needed: An Empirical Science of Algorithms, Operations Research, Vol. 42, No. 2, March-April, 201-212.

Hooker J.N. (1995), Testing Heuristics: We Have It All Wrong, Journal of Heuristics, Vol. 1, No. 1, Fall, 33-42.

Hopcroft J., Schwartz J.T., Sharir M. (1984) On the Complexity of Motion Planning for Multiple Independent Objects; PSPACE-Hardness of the "Warehouseman's Problem", International Journal of Robotics Research, Vol. 3, No. 4, 76-88.

Horsch Th., Schwarz F., Tolle H. (1994) Motion Planning with Many Degrees of Freedom - Random Reflections at C-Space Obstacles, Proceedings of the 1994 IEEE International Conference on Robotics and Automation, IEEE, 3318-3323.

Hsu D., Latombe J. C., Motwani R. (1997) Path Planning in Expansive Configuration Spaces, Proceedings of the 1997 IEEE International Conference on Robotics and Automation, IEEE, 2719-2726.

Hsu D., Kavraki L. E., Latombe J.-C., Motwani R., Sorkin S. (1999) On Finding Narrow Passages with Probabilistic Roadmap Planners, Workshop on Algorithmic Foundations of Robotics, A.K. Peters, Wellesley, 141-154.

Hsu D., Kavraki L. E., Latombe J.-C., Motwani R.(1999a) Capturing the Connectivity of High-Dimensional Geometric Spaces by Parallelizable Random Sampling Techniques, in P. M. Pardalos and S. Rajasekaran (eds), Advances in Randomized Parallel Computing, Combinatorial Optimization Series, Kluwer Academic Publishers, Boston, 159-182.

Hwang Y.K. (1996) Completeness vs. Efficiency in Real Applications of Motion Planning, Proceedings of the Workshop on Practical Motion Planning in Robotics: Current Approaches and Future Directions, IEEE.

Hwang Y. K., Ahuja N. (1992) Gross Motion Planning - A Survey, ACM Computing Surveys, Vol. 24, No. 3, September, 219-291.

Hwang Y.K., Chen P. C. (1995) A Heuristic and Complete Planner for the Classical Mover's Problem, Proceedings of the 1995 IEEE International Conference on Robotics and Automation, IEEE, 729-736.

Isto P. (1996) Toward Efficient Search Algorithm for Motion Planning, Proceeding of the 15th Workshop of the UK Planning and Scheduling Special Interest Group, Liverpool John Moores University, Liverpool, Vol.2, 182-183.

Jackson R.H.F., Boggs P.T., Nash S.G., Powell S. (1991) Guidelines for Reporting Results of Computational Experiments, Report of the Ad Hoc Committee, Mathematical Programming, Vol. 49, 413-425.

Joseph D.A., Plantinga W.H. (1985) On the Complexity of Reachability and Motion Planning Questions, Proceedings of the First ACM Symposium on Computational Geometry, ACM, 62-66.

Järvenpää E. and Kosonen K. (1997) Johdatus tutkimusmenetelmiin ja tutkimuksen tekemiseen, Teaching Material No. 1, Department Industrial Management, Helsinki University of Technology, Otamedia Oy, Espoo, 2000. (In Finnish)

Järvinen P. (1999), On Research Methods, Tampereen Yliopistopaino Oy, Juvenes-Print, Tampere.

Kavraki L., Latombe J.-C. (1994) Randomized Preprocessing of Configuration Space for Fast Path Planning, Proceeding of the 1994 IEEE International Conference on Robotics and Automation, IEEE, 2138-2145.

Kavraki L., Latombe J.-C., Motwani R., Raghavan P. (1995) Randomized Query Processing in Robot Motion Planning, Proceeding of the ACM SIGACT Symposium on Theory of Computing (STOC), ACM, Las Vegas, 353-362.

Kavraki L.E., Švestka P., Latombe J.C., Overmars M (1996), Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces, IEEE Transactions on Robotics and Automation, Vol. 12, No. 4, August, 566-580.

Khatib O. (1985) Real-Time Obstacle Avoidance for Manipulators and Mobile Robots, Proceedings of the 1985 IEEE International Conference on Robotics and Automation, IEEE, 500-505.

Kineo (2001) Move3D – Software Development Kit for Motion Synthesis, Technical Presentation, Kineo Computer Aided Motion.

Kineo (2003) http://www.kineocam.com/kineoframes.php?choix=products, 30.5.2003.

Koditschek D. E. (1987) Exact Robot Navigation by Means of Potential Functions: Some Topological Considerations, Proceedings of the 1987 IEEE International Conference on Robotics and Automation, IEEE, 1-6.

Koga Y., Kondo K, Kuffner J., Latombe J.-C. (1994) Planning Motions with Intentions, Proceedings of SIGGRAPH 94, ACM SIGGRAPH, 395-408.

Kondo K. (1991a) Collision Avoidance by Free Space Enumeration Based on Heuristic Graph Search, Advanced Robotics, Vol. 5, No. 3, 293-308.

Kondo K. (1991b) Motion Planning with Six Degrees of Freedom by Multistrategic Bidirectional Heuristic Free-Space Enumeration. IEEE Transactions on Robotics and Automation, Vol. 7, No. 3, June, 267-277.

Kondo K., Kimura F. (1989) Collision Avoidance Using a Free Space Enumeration Method Based on Grid Expansion, Advanced Robotics, Vol. 3, No. 3, 159-175.

Kuffner jr J. (2002) Some Computed Examples, http://www.kuffner.org/james/plan/examples.html, 19.5.2003

Ladd A., Kavraki L.E. (2002) Generalizing the Analysis of PRM, Proceedings of the 2002 IEEE International Conference on Robotics and Automation, IEEE, 2120-2125.

Lamiraux F., Kavraki L. E. (2001) Planning Paths for Elastic Objects under Manipulation Constraints, International Journal of Robotics Research, Vol. 20, No. 3, March, 188-208.

Laumond J.P., Sekhavat S., Lamiraux F. (1998) Guidelines in Nonholonomic Motion Planning for Mobile Robots, In Laumond J.P. (ed.), Robot Motion Planning and Control, Lectures Notes in Control and Information Sciences 229, Springer Verlag, 1-53.

Laumond J.-P., Siméon T. (2001) Notes on Visibility Roadmaps and Path Planning, Workshop on Algorithmic Foundations of Robotics, A K Peters, Wellesley, 317-328.

Latombe J.-C. (1991), Robot Motion Planning, Kluwer Academic Publishers, Boston / Dordrecht / London.

Latombe J.-C. (1999), Motion Planning: A Journey of Robots, Molecules, Digital Actors, and Other Artifacts, International Journal of Robotics Research, Vol. 18, No. 11, November, 1119-1128.

LaValle S.M., Kuffner J.J (1999) Randomized Kinodynamic Planning, Proceedings of the 1999 IEEE International Conference on Robotics and Automation, IEEE, 473-479.

LaValle S.M., Kuffner J.J (2001) Rapidly-Exploring Random Trees: Progress and Prospects, Workshop on Algorithmic Foundations of Robotics, A K Peters, Wellesley, 2001, 293-308.

LaValle S.M., Yakey J.H., Kavraki L.E (1999) A Probabilistic Roadmap Approach for Systems with Closed Kinematic Chains, Proceedings of the 1999 IEEE International Conference on Robotics and Automation, IEEE, 1671-1676.

LienJ.-M., Thomas S.L., Amato N.M. (2003) A General Framework for Sampling on the Medial Axis of the Free Space, To appear in the proceedings of the 2003 IEEE International Conference on Robotics and Automation, IEEE.

Lindemann S.R., LaValle S.M. (2003a) Incremental Low-Discrepancy Lattice Methods for Motion Planning, To appear in the proceedings of the 2003 IEEE International Conference on Robotics and Automation, IEEE.

Lindemann S.R., LaValle S.M. (2003b) Steps Toward Derandomizing RRTs, To appear in the proceedings of the 2003 IEEE International Conference on Robotics and Automation, IEEE.

Lozano-Pérez T. (1981) Automatic Planning of Manipulator Transfer Motions, IEEE Transactions on Systems, Man and Cybernetics, Vol. 11, No. 10, October, 681-698.

Lozano-Pérez T. (1987) A Simple Motion Planning Algorithm for General Robot Manipulators, IEEE Journal of Robotics and Automation, Vol RA-3, No. 3, June, 224-238.

Lozano-Pérez T., Jones J.L., Mazer E., O'Donnell P.A. (1992) HANDEY: A Robot Task Planner, MIT Press, Cambridge, Mass.

Lozano-Pérez T., Jones J.L., Mazer E., O'Donnell P.A., Grimson E.L., Tournasoud P., Lanusse A. (1987) Handey: A Robot System that Recognizes, Plans, and Manipulates, Proceedings of the 1987 IEEE International Conference on Robotics and Automation, IEEE, 843-849.

Lozano-Pérez T., Wesley M. (1979) An Algorithm for Planning Collision-free Paths among Polyhedral Obstacles. Communications of the ACM, Vol. 22, No. 10, October, 560-570.

Loeff L.A., Soni A.H. (1975) An Algorithm for Computer Guidance of a Manipulator in Between Obstacles, Transactions of the ASME, Journal of Engineering for Industry, Vol. 97, Series B, No. 3, August, 836-842.

March S.T., Smith G.F. (1995) Design and Natural Science Research on Information Technology, Decision Support Systems, Vol. 15, 251-266.

Montgomery D. C. (1997) Design and Analysis of Experiments, John Wiley & Sons, New York.

Moret B.M.E. (2002), Towards a Discipline of Experimental Algorithmics, In M.H. Goldwasser, D.S. Johnson, and C.C. McGeoch (eds.), Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges, DIMACS Monographs 59, 197-213, American Mathematical Society, Providence.

Newell, A., Perlis, A.J., Simon, H.A. (1967), Computer Science, Letter to the editor, Science, Vol. 157, No. 3795, September, 1373-1374.

Newell A., Simon H.A. (1976) Computer Science as Empirical Inquiry: Symbols and Search, Communications of the ACM, Vol 19, No 3, March, 113-126.

Nielsen, C., Kavraki, L.E. (2000) A Two-Level Fuzzy PRM for Manipulation Planning. Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE 1716-1722.

Niiniluoto I. (1980) Johdatus tieteenfilosofiaan - käsitteen- ja, teorianmuodostus, Otava, Helsinki. (In

Finnish)

Nilsson N.J. (1969) A Mobile Automaton: An Application of Artificial Intelligence Techniques, Proceedings of the 1st International Joint Conference on Artificial Intelligence, Morgan Kaufman, 509-520.

Overgaard L., Larsen R., Jacobsen N. (1998) Industrial Applications of the AMROSE Motion Planner, In (Gupta and Pobil 1998, p.155-184).

Overmars, M.H. (1992), A Random Approach to Motion Planning, Technical Report RUU-CS-92-32, October, Department of Computer Science, Utrecht University.

Overmars M.H. (2002) Recent Developments in Motion Planning, Technical Report 2002-004, Institute of Information and Computing Sciences, Utrecht University.

Overmars M. H., Švestka P. (1994) A Probabilistic Learning Approach to Motion Planning, Technical Report UU-CS-1994-03, Department of Computer Science, Utrecht University, The Netherlands, January.

Paden B., Mees A., (1989) Path Planning Using a Jacobian-Based Freespace Generation Algorithm, Proceeding of the 1989 IEEE International Conference on Robotics and Automation, IEEE, 1732-1737.

Pirie W. (1983) Jonckheere Tests for Ordered Alternatives, In Kotz & Johnson (eds.), Encyclopedia of Statistical Sciences, Vol. 4, John Wiley & Sons, 315-318.

Pohl, I. (1971) Bi-directional Search, Machine Intelligence 6, Edinburgh University Press, Edinburgh, 127-140.

Qin C., Henrich D. (1996) Randomized Parallel Motion Planning for Robot Manipulators, Technical Report 5/96, Computer Science Department, University of Karlsruhe.

Reif J.H. (1979) Complexity of the Mover's Problem and Generalizations, Proceedings of the 20th IEEE Symposium on Foundations of Computer Science, IEEE, New York, 421-427.

Rich, E., Knight K. (1991) Artificial Intelligence, International Edition, McGraw-Hill, Singapore.

Russell, S. (1992) Efficient Memory-Bounded Search Methods, Proceedings of the 10th European Conference on Artificial Intelligence, John Wiley & Sons, Chichester, 1-5.

Schwartz J.T., Sharir M. (1983) On the `Piano Movers' Problem: II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds, Advances in Applied Mathematics, Vol. 4, 298-351.

Schwartz J.T., Sharir M. (1990) Algorithmic Motion Planning in Robotics, In van Leeuwen J. (ed.), Handbook of Theoretical Computer Science, Vol. A: Algorithms and Complexity, Elsevier Science Publishers B.V., Amsterdam, 391-430.

Schwartz J.T., Sharir M., Hopcroft J. (1987) Planning, Geometry and Complexity of Robot Motion, Ablex, Norwood.

Sent D., Overmars M. (2001) Motion Planning in Environments with Danger Zones, Proceedings of the 2001 IEEE International Conference on Robotics and Automation, IEEE, 1488-1493.

Sharir M. (1997) Algorithmic Motion Planning, In Goodman J.E., O'Rourke J. (eds), Handbook of Discrete and Computational Geometry, CRC Press, Boca Raton, Florida, 733-754.

Siegel S. (1956) Nonparametric Statistics, McGraw-Hill, Tokyo.

Siméon T. (2001) Personal communication, 7.9.2001.

Siméon T, Cortés J., Sahbani A., Laumond J.-P. (2002) A Manipulation Planner for Pick and Place Operations under Continuous Grasps and Placements, Proceedings of the 2002 IEEE International Conference on Robotics and Automation, IEEE, 2022-2027.

Siméon T, Laumond J.-P., Nissoux C. (2000) Visibility-based Probabilistic Roadmaps for Motion

Planning, Advanced Robotics, Vol. 14, No. 6, 477-494.

Siméon T, Laumond J.-P., Van Geem C., Cortes J. (2001) Computer Aided Motion: Move3D within MOLOG, Proceedings of the 2001 IEEE International Conference on Robotics and Automation, IEEE, 1494-1499.

Singh A.P., Latombe J.C., Brutlag D.J. (1999) A Motion Planning Approach to Flexible Ligand Binding, Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology (ISMB), AAAI Press, Menlo Park, CA, 252-261.

Song G., Amato N. M. (2000) A Motion Planning Approach to Folding: From Paper Craft to Protein Structure Prediction, Technical Report 00-001, Department of Computer Science, Texas A&M University, January 17.

Souccar K., Coelho, Jr. J.A., Connolly C.I., Grupen R.A. (1998) Harmonic Functions for Path Planning and Control, In (Gupta and Pobil 1998, p. 277-301).

Tannenbaum A., Yomdin Y. (1987) Robotic Manipulators and the Geometry of Real Semialgebraic Sets, IEEE Journal on Robotics and Automation, RA-3, 4 August, 301-307.

Deneb (1994) TELEGRIP User Manual. Deneb Robotics Inc.

Tuominen J., Autere A., Berger U., Meier I. R. (1995) Autonomous Robot Based Disassembly of Automotive Components, Proceedings of the Conference on Integration in Manufacturing, IOS Press, Amsterdam, 341-352.

Vallejo D., Jones C., Amato N.M. (2000) An Adaptive Framework for `Single Shot' Motion Planning, Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 1722-1727.

Vallejo D., Remmler I., Amato N. M (2001) An Adaptive Framework for `Single Shot' Motion Planning: A Self-Tuning System for Rigid and Articulated Robots, Proceedings of the 2001 IEEE International Conference on Robotics and Automation, IEEE, 21-26.

Watterberg P., Xavier P., Hwang Y. (1997) Path Planning for Everyday Robotics with SANDROS, Proceedings of the 1997 IEEE International Conference on Robotics and Automation, IEEE, 1170-1175.

Wilmarth S.A., Amato N.M., Stiller P.F. (1999a) MAPRM: A Probabilistic Roadmap Planner with Sampling on the Medial Axis of the Free Space, Proceedings of the 1999 International Conference on Robotics and Automation, IEEE, 1999, 1024-1031.

Wilmarth S.A., Amato N.M., Stiller P.F. (1999b) Motion Planning for a Rigid Body Using Random Networks on the Medial Axis of the Free Space, Proceedings of the ACM Symposium on Computational Geometry (SoCG), 173–180.

Zhu X., Gupta K. (1993) On Local Minima and Random Search in Robot Motion Planning, Unpublished manuscript.

APPENDIX: PAPERS I-VI