SOLVING MOTION PLANNING PROBLEMS
BY
ITERATIVE RELAXATION OF CONSTRAINTS

A Dissertation

by

OSMAN BURÇHAN BAYAZIT

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

DOCTOR OF PHILOSOPHY

May 2003

Major Subject: Computer Science

SOLVING MOTION PLANNING PROBLEMS
BY
ITERATIVE RELAXATION OF CONSTRAINTS

A Dissertation

by

OSMAN BURÇHAN BAYAZıT

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

DOCTOR OF PHILOSOPHY

Approved as to style and content by:

Nancy M. Amato (Chair of Committee)	Ergun Akleman (Member)
Donald H. House (Member)	Bruce H. McCormick (Member)
Richard A. Volz (Member)	Valerie E. Taylor (Head of Department)

May 2003

Major Subject: Computer Science

ABSTRACT

Solving Motion Planning Problems by Iterative Relaxation of Constraints.
(May 2003)

Osman Burçhan Bayazıt, B.S., Middle East Technical University;
M.S., Texas A&M University
Chair of Advisory Committee: Dr. Nancy M. Amato

This dissertation presents a new approach to improve automated motion planners. Automatic motion planning has application in many areas such as robotics, virtual reality systems, and computer-aided design. Surprisingly, a single class of planners, called probabilistic roadmap methods (PRMs), have proven effective on problems from all these domains. Strengths of PRMs are simplicity and efficiency, even in high-dimensional configuration spaces. Nevertheless, PRMs are not as effective in environments where the solution path requires the robot to pass through a narrow passage.

In this dissertation, we suggest a hierarchical strategy addressing this problem where we first simplify the problem by relaxing some feasibility constraints, solve the easier version of the problem, and then use that solution to help find a solution for the harder problem. We show how this strategy can be applied to (i) “virtual prototype” analysis, where the goal is to find a removal path for one part (the robot) from an assembly of other parts (obstacles), (ii) “ligand binding”, where we generate candidate binding sites for a ligand in a large protein molecule, and (iii) “deformable objects”, where the robot can deform itself to avoid collision while following a path.

In the living memory of my Father,
and to my Mom ...

...
Sitem etmek için çok erken
Sevda icre sevdalı olmak dururken
Biraz acımsı bakmayı artık bırak
Bu çılgın gençliğimiz elimizdeyken

MD. Sadi Bayazıt
Neşter Magazine, April 1961

ACKNOWLEDGMENTS

I first met my advisor, Dr. Amato in 'Motion Planning' class I took during my very first year at Texas A&M University. During the geometric modeling class we had together I had always thought that she was a student just like I was. She was young, bright and energetic. Soon after it was with great surprise that I discovered she actually was a very successful academician despite how young she was. It was then that I made up my mind to become one of her graduate students. My seven years at Texas A&M University with her proved how right that decision was.

I believe a lot of professors can be very good academicians but there are only a few who can be both very good academicians and very good advisors. To me and all her other students, whom I know, Dr. Amato is a very special person who is one of those. As an academician she is one who always seeks perfection and as an advisor motivates to achieve the best. Dr. Amato herself forms an excellent model showing how the combination of intelligence, drive, dedication and perfect communication skills can transform into the achievement. For all she has done and dedication she has displayed for my work, I cannot thank enough to Dr. Amato.

I would like to take this opportunity to thank my committee members as well. I am grateful to Dr. House for his lecture on physically based modeling which is one of the top courses I took at Texas A&M University. My special thanks also goes to Dr. Volz for his understanding while I was working as a system administrator, responsible from the SGI computers in his laboratory. Together with Dr. House, he helped me with my job search and I'm forever grateful to them both for their warm support.

Dr. McCormick was the instructor of one of the most excellent courses I have ever taken- Neural Networks- and I have always admired his intelligence. It was with

regret that I could not take part in his 'Brain Scanner Project' which came a bit late.

Here I would also like to thank Dr. Wu, my GCR, for taking the time for my research and for helping me to schedule of my defense.

During my years at Texas A&M, I have been given the opportunity to work with several excellent students and researchers. Here, I would like to mention the members of my research group: Ping An, Lucia Dale, Erin Devoy, Li Han, Christopher Jones, Kasthuri Srinivasan Kannan, Jinsuck Kim, Kyunghwan Kim, Derek Kurth, Sooyong Lee, Jyh-Ming Lien, Marco Morales, Roger Pearce, Sam Rodrigez, Bharatinder Sandhu, Whookho Son, Guang Song, Sujay Sundaram, Rick Stover, Xinyu Tang, Paul Thomas, Shawna Thomas, Daniel Vallejo, Aunee Vargas, Jennifer Walter, Steve Wilmarth, and Dawen Xie. I would like to thank them all for sharing their work with me and for fruitful hours we spent together to discuss our research. I feel very much in debt to Guang and Jyh-Ming for helping my research. I have always admired both of them and it is my privilege to be their 'friend'. Marco, Xinyu, Jack and Tim Smith, my colleagues from the system administration post, have always been understanding and helpful during my busy academic research. Jack also helped me to get to know the best science fiction series, Babylon-5, and I can not thank him enough for that.

My special thanks goes to Turkish Ministry of Education for providing the funds for my study. It is through that I have reached new horizons in my area.

My years in College Station, far from home, have been enriched by very special friends who always supported me. I would like to thank Taner R. Ozdil for his warm support, and true friendship. He has always been there for me in times of hardship and for me he is no less than a brother. Huseyin Berkmen has been one of the warmest people I met here. I'm grateful to him for his advice and support. Ergun Akleman has been my mentor since the day I arrived College Station. He, as a very special friend played an important role as I grew in the academic area. His family Derya and

Asli Akleman as well as himself will always be part of my family in College Station.

I am grateful to the love of my life, Canan Tunca, for her true love and faith in me. My greatest achievement is not finishing my degree at Texas A&M University but rather finding her here. Words are not enough to express how special I feel for her and her endless support.

Most of all, I will always be thankful to my wonderful family. My parents for their amazing support and effort to help us be what we are today. My brother, Alphan, has been the joy and the success of our family, many thanks to him for his faith in me. My sister, Gizem, has been my biggest support in my work. She has been not only dedicated to help me with my dissertation using all her intelligence, knowledge and background in computer science, but also gave invaluable advice and support through out my life.

My only regret in my dissertation is that I was not fast enough to finish my dissertation before my father reached the eternal peace. I just hope that Dad is proud of my achievement as much as I have been proud of his all through my life.

TABLE OF CONTENTS

Page

ABSTRACT iii
ACKNOWLEDGMENTS v
TABLE OF CONTENTS viii
LIST OF TABLES xi
LIST OF FIGURES xii

CHAPTER

I INTRODUCTION 1
  A. Motivation 4
  B. Contribution 4
  C. Outline 6

II PROBABILISTIC ROADMAP METHODS 7
  A. Configuration Space (C-space) 7
  B. Probabilistic Roadmap Methods 12
    1. Basic Probabilistic Roadmap Method 12
    2. Obstacle Based Probabilistic Roadmap Method (OBPRM) 14
  C. Feasibility Constraints 14

III ITERATIVE RELAXATION OF CONSTRAINTS 17
  A. Related Work 19
  B. Algorithm 20
  C. Feasibility Constraints and Relaxation 22
  D. Improving the Approximate Solution 24
  E. Choosing Initial Constraint Values 25

IV VIRTUAL PROTOTYPING 27
  A. Related Work 30

CHAPTER Page

- B. Overview 32
  - 1. Achieving Useful Force Feedback 33
  - 2. Variable Reactive Forces 36
  - 3. Roadmap Visualization 36
- C. Approximate Solution Generation 38
  - 1. Automated Path Generation 38
  - 2. Manual Path Generation 39
- D. Improving the Approximate Solution 40
  - 1. Pushing Methods 40
- E. Experiments 43
  - 1. Environments 44
  - 2. Overall Evaluation 45
  - 3. Evaluation of Pushing Methods 47
  - 4. Evaluation of Heuristic Collision Detection 48
- F. Conclusions and Future Work 49

V LIGAND BINDING 51

- A. Related Work 54
- B. Preliminaries 56
  - 1. C-spaces of Ligands 56
  - 2. Potential Energy 59
- C. Overview 60
- D. Approximate Solution Generation 62
  - 1. Automatically Generated Conformations Using Geometrical Information 62
  - 2. Automatically Generated Conformations Using Potential Energy Information 64
  - 3. Haptics-based User Generated Conformations 66
- E. Improving the Approximate Solution 67
  - 1. Approximate Gradient Decent 67
  - 2. Node Connection 67
- F. Experiments 69
  - 1. Scoring Binding Sites 70
  - 2. Results 71
    - a. Rigid vs. Flexible Ligand Models 73
    - b. Haptic-based User Input 74
- G. Conclusion 75

CHAPTER Page

VI DEFORMABLE OBJECTS 76

A. Related Work 78
B. Overview 79
C. Approximate Solution Generation 81
  1. Scaled Robots 82
  2. Penetration 84
  3. Query 85
D. Improving the Approximate Solution 86
  1. Bounding-Box Deformation 87
  2. Geometric Deformation 89
E. Experiments 91
F. Conclusion and Future Work 95

VII CONCLUSIONS AND FUTURE WORK 96

REFERENCES 99

VITA 116

LIST OF TABLES

TABLE Page

I Comparison of Different Approaches 45

II Pushing Methods 48

III Collision Detection Statistics 49

IV Statistics for the Different Protein-Ligand Complexes Studied 69

V Results for OBPRM-like Generation Methods for Flexible Ligands . . 71

VI Comparison of Different Node Generation Strategies 72

VII Roadmap and Deformation Statistics 93

VIII Comparison of Roadmap Generation and Deformation Techniques . . 94

LIST OF FIGURES

FIGURE Page

1 Different motion planning applications. 3

2 Configurations of a fixed base robot. 8

3 Workspace vs. C-space 10

4 Workspace obstacle vs. C-space obstacle. 11

5 A PRM roadmap in C-space 13

6 Configuration Space. 18

7 Using an approximate (invalid) path to find a solution to the original problem. 21

8 Flow diagram of IRC 22

9 If the constraints are relaxed too much, it might be hard to transform the approximate solution into a solution to the original problem. 25

10 Penetration in haptic feedback. 29

11 System architecture 32

12 Flow chart of haptic process. 34

13 Collision detection heuristic when last known result was (a) free and (b) a collision. 35

14 Roadmap visualization. 37

15 Relaxation of collision constraint. 39

16 Pushing approximate paths to free space: (a) simple push ($k = 1$), (b) shortest push, (c) workspace assisted push. 41

17 The flange and alpha-puzzle problems. 44

CHAPTER FIGURE Page

18 A protein in wireframe with a space-fill ligand. PDB file: 1LDM 51

19 Using the PHANToM haptic device for in ligand binding. 53

20 Configuration of Napap (MID). 58

21 Correlation between (a) user-felt force and (b) potential energy. . . . 60

22 Automatically generating conformations using geometrical information. 63

23 Automatically generating conformations using potential energy information. 65

24 The (sorted) potential distribution of all roadmap nodes before and after gradient descent for three protein-ligand complexes (given by pdb identifier): (a) 1LDM, (b) 1A5Z and (c) 1STP. 68

25 A brick deformed by a spherical obstacle. 77

26 Roadmaps for different generation methods and an example query. . 83

27 Finding penetration of a configuration in C-Space 85

28 FFD deformation with a 5 × 5 × 5 lattice. 88

29 An example geometric deformation 91

30 Snapshots of experiments 92

CHAPTER I

INTRODUCTION

Motion planning consists of finding a valid sequence of configurations that moves an object from a start position to a goal position. We face motion planning problems everyday. Finding a route to commute between home and work is a motion planning problem where the car is the movable object. Computing the movements required to reach the car is another example of motion planning. Even the simplest task such as reaching into your pocket and picking up the car keys involves complex movements requiring the selection of values for the shoulder, elbow, wrist, and finger joints to grasp the keys. Although we take such abilities for granted, automating the process is very difficult. Nevertheless, automatic motion planning has captured the imagination of numerous authors ever since Aristotle's suggestion that automatons could replace manual labor.

Today, automatic motion planning is no longer restricted to the imaginations of philosophers or writers, but has application in many areas such as robotics, virtual reality systems, computer-aided design and even computational biology. For example, automated motion planners can be used to validate an engine design by verifying that each part is accessible to the mechanics [8, 21, 22]. In this case, the automated motion planner would try to find a removal path for each part from its position in the assembly (Figure 1(a)). Automated motion planners can also find the necessary movements to fold a carton [115] enabling more adaptable packaging machinery which can be used for different kind of products ((Figure 1(b)). The applications of automated motion planners are not restricted to our familiar world, but include

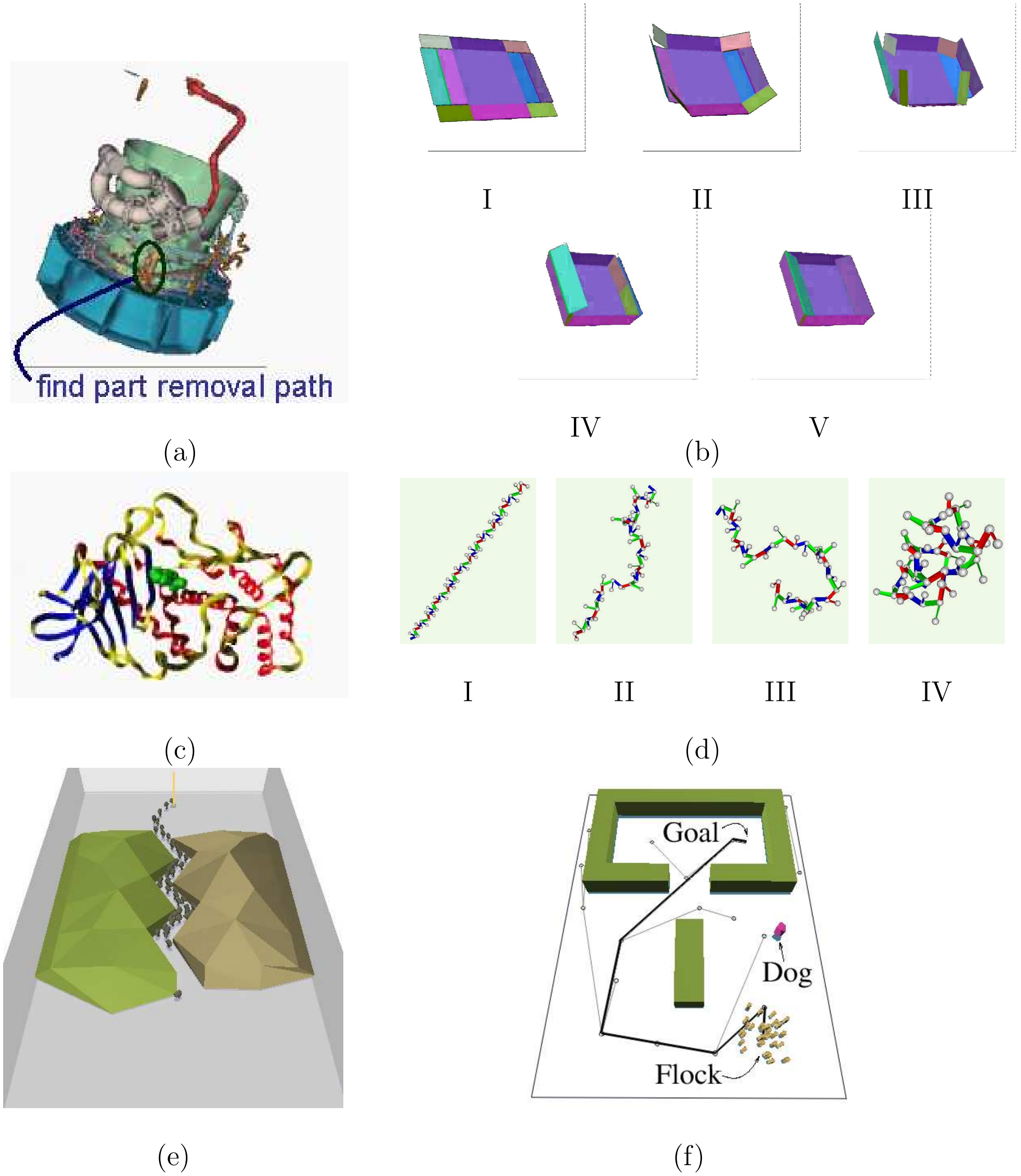
---

The journal model is *IEEE Transactions on Robotics and Automation.*

such seemingly unrelated domains as modeling molecular motions. For example, a drug molecule can be treated like a robot, and an automated planner may evaluate a drug's promise by finding paths for the drug to move to and bind with a protein molecule [23, 24] (Figure 1(c)). Automated motion planners can also be used to simulate a protein molecule folding from a denatured state to its native state [11] (Figure 1(d)). Combined with behavioral rules, automated planners can generate sophisticated group behaviors in artificial life, such as ant-like searching or shepherding of a herd [18, 19, 20] (Figures 1(e) and (f)).

Although many different motion planning methods have been proposed, most are not used in practice since they are computationally infeasible except for some restricted cases, e.g., when the robot has very few degrees of freedom (dof) [60, 77]. Indeed, there is strong evidence that any complete planner (one that is guaranteed to find a solution or determine that none exists) requires time exponential in the number of dof of the robot [106].

For this reason, attention has focused on randomized motion planning methods. In general, these methods can be quite effective when the configuration space (C-space) is relatively uncluttered. Recently, a new class of randomized motion planning methods has gained much attention [69]. These methods, known as *probabilistic roadmap methods* (PRMs), use randomization during preprocessing to construct a graph in C-space (a *roadmap*). They have been successfully applied to several domains: intelligent CAD [8, 21, 22], parts disassembly [122], closed chain linkage systems [52, 79], simulating flocking systems [18, 19, 20], ligand binding [23, 24, 112], paper and/or protein folding [11], etc.. Thus PRMs have been very successful, solving difficult high-dimensional problems that had resisted solution before.



(a) (b) (c) (d) (e) (f)

Fig. 1. Different motion planning applications. (a) In virtual prototyping, parts accessibility is checked by finding paths for each part. (b) In paper folding, the successive set of configurations found will fold the paper. (c) An automated planner can find a path resulting in a drug molecule binding to a drug molecule. (d) Similar to paper folding, an automated planner can find the folding pathway for a protein. (e) When applied to groups, an automated planner can generate ant-like behaviors, or (f) can be used to shepherd a herd of animals.

A. Motivation

One of the most difficult problems faced by PRMs is the so called *narrow passage* problem. In particular, PRMs often fail if the solution path requires the robot to pass through a narrow passage. This is because these methods are based on random sampling strategies and the probability of sampling points in small volume corridors approaches zero as the volume of the corridor decreases. Although some PRM variants have been proposed that address this problem, as we will see in Chapter II, no approach proposed so far consistently solves it.

B. Contribution

In this research, we propose a hierarchical strategy for addressing the narrow passage problem. Briefly, we first relax some feasibility constraints, solve the relaxed version of the problem, and then use that solution to help find a solution for the more constrained problem. For example, given a motion planning query, we first find a path that satisfies the relaxed feasibility constraints (e.g., allowing small penetration) but which may be infeasible for the original problem (which requires a collision-free path). Our strategy is to refine this path to make it feasible for the original problem. There are two benefits to this approach. First, in many cases relaxing the feasibility constraints increases the volume of the free C-space for the original problem, making the relaxed version easier to solve. Second, the solution path for the relaxed problem identifies a smaller region of the C-space to search for a solution to the original problem. We also investigate techniques for incorporating human assistance, which is another way to identify restricted regions of C-space for further processing.

We experiment with several types of robots selected from different domains. In the first two domains the robots have hard surfaces, i.e., the robot is composed of one or more rigid bodies. Our first domain is *virtual prototyping* (Chapter IV) where the robot is a rigid object with 6 dof. Our goal is to find a removal path for one part (the robot) from an assembly of parts (obstacles). Our second domain illustrates the effects of higher dof. In *ligand binding* (Chapter V), the ligand (drug molecule) is represented as a tree-like articulated robot. For this problem, we try to generate candidate binding sites for the ligand in a large protein molecule. In the last domain, *deformable objects* (Chapter VI), we have robots that can change their shape by deforming according to the obstacles in the environment. The initial path, which is allowed to have collision, is found by an automated planner and then the robot deforms to avoid the collisions along the initial path.

These domains help us investigate additional issues in automated motion planning. We investigate techniques for integrating user input and automatic planning methods. In particular, we study methods for enabling human operators to indicate critical regions of the planning space to the automated planner. We develop novel roadmap visualization techniques to show the operator the planning progress. We investigate novel techniques for haptic (sense of touch) sensing of the environment to facilitate the users investigation of the planning space. One of our achievements in this research is being able to get realistic touch feedback between two complex objects. As far as we know, this has not been achieved before.

Preliminary results of our work on virtual prototyping were presented at the *Stanford Workshop on Motion Support in Virtual Prototyping* [7] and at the *Fifth Phantom User's Group Workshop* [8]. The results of our later research were published in the *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* in 2000 and in the *Autonomous Robots Journal's Special Issue on Personal Robotics* in 2001. Our work on ligand binding was published in the *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2001.* It was also presented as a poster at the *Research on Computational Molecular Biology (RECOMB)* conference in 2001 and was published in *Currents in Computer Molecular Biology* which contains short papers from *RECOMB'01* [24]. Finally, our work on deformable objects appears in the *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2002* [17].

C. Outline

In Chapter II we give a brief introduction to configuration space (C-space), PRMs and feasibility criteria and discuss relevant previous work. Chapter III describes our approach to the *narrow passage* problem. Chapters IV, V, and VI describe the application of our approach to virtual prototyping, ligand binding and deformable objects. Chapter VII concludes this dissertation.

CHAPTER II

PROBABILISTIC ROADMAP METHODS

A. Configuration Space (C-space)

The *Configuration space (C-space)* of a robot is the space C of all robot configurations, flexible or not. Each degree of freedom (dof) of the robot corresponds to an axis in C-space and each point in C-space corresponds to a configuration of the robot in the workspace. For a rigid three dimensional robot, three dof are required for translational information ($\mathbb{R}^3$), and three dof are required for rotational information (S^3, where S^1 is a unit circle in the plane). Thus a six-dimensional configuration is sufficient to represent any position and orientation of the robot in a three-dimensional workspace. The six-dimensional C-space of a rigid robot can be represented as

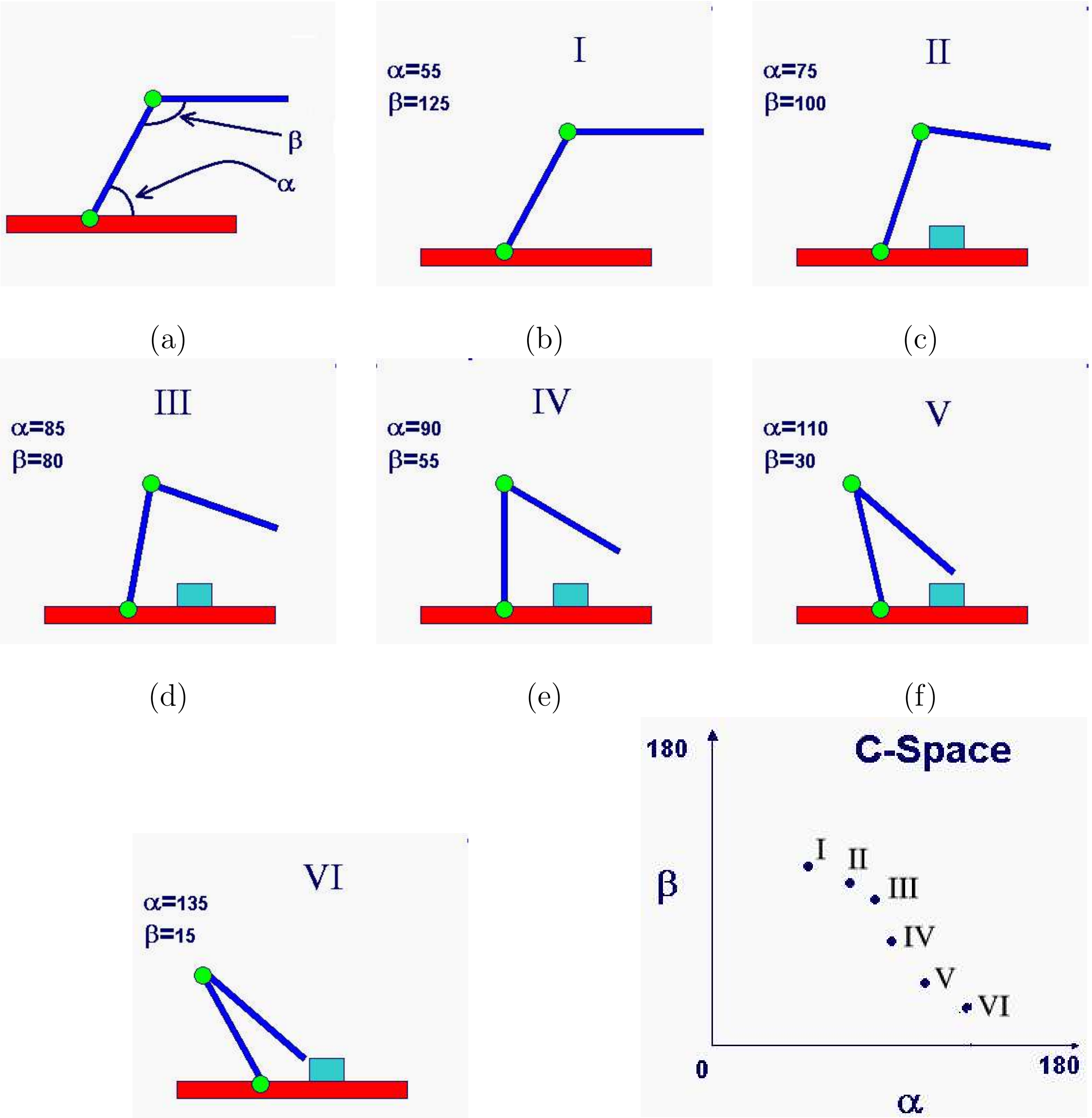
$$\mathcal{C} = \{q \mid q \in \mathbb{R}^3 \times S^3\}. \tag{2.1}$$

If the robot has a fixed base, then the configuration of an n joint articulated robot can be represented by n angles. If the robot is free flying then six additional dof are needed to represent the configuration of the base link. We can represent the C-space of an n link ($n-1$ joint) robot as

$$\mathcal{C} = \{q \mid q \in \mathbb{R}^3 \times S^3 \times S^{n-1}\}. \tag{2.2}$$

Figure 2(a) shows a robot with a fixed base and two links. To represent this robot, one parameter for each angle (α and β) is sufficient. C-space is two dimensional in this case and each configuration of the robot (Figures 2(b)-(g)) is described by a two-dimensional point in the C-space plane (Figure 2(h)).

Similarly, Figure 3 (a) shows a robot that can move on a two dimensional plane



(a) (b) (c)

(d) (e) (f)

(g) (h)

Fig. 2. Configurations of a fixed base robot. (a) A fixed-base, two-link robot requires two parameters. (b)-(g) A successive list of configurations required to reach a package. (h) C-space of this robot with corresponding configurations in it.

with an ability to rotate around its base. Three parameters are required to represent this robot: coordinates $||x||$ and $||y||$ indicate the base's position and an angle θ specifies rotation. Figure 3(b) shows two instances of this robot in the workspace, $c_1 = [x = 4, y = 12, \theta = 45]$ and $c_2 = [x = 15, y = 2, \theta = 220]$. The corresponding C-space points can be seen in Figure 3(c).

In some cases, a configuration of the robot may be invalid, for example a configuration resulting in a collision with an environmental obstacle, a configuration where the joint limits are not honored, or a configuration which has self collisions. Such a configuration is said to be in a *C-space obstacle (C-Obstacle)*. For example, in Figure 4(a), there is an obstacle in the environment. If the robot's angle is 45^o, then any configuration $(x, y, 45)$ where $\{x, y \mid x, y \in$ bordered region in Figure 4(b)$\}$ is an invalid configuration. The full configuration space and the C-obstacle for this environment can be seen in Figure 4(c).

Our representation of C-space in ligand binding is similar to articulated robots. Proteins usually consist of thousands of atoms, while the ligand typically consists of tens. While both protein and ligand are in fact flexible, we model the protein as a rigid body and the ligand as a flexible body (as was done in [112]). In particular, the ligand is modeled as an articulated body with a free base, where the torsional movement of atomic bonds is modeled by one degree of freedom revolute joints (bond angles and lengths are fixed). Thus, the base atom requires 6 dof and each additional torsional movement requires one dof. That is, the C-space for a ligand is described by Equation 5.1 in Chapter V.

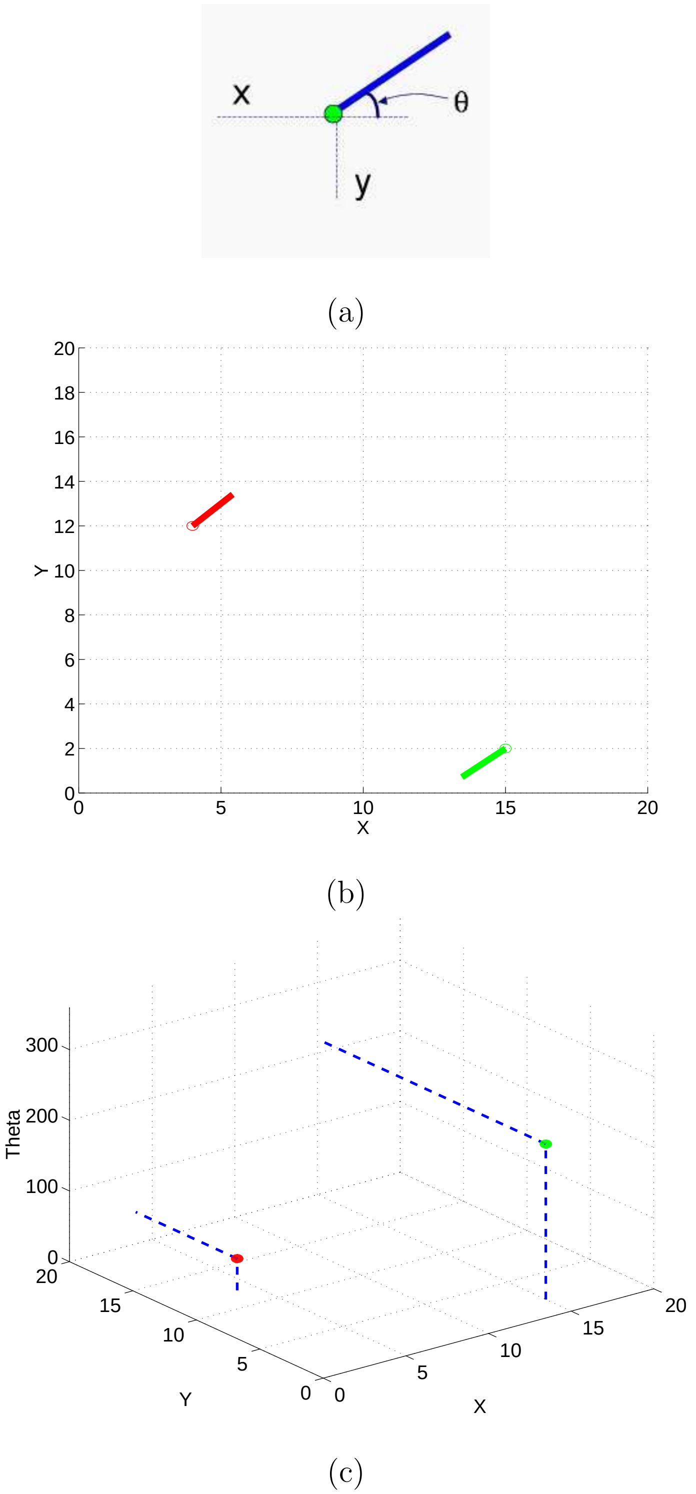
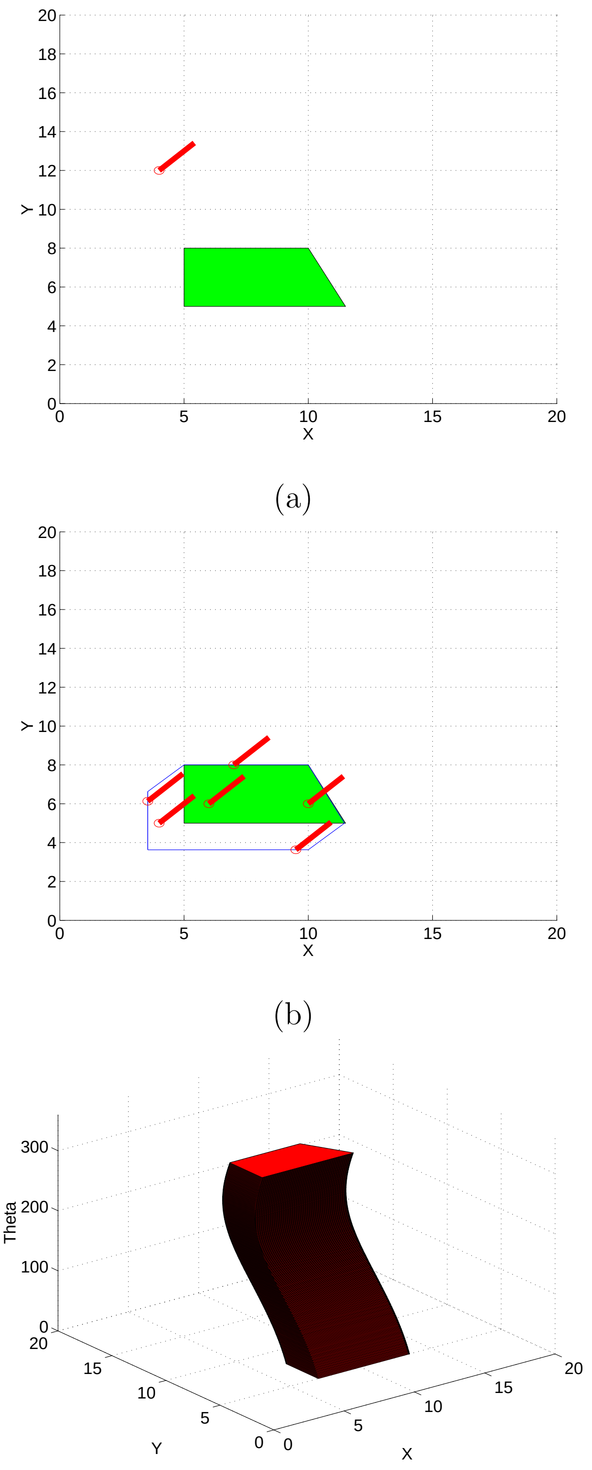


Fig. 3. Workspace vs. C-space: (a) a robot with dof (x, y, θ), (b) two instances of the robot, and (c) corresponding C-space points.



(a)

(b)

(c)

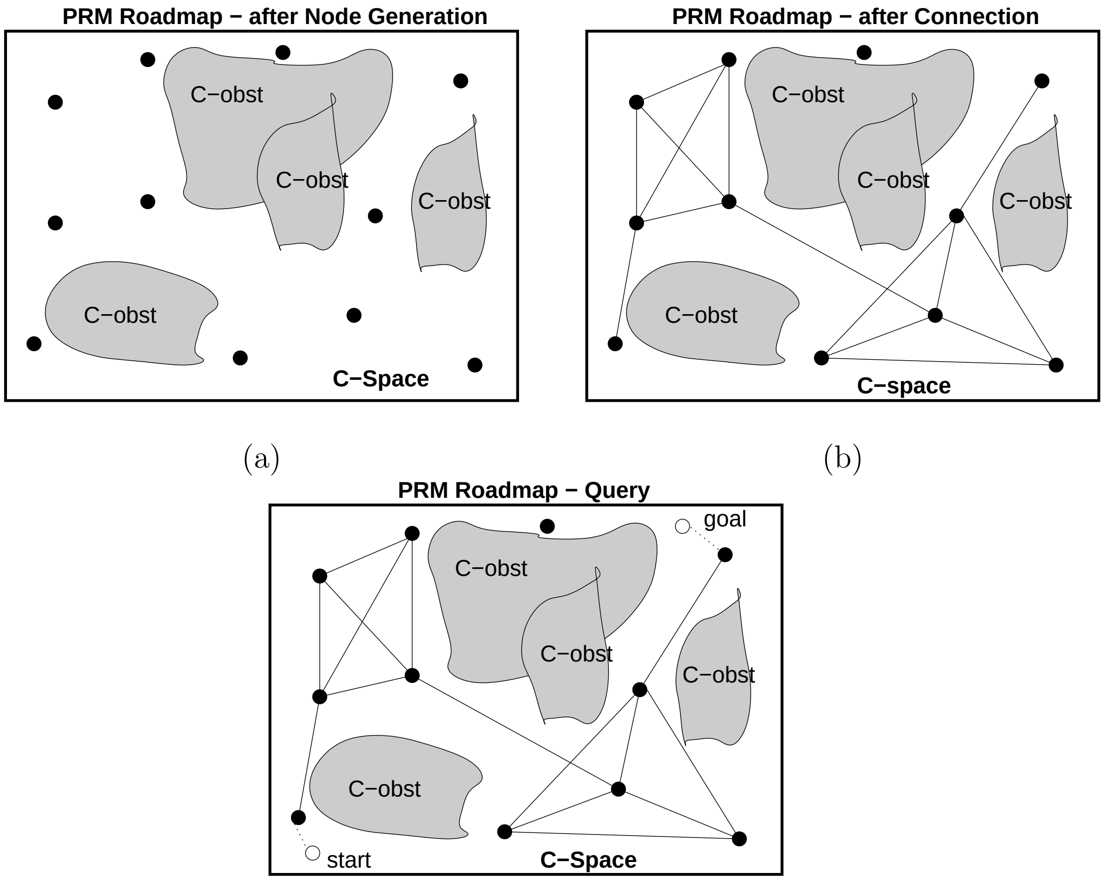
Fig. 4. Workspace obstacle vs. C-space obstacle. (a) A robot with 3 dof $(x, y, \theta = 45)$ and an obstacle. (b) If the robot is inside the border, then that configuration is invalid. (c) C-space obstacle (set of all invalid configurations) for this robot.

B. Probabilistic Roadmap Methods

Given a description of the environment and a movable object (the 'robot'), the motion planning problem is to find a feasible path that takes the movable object from a given start to a given goal configuration [77]. Although many different motion planning methods have been proposed, most are not used in practice since they are computationally infeasible except for some restricted cases, e.g., when the movable object has very few degrees of freedom (dof) [77]. Indeed there is strong evidence that any complete planner (i.e., one that is guaranteed to find a solution, or determine that none exists) requires time exponential in the number of degrees of freedom (dof) of the movable object [77]. For this reason, attention has focused on randomized or probabilistic motion planning methods. In particular, we note the *probabilistic roadmap methods*, or (PRMs), that have recently proven successful on many previously unsolved problems involving high-dimensional C-spaces (see, e.g., [1, 6, 9, 10, 11, 17, 18, 19, 20, 21, 22, 23, 26, 29, 49, 52, 55, 57, 58, 59, 69, 71, 75, 76, 78, 83, 99, 115, 116, 117, 118, 119, 120, 121, 122, 128, 129, 134, 135]).Our approach is based on the this *probabilistic roadmap (*PRM*)* approach to motion planning.

1. Basic Probabilistic Roadmap Method

Briefly, PRMs work by sampling points 'randomly' from C-space, and retaining those that satisfy certain feasibility requirements (e.g., they must correspond to feasible configurations of the movable object, see Figure 5(a), Node Generation). Then, these points are connected to form a graph, or roadmap, using some simple planning method (local planner) to connect 'nearby' points (see Figure 5(b), Connection). During query processing, the start and goal configurations are connected to the roadmap, and a path connecting these connection points is extracted from the roadmap using standard graph search techniques (see Figure 5(c), Query.)



(a) (b)

(c)

Fig. 5. A PRM roadmap in C-space. (a) After node generation, (b) after the connection phase, and (c) using it to solve a query.

The initial PRMs were shown to be very successful in solving difficult problems in high-dimensional configuration spaces that had previously resisted efficient solution [69]. Moreover, they were simple and easy to implement, requiring only collision detection as a primitive operation. These successes sparked a flurry of activity in which PRM motion planning techniques were applied to a number of challenging problems arising in a variety of fields including robotics (e.g., closed-chain systems [52, 79]), CAD (e.g., maintainability [21, 33], deformable objects [12, 68]), and even computational Biology and Chemistry (e.g., ligand docking [112], protein folding [14, 114, 117]). Indeed, it can be argued that the PRM framework was instrumental in this broadening of the range of applicability of motion planning, as many of these problems had never before been considered candidates for automatic methods.

At the same time, these new applications served to point out some weaknesses in the PRM approach. In particular, it was observed that there were some classes of problems for which PRMs did not yield efficient solutions, such as problems where the solution path must traverse *narrow passages* in C-space [57]. As a result, a number of PRM variants specifically targeted at this problem have been proposed, e.g., [6, 27, 57, 134]. Also, some novel approaches for improving PRM efficiency through the use of lazy evaluation techniques have shown promising results [26, 98, 119] (these methods do not address the narrow passage problem).

2. Obstacle Based Probabilistic Roadmap Method (OBPRM)

In several of our applications we will use a particular variant of the PRM called obstacle-based PRM or OBPRM [6]. In OBPRM, instead of generating the nodes uniformly at random in C-space, they are generated on or near constraint surfaces (e.g., contact configurations in traditional motion planning applications). As we will see, OBPRM is particularly well suited for our different domains.

C. Feasibility Constraints

A major strength of PRMs is that they are quite simple to apply, requiring only the ability to randomly generate points in C-space, and then test them for feasibility. As we will see, we have different feasibility criteria for different domains.

Regardless of the PRM stage, feasibility constraints play an important role. At node generation, the sampled configurations must be feasible. At connection, the local planner should check if two configurations are reachable while enforcing the feasibility requirements. Similarly, at the query stage the start and goal configurations must be connected to the roadmap through feasible local paths. Feasibility constraints also can be specified so that among the several solutions, some may be favored among others.

The most commonly used feasibility criterion is that the configuration should be collision-free. The goal of most motion planning problems is to find a collision-free path for a given movable object (robot). Hence, every stage of a PRM should check if the configurations are collision free using a collision detection algorithm, which itself is a much studied problem [31, 32, 39, 63, 64, 66, 70, 72, 88, 93, 136]. In some cases, instead of a collision-free configuration, some penetration of the robot inside the obstacles is allowed. This would have similar effects to shrinking the robot or reducing the volume of the workspace obstacle, both of which would end up with more free space in C-space and result in a higher probability of generating nodes in narrow areas. This replaces the feasibility criterion of being collision free with being within some permissible penetration. Although algorithms exist that calculate penetration depth, they are usually time consuming and some work only in restricted scenarios (e.g., for convex objects [93]). In our problems, it is often sufficient to use an approximation of the penetration depth which can frequently be computed more efficiently than the exact solution. We achieve the approximate penetration: (i) by using a heuristic for collision detection in virtual prototyping (Chapter IV) , (ii) by reducing the size of the robot (scaling) for virtual prototyping and deformable objects (Chapters IV and VI), and (iii) by locally searching C-space to find the closest distance to a nearby free configuration and accepting the penetration if that distance is permissible for deformable objects (Chapter V).

Energy constraints can be used as another feasibility criterion. In ligand binding (Chapter V), the potential energy can be used as a model of the interaction between a ligand and its receptor, a protein. During binding, the ligand can be viewed as moving in the potential field created by the protein's atoms, searching for a stable low energy configuration. Thus, the goal of the planner is to find these stable low energy configurations, and avoid high energy configurations. Usually a complete, accurate model of the potential is not computationally feasible and so the potential is only computed approximately. In our case, we approximate the potential by the van der Waals terms only. While this is an extreme simplification, we note that our goal is not to develop new potential functions but to study the promise of our approach. In a fully developed system, one could use one of the many existing, optimized potential functions (such as AMBER [133], CHARMM [30], CVFF [38], GROMOS [130], MM2/MM3/MM4 [2, 4, 3], MMFF [51], TRIPOS [35] etc.).

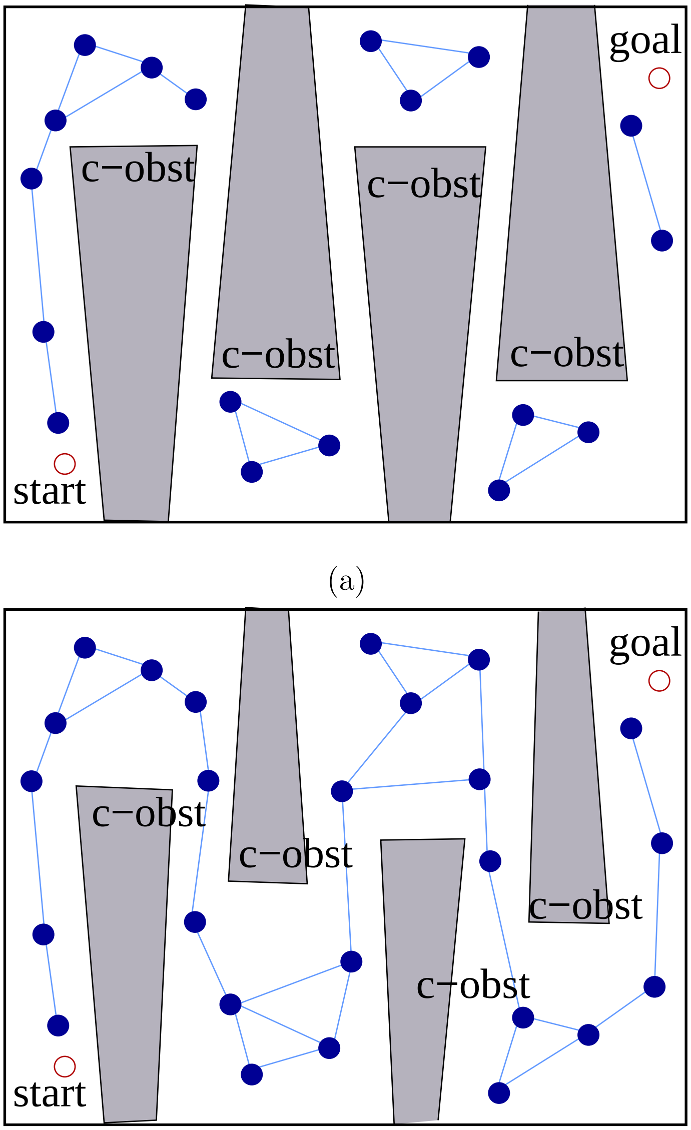
In addition to identifying valid configurations, a parametric representation of feasibility values can be used to weight roadmap edges and assist in selecting paths. For example, clearance values associated with roadmap nodes and edges enable the selection of paths with large clearance. For deformable objects in Chapter VI, we use deformation energy to choose paths tending to reduce deformation.

CHAPTER III

ITERATIVE RELAXATION OF CONSTRAINTS

The main challenge faced by motion planning algorithms is that the configuration space is high-dimensional for any non-trivial problem. Since complete methods for searching such high dimensional spaces are not practical [77, 106], the only feasible methods for searching these spaces are randomized methods such as the probabilistic roadmap methods. Nevertheless, in many cases, the search space is so large, and the (critical) feasible region of this space so small, that the global random sampling approaches of most PRMs are not effective (see Figure 6). In our work, we address this problem with a hierarchical search strategy which iteratively restricts the search to an increasingly refined region of C-space. Our approach is to relax the feasibility criteria (thereby making the problem easier) so that the PRM is able to find a solution to the relaxed version of the problem. Later, the solution of the relaxed version is modified so that it meets the feasibility requirements of the original problem.

The success of our method depends on relaxing the constraints and improving the resulting solutions. Hence our research includes investigation of different feasibility measures and different improvement techniques. In order to evaluate our approach, as well as the measures and techniques, we have selected various domains which enable us to study both issues. In the *virtual prototyping* domain (Chapter IV), we are interested in design for maintainability, i.e., the accessibility of particular parts. We treat the parts to be removed as robots and apply motion planning techniques to determine if the parts can be removed from the environment (other parts) without collision. In our second domain, *ligand binding* (Chapter V), we use a similar technique. In this case, the robot is a flexible drug molecule and the goal is to find a binding site on a given protein molecule (the environment or obstacle). The problem is very high dimensional



(a)

(b)

Fig. 6. Configuration Space. (a) Narrow passages and (b) wider passages

.

with constraints imposed by energy-related, geometrical and biochemical properties. Our last domain, *deformable objects* (Chapter VI) has applications in areas such as computer-aided animation and soft tissue modeling in surgery. Here, we attempt to find a path for a robot which is allowed to change its shape if it collides with any obstacles on the path.

A. Related Work

Although our approach to finding a solution to the relaxed version of the problem and then using that solution to find a solution to the original version of the problem is novel, there has been some related research. Hsu et al. consider a "dilated" space [57]. They initially generate a roadmap in this space by allowing some penetration of the robot. Later, they push those nodes to the free space to generate a collision free roadmap. In contrast, we find a path in dilated space and then push the invalid configurations of the path to valid configurations. We believe our approach is more efficient because it allows targeted exploration of C-space.

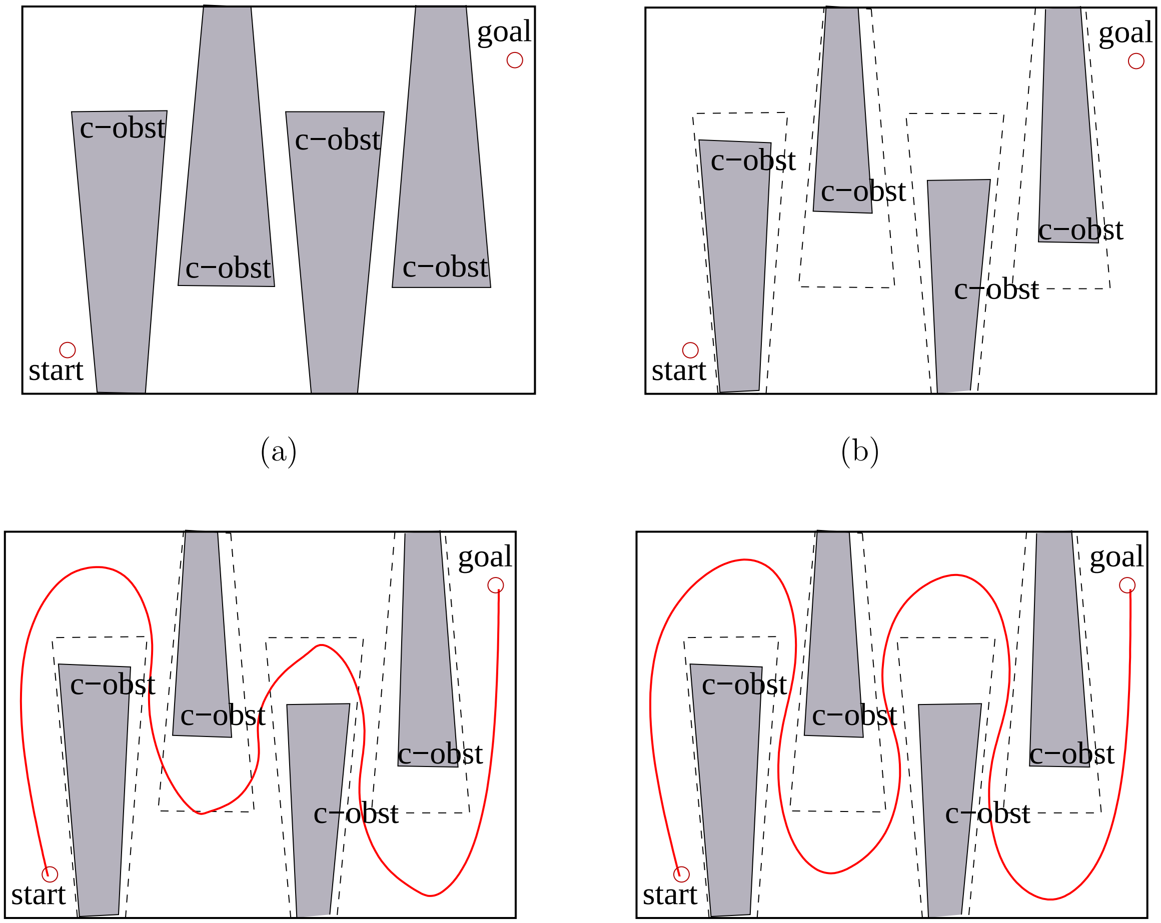
Our philosophy of initially finding an approximate solution is similar to Fuzzy PRM [98], Lazy PRM [47], or Customizable PRM [119] where the roadmap nodes and/or edges are not validated, or are only partially validated, during roadmap construction, and are validated completely only at query time. These methods have been shown to greatly decrease the roadmap construction costs, while only slightly increasing the query costs. Moreover, as noted by Song *et al.* [119], there are actually some other benefits to this approach, such as enabling the same roadmap to be used for different robots, or for queries with different requirements.

B. Algorithm

We believe an approximate solution for a given problem would in many cases help us to solve the original problem. For example, for a given motion planning problem (Figure 7(a)), we may use a reduced scale version of the robot which would reduce the size of the C-space obstacles (Figure 7(b)). If we find a solution in this new space, it may be invalid for the original problem (Figure 7(c)). However, we may focus on the invalid portions of the approximate solution and attempt to correct them, resulting in a solution to the original problem. The potential for performance gains comes because we can restrict our exploration to the regions where the solution is difficult. That is this technique assists us in identifying the easy and difficult parts of the problem and allows us to neglect entire regions of the space in our search (Figure 7(d)).

By defining a hierarchical set of feasibility constraints, this approach can be generalized to an iterative process which starts with the least constrained problem and progressively tightens the constraints until the fully constrained problem is obtained. That is, we first relax the feasibility constraints for a given problem and find a solution to this problem. Next, we check if this solution applies for the problem with strengthened constraints. If not, we improve the solution until it is valid for these new constraints. This process of strengthening and improving the solution is repeated until a solution to the original problem is found (or the iteration limit is reached). A block diagram illustrating our approach is shown in Figure 8 and pseudo-code is provided in Algorithm B.1.

Identifying and relaxing constraints (Step 2) and improving the solution for the strengthened constraints (Step 5) are the most important steps in Algorithm B.1. Next we will see how these steps can be implemented.



(a)

(b)

(c)

(d)

Fig. 7. Using an approximate (invalid) path to find a solution to the original problem. (a) Original C-space. (b) C-space after relaxing the constraints. (c) Approximate path. (d) Solution to the original problem after fixing the approximate path.

Algorithm B.1 Iterative Relaxation of Constraints

```
Relax feasibility constraints (set to minimal values)
Find a valid solution for the current feasibility constraints (an Approximate Solution)
while Original problem is not solved or iteration limit is reached do
    Strengthen feasibility constraints
    Modify/Improve current solution until it is valid for current feasibility constraints
end while
```

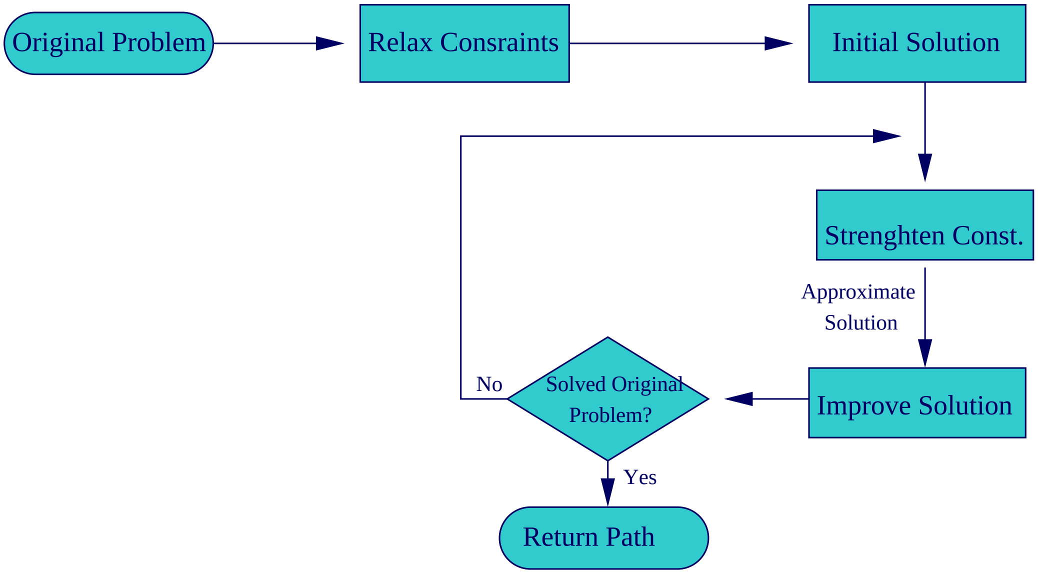


Fig. 8. Flow diagram of IRC

C. Feasibility Constraints and Relaxation

By relaxing a feasibility constraint, we hope to obtain an easier version of the problem whose solution, if invalid for the original problem, can be used to guide the automated planner in solving the original problem. Hence, the selection of feasibility constraints and the relaxation methods will depend on the problem. For example, if the problem is moving a car-like non-holonomic robot in a narrow environment, the feasibility constraint can be the penetration. In the original problem, the robot must not penetrate an obstacle. But this constraint can be relaxed by allowing some small amount of penetration. For an arm like robot, the relaxation can be done by adding one extra degree of freedom (an extra joint). Once a path is found for the end effector, then the joint values for the original robot can be found.

In the *virtual prototyping* and *deformable objects* domains we are interested in finding a path. Hence, our use of relaxed feasibility constraints returns an "Approximate Path" (Step 2). In *ligand binding* we are interested in generating candidate binding sites and our relaxation of feasibility constraints results in the acceptance of high energy configurations.

At Step 3 of our algorithm, we check if the approximate solution is valid for the original feasibility constraints of the problem. If not, we strengthen the feasibility constraints and improve the solution (Steps 4 and 5). Below, we describe the constraints and relaxation methods we applied in our application domains. In the next section, we describe how the approximate solution is improved the different domains. methods.

Virtual Prototyping

Our goal in this domain is to find a path for a rigid body part. The resulting path should be collision free. Hence, the constraint in this domain is collision. We relaxed this constraint by scaling the robot which may result in penetration in the solution path for the original version of the problem. In this domain, we also investigate finding a solution with user assistance in which the user collects some configurations in the region of the actual solution path using haptic force feedback. Due to heuristic collision detection (for fast force update), the user collected configurations may penetrate obstacles, which is infeasible for the original problem, but will be accepted in the approximate solution path.

Ligand Binding

Our goal in this domain is to to find a feasible binding site for a given protein-ligand complex. We used the energy as the feasibility criterion for this problem. This criterion is relaxed by accepting configurations with higher potential energy.

Deformable Objects

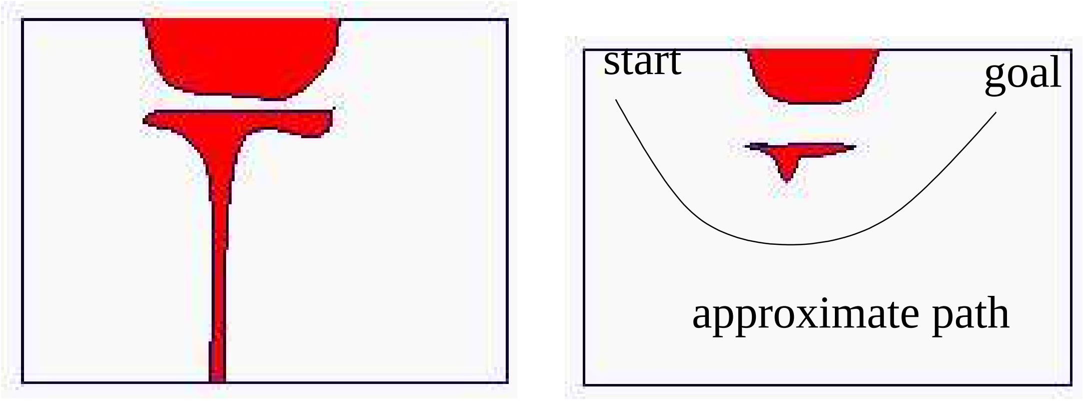
In this domain, we want to find paths for deformable robots. Our initial path is planned for rigid body robots. We use a similar criterion as for *virtual prototyping*, where a configuration is feasible if the robot is collision free or has little penetration. To achieve this, we utilize two feasibility metrics, the collision status of a reduced-scale version of the robot and the penetration depth of the robot.

D. Improving the Approximate Solution

Once a feasible solution is found for the relaxed constraints, Step 4 of Algorithm B.1 strengthens the feasibility constraints. Step 5 improves the solution if the solution for the relaxed constraints does not hold for the current constraints. The search for the solution can be found by performing a local search for valid configurations near the invalid configurations. The search can be implemented in either C-space or the workspace. In *virtual prototyping*, a path with relaxed constraints may contain collisions. The planner should improve this solution by transforming invalid configurations (in collision) into valid configurations. We have investigated three methods to search for valid configurations: (i) allowing the last valid configuration before and the last valid configuration after an invalid segment to direct the search, (ii) directing the search towards the closest valid configuration to an invalid configuration, (iii) using workspace information to choose a search direction. In *ligand binding*, low energy configurations of the ligand are desired because they are more stable. Our relaxed constraint may return high energy configurations. In order to improve such configurations, we implemented an approximate gradient-descent search. In our last domain, *deformable objects*, our relaxed solution may contain configurations with collisions. When we improve the solution, the robot is deformed to avoid the collisions.

E. Choosing Initial Constraint Values

Finding initial constraints plays an important role in IRC. If the initial relaxation is too small, the relaxed problem may be as hard to solve as the original problem. If the initial relaxation is too much, it may result in topological changes in the free configuration space. In such cases, it may be very difficult, or even impossible to transform the invalid solution into a valid solution. Figure 9 shows such a case. Figure 9(a) shows the C-space for the original problem. After the relaxation, a path can be found as in Figure 9(b) which is hard to improve since the solution to the original problem has to pass through the narrow passage.



(a) (b)

Fig. 9. If the constraints are relaxed too much, it might be hard to transform the approximate solution into a solution to the original problem. (a) Original C-space. (b) Approximate solution found in a relaxed C-space.

An approach to determine the initial relaxation could be to build several roadmaps with different relaxation values and then evaluate the resulting C-spaces. Hsu et al [57] suggested ϵ-goodness as a criterion to characterize C-space. The problem with this approach is that, although ϵ-goodness defines a good theoretical representation, it is hard to compute for all but the simplest problems.

In our approach, if the automated planner can't solve the problem within the current constraint set, we relax the constraint and check if a solution can be found. Relaxation and checking can be applied until a solution is found. Once a solution is found, the constraints would be the initial constraints for the IRC. Since this linear relaxation could potentially be slow to converge, we suggest a binary search between the original constraints and some minimum constraints. For example, if the reduced-scale version of the robot is used in the relaxation of constraints, the search starts with a very small robot and the scale value is changed based on the binary search. Similarly, if penetration is used for constraint relaxation, the search starts with a large penetration and changes it as the binary search progresses.

CHAPTER IV

VIRTUAL PROTOTYPING

Since the introduction of CAD/CAM systems, the design process in industry has become cheaper, easier, and faster. Once a design is prepared, the next step is to validate the design based on several criteria such as the structural integrity of the design, the parts' ability to fit together, their accessibility, etc. Implementation of the design phase in the computer is straight forward since there are minimal computational tasks. However an automated solution to the validation phase may require very complex algorithms, as in the case of checking part accessibility. Due to such difficulties, the validation of a design is still usually done by building a physical mock-up and manually validating the design. This process is costly, time consuming, and error prone.

In this chapter, we propose an algorithm for validating a design for part accessibility. Part accessibility plays an important role in the maintenance of a product. Each part must be accessible so that a malfunctioning part can be replaced by a mechanic during the life time of the product. This problem is naturally formulated as a motion planning problem. In particular, the parts can be represented as rigid robots that can translate and rotate in three-dimensional space. A part's accessibility from outside then can be checked by finding a collision free path for the part (i.e., rigid robot) from an outside position.

The motion planning approach to maintenance problems suffers from the *narrow passage* problem we have seen in Chapter III. In order to address this, we apply the Iterative Relaxation of Constraints (IRC) technique. In our approach, we first find a solution for a smaller version of the part we are checking for accessibility, then we use that solution to guide the automated planner in its search for a removal path for a slightly larger version of the part. We repeat this process until we find a removal path for the original part or we fail to find a removal path after some period.

One of our observations about automatic motion planners is that, they sometimes fail due to the difficulty of discovering ‘critical’ configurations of the ‘robot’ that are in tight, or crowded, regions of the robot’s configuration space but which are crucial configurations in the resulting path. Even the IRC approach may not be sufficient if the environment if too crowded. In contrast, such configurations are often naturally apparent to a human observer (see, e.g., [5]). On the other hand, automatic methods are very good at computations that human operators find cumbersome and/or overly time consuming, e.g., detailed computations necessary to fully determine a continuous path.

In this work, we consider how to incorporate the strengths of both a human operator and an automatic planning method. In particular, we investigate how haptic and visual interfaces can be used to enable a user and an automatic motion planner to cooperatively solve a motion planning query. Haptic interfaces enable the user to feel and naturally manipulate virtual objects by estimating the forces generated by the user’s contact with the virtual objects and generating contact forces to be delivered to the user via the haptic device. These contact forces allow the user to “feel” the virtual object. For example, the force needed to prevent penetration into a virtual surface is often correlated to the depth of penetration and is delivered in the direction of the surface normal (see Figure 10). Such calculations require geometric knowledge such as penetration depth, and dynamic parameters such as velocity, elasticity, and damping coefficient.

We consider a scenario in which the human operator manipulates a virtual object

---

[1] This research is published in [8, 21, 22].

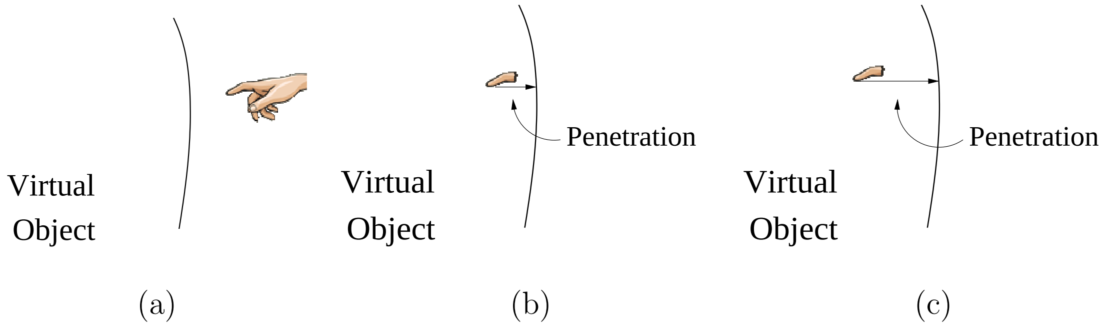


Fig. 10. Penetration in haptic feedback. (a) There is no penetration, so the user feels no touching sense, e.g., no force feedback. (b) There is a little penetration, so the user is pushed back in the direction of the penetration vector and feels a force that is a function of the depth of the penetration. (c) The user has a greater penetration, so greater force is applied in the direction of the penetration vector, giving the sense of touching a solid object.

using the haptic interface, captures (some) configurations of the object, and passes them to the planner. The configurations that the human operator captures can also be seen as an approximate path. The planner then attempts to use these configurations to generate a valid motion plan for given constraints. The planner's progress is communicated to the operator using a visual overlay on the virtual scene. In this work, in addition to evaluating the IRC approach, we try to answer the following questions:

- How can the motion planner best utilize the user-generated input?
- What are 'natural' ways for the user to understand the progress made by the motion planner?

Outline. Section A discusses the previous work. Section B describes our prototype VE system and outlines the automatic motion planner used in our study. In Sections C and D, we describe the approximate solution to the problem and how to improve it. Section E presents our experimental findings and Section F concludes this chapter.

A. Related Work

Although our goal of having a human operator and an automatic motion planner work cooperatively to solve a motion planning problem using haptic and visual interfaces is novel, work in related areas, i.e., human-computer cooperation, haptic interfaces, and motion planning, is abundant.

In human-computer cooperation, especially teleoperation [42, 50, 110], researchers have addressed many of the same challenges we face in our work. For example, current robots cannot operate completely autonomously in complicated environments due to inadequacies in sensor and image processing technology [42]. On the other hand, purely human operation encounters problems such as tedium, fatigue, and operator skill requirements. In telerobotics, there has been some some work with goals similar to ours. Das [37] describes how a ten dof linkage is cooperatively controlled by a human operator and a computer. The operator guides the position of the end effector, while the computer plans the joint angles to avoid collisions. Guo *et al.* [50] show that a cooperative telerobotic system can finish some (planning) tasks that are not achievable by either party alone. They also mention how semi-autonomous recovery can be performed by providing the computer with some simple obstacle avoidance hints. This is similar to our situation, in which operator hints can aid discovery of narrow C-space corridors and planning within these corridors.

Visual interfaces have been extensively studied, and much progress has been made in the area of visualizing contacts and collisions among virtual objects. Although this is very helpful, many realistic tasks are sufficiently complex, requiring delicate translation and rotation at the same time, that visual images alone become insufficient [34]. In particular, haptic interfaces are indispensable for tasks which involve touch and manipulation [34, 125], since touch is the only sense which enables the user to both receive feedback from and transfer influence to the environment [111]. Most importantly, haptic interfaces can improve performance and ease problem solving. For example, Fabiani *et al.* [43] and Richard *et al.* [107] found that haptic feedback increased performance (decreasing errors) by about 50% over the case without haptic feedback. Similarly, Maneewarn and Hannaford [87] suggest that haptic feedback improves telecontrol performance near kinematics singularities, and Taylor *et al.* [61] show that haptic feedback is essential in finding the right starting position and guiding changes along a desired path during surface modification using Scanned-Probe Microscopes.

Indeed, there is growing interest in haptic interfaces, both in development of new devices [25, 85, 43], and in their application to a variety of fields such as CAD [86, 97], training [36], nanorobotics [113], teaching [73], and volume visualization [15]. One major challenge faced with haptic devices is obtaining realistic force feedback in real-time, which requires on-line collision detection and penetration depth calculations. For complicated environments, force feedback delay is inevitable as update rates as high as 1000 times a second may be required to maintain a stable system [48]. Similar problems are encountered in teleoperation, in which methods to deal with visual image delay have received much attention. For example, Jagersand [62] proposes an image-based extrapolation method for this problem. Several approaches have been proposed for rendering force feedback with haptic interfaces, e.g., the magnitude and direction of the force if in collision, including approximate, augmented or new collision detection methods [36, 48, 54, 111], some for flexible objects [34], or separate processes for collision detection and servo loop updates [111]. In our work, we employ

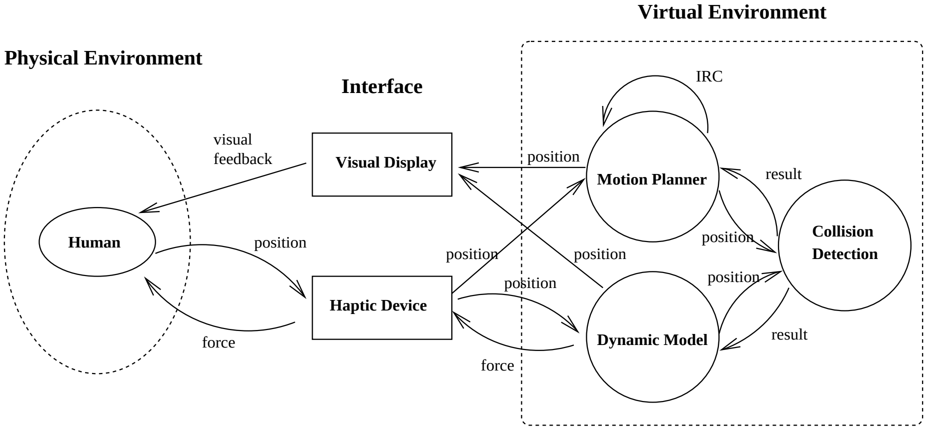


Fig. 11. System architecture

techniques similar to these such as an extrapolation method for dealing with delay and separate processes for collision detection and servo loop updates.

B. Overview

The architecture of our system is described in Figure 11. Our prototype system consists of three components: input, output and motion planning. The motion planning component solves the motion planning problem within the current feasibility constraints. The inputs to the motion planning component are the original problem and an approximate path (either user collected configurations or an intermediate solution generated by the automated planner). The progress of the automated planner is shown to the user through an output component. The system is composed of a PHANToM haptic device (6 dof input, 3 dof output) [89], an SGI O2 graphics workstation (graphics display and position tracking), and an SGI Octane (collision detection). The operator uses the PHANToM to manipulate a single rigid object in the virtual scene. Haptic interaction consists of the following steps: (i) tracking the user's motion, (ii) detecting collisions between the user controlled probe and virtual objects, (iii) computing appropriate reaction forces, and (iv) exerting forces on the user via the PHANToM. Our haptic-interaction applications were developed using the C++ *General Haptic Open Software Toolkit* (GHOST SDK) [109].

Although our system is fairly simple, we encountered many of the technical challenges involved in providing realistic feedback to the user. For example, collision detection and penetration calculations are a major issue since the motion planning problems we are interested in deal with complex CAD models (some involving hundreds of thousands of polygons). On the other hand, our objective is different from most haptic applications, i.e., our primary goal is *not* to provide the user with a realistic experience, but to help him collect information for the planner. As described below, this relaxation of the 'realistic' requirement enables non-conventional approaches to some common problems faced in haptic interaction.

1. Achieving Useful Force Feedback

It is well known that to achieve realistic haptic feedback, all steps should be executed with approximately 1 kHz frequency [48]. However, in our applications involving complex CAD models, the collision detection rate was highly variable, averaging around 10 Hz – which is 100 time slower than needed (see Section E.4). Thus, to achieve realistic interaction rates, we need to determine some reasonable feedback to apply while waiting the collision detection results. Our solution was to use a separate process (either on the local machine or on the remote machines) to compute the collision and to use a heuristic to decide if a collision occurred while waiting for the decision from the collision detection process. Figure 12 illustrates this approach. Note that since we collect approximate paths anyway (Section C), a heuristic is an acceptable solution (i.e., undetected collisions are acceptable).

Our heuristic works in the following way. Consider the scenario shown in Fig-

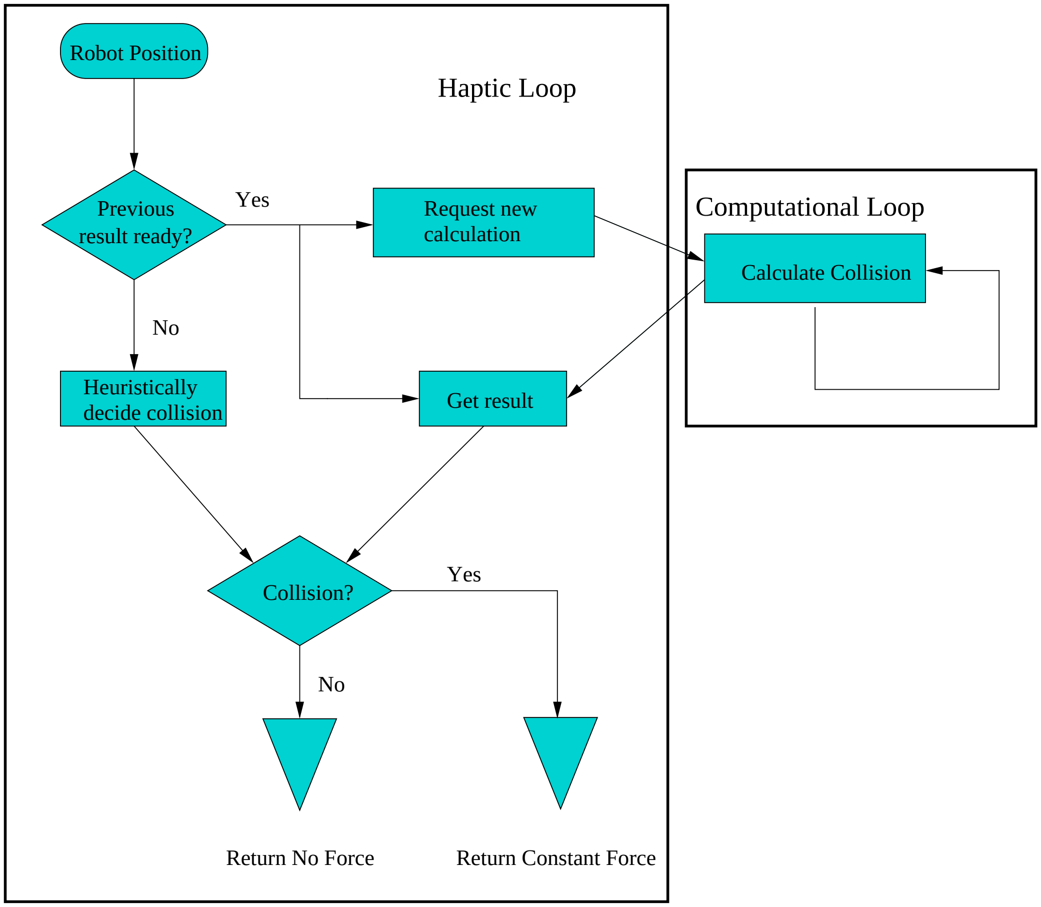
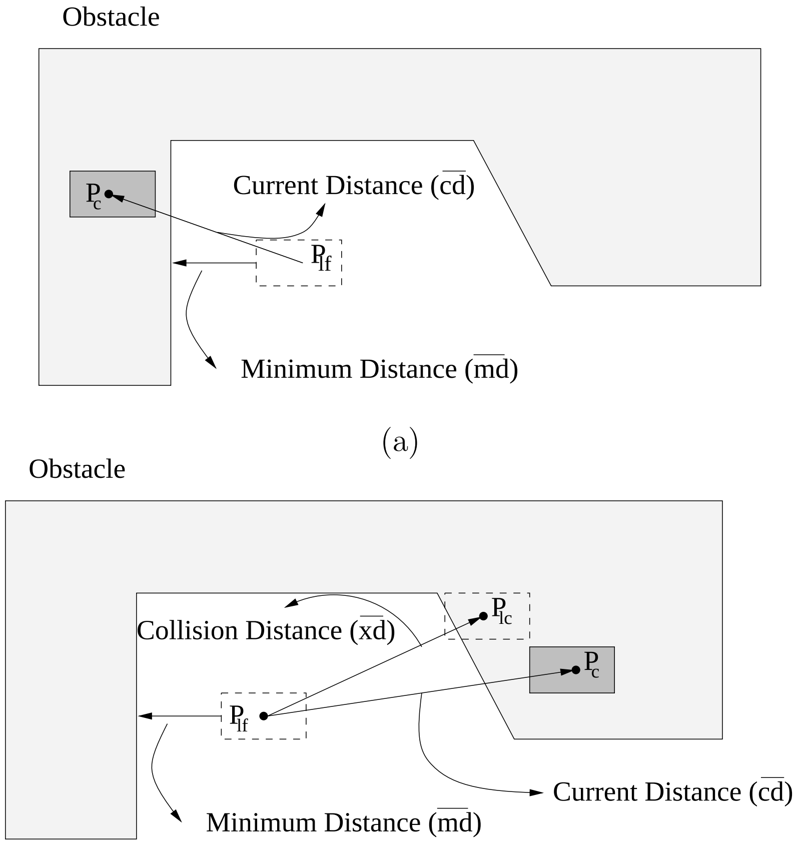


Fig. 12. Flow chart of haptic process. The haptic process checks the results of collision detection in the external process. If it is ready uses that result, otherwise heuristically decides if a collision occurred.


Obstacle
P_c
Current Distance ($\overline{cd}$)
P_{lf}
Minimum Distance ($\overline{md}$)
Obstacle
Collision Distance ($\overline{xd}$)
P_{lc}
P_c
P_{lf}
Current Distance ($\overline{cd}$)
Minimum Distance ($\overline{md}$)

(a)

(b)

Fig. 13. Collision detection heuristic when last known result was (a) free and (b) a collision.

ure 13(a). Suppose the last collision detection call returned a collision free result when the robot was in the configuration c_{lf}, and let md denote the minimum distance vector from c_{lf} to an obstacle (also returned by the collision detection call). While the system waits for the result of the next collision detection call, the user continues to move the robot. During this time, the PHANToM tracks the current configuration of the robot c. If the projection of cd (the vector from c_{lf} to c) on md is greater than md, then our heuristic determines that a collision occurred and applies a force opposite to md. Now suppose the last collision detection request returned a collision when the robot was in configuration c_{lc} (Figure 13(b)), and let xd denote the vector from our last known free configuration to c_{lc}. In this case, we heuristically determine the robot is still in collision if the projection of cd on xd is greater than xd, and apply a reaction force opposite to xd.

Our experiments (described in Section E.4) showed the importance of the heuristic function by illustrating that almost all collision detection calls are handled by the heuristic function.

2. Variable Reactive Forces

In the current system, the forces can be applied by the PHANToM in three modes: maximum-force, half-force, and zero-force. This variable strategy enables users to choose between freedom and restriction of movement. For example, in the half-force case if the robot is close to the obstacle, the user will be aware of the closeness while still being able to move inside the obstacle if he desires. However, in the maximum-force case, he will be forced to move away from the obstacle once a collision occurs. The zero-force case provides complete freedom of movement and is implemented by turning off the collision detection module.

3. Roadmap Visualization

For the human operator to help the automatic planner, he must first understand where the planner is having difficulty. This is, however, a challenging task since most planners, work in C-space, which is a higher dimensional space than the graphical interface available for display. The naive method of simply displaying a configuration in the workspace is acceptable if only a few configurations will be shown. However, if many configurations must be displayed simultaneously (e.g., a roadmap), then the resulting scene could be as difficult to understand as a high-dimensional configuration space. Thus, what is needed are methods of ‘projecting’ configurations into the

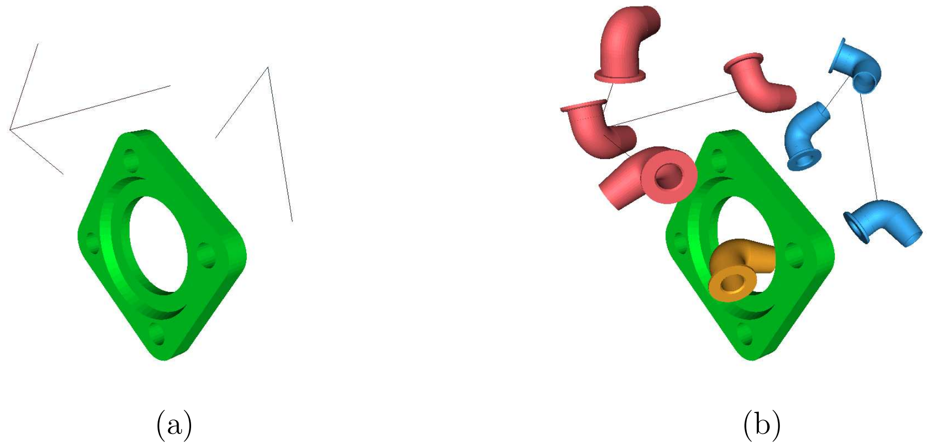

(a) (b)

Fig. 14. Roadmap visualization. The obstacle is the hole and the robot is an elbow shaped object. (a) Only roadmap edges are shown. (b) Roadmap configurations are shown in reduced scale, with a gray scale for each connected component.

workspace. We have implemented two such methods.

In the first, we represent each robot configuration by a single point in the workspace, e.g., the positional coordinates of a reference point on the robot's local reference frame. Connections between configurations are displayed as edges between their projections. Since multiple configurations can project to the same workspace representation, we use colored edges to differentiate roadmap components (see Figure 14(a)).

A problem with this first approach is that all orientation information is lost, and this can be very important in crowded environments. Our second method addresses this problem by displaying a reduced-scale version of the robot, in its actual orientation, instead of a single point. This method proved more valuable for the operators, while still avoiding excessive cluttering of the scene (see Figure 14(b)).

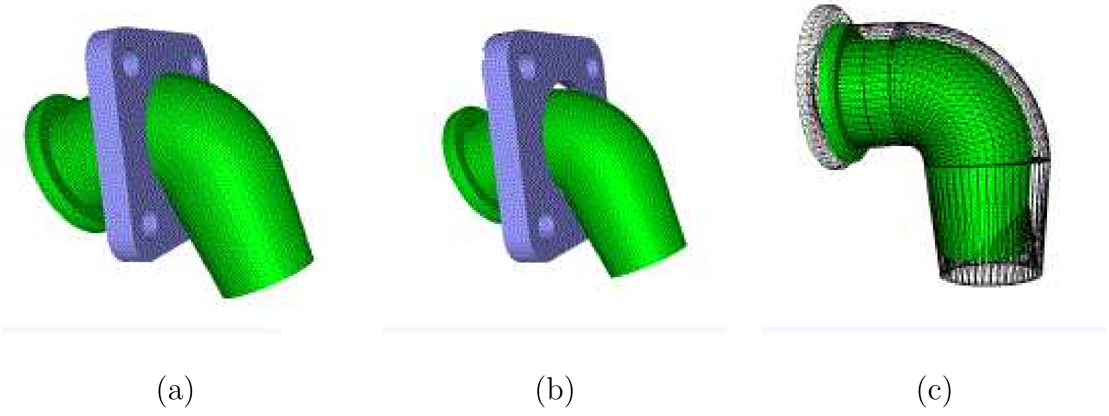
C. Approximate Solution Generation

As discussed in the previous sections, an initial path was generated either automatically or manually. In either case, the initial path may contain collisions for the original problem. In this section, we describe how we generate the approximate path and in the next section we will see how we can improve that path.

1. Automated Path Generation

In the IRC paradigm, Our strategy is to solve a simplified version of the original problem (one may do this, for example, by scaling the robot), and then use the resulting solution path as an 'approximate path' input for the harder problem. For the virtual prototyping applications, we simplified the original problem by relaxing the collision free constraint. This is done by using a reduced scale version of the robot. Figure 15 (a) shows an example case. The elbow shaped object is the robot and the hole is the obstacle. Figure 15 (a) is hard to solve since it is a tight fit and the robot has little free space to maneuver. If a reduced-scale version of the robot is used as in Figure 15 (b), the problem becomes easier since the robot has a larger space in which to maneuver. This scaling can be seen as an approximation to the amount of penetration the robot would have in the obstacle (See Figure 15 (c)). Hence, an initial path can be found by building a roadmap with the reduced-scale version of the robot. The path found in this roadmap then may contain some configurations that are not collision free for the original robot but are collision free for the scaled robot.

This approach is similar to the idea of building a roadmap in a dilated freespace [57]. The main difference is that we concentrate only on a select path, which is close (or at least hopefully transformable) to a valid path for the original problem. That is, we are focused exclusively on the important regions of the free space.



(a) (b) (c)

Fig. 15. Relaxation of collision constraint. (a) The original problem is hard. (b) If the size of the elbow shaped robot is reduced the problem becomes easier. (c) The difference between the original robot and the reduced-scale robot. Scaling can be seen as an approximation of the amount of penetration the robot can make in the obstacle.

2. Manual Path Generation

The operator's role is to capture configurations that will be useful for the planner. For example, the operator could help the planner by viewing the workspace representation of the evolving roadmap, and capturing paths of configurations that will enable the planner to connect different roadmap components.

In our implementation, the operator generated sequences of configurations (*approximate paths*) by attaching the haptic device to the virtual robot and manipulating it in the virtual environment. The system recorded these configurations at regular intervals, and then passed them to the planner.

The configurations generated by the operator could be **free** (i.e., collision-free configurations of the robot), or **approximate** (some penetration of the robot into the obstacles is allowed, which can be viewed as 'dilating' the robot's free space [57]). In the former case, the planner can directly incorporate the path, while in the latter, it must first transform it into a free path (note that this is not always possible). In both cases, the planner can use the temporal ordering of the collected configurations to speed the connection process.

D. Improving the Approximate Solution

The planner's goal is to use approximate paths to solve the original problem. If an approximate path that contains collisions with environmental obstacles in the original problem space (i.e., passes through C-space obstacles) was collected, then the planner must attempt to 'push' the colliding portions to the free space (based on the current feasibility constraints).

One approach to find a collision free path might be to use the colliding configurations of the approximate path as milestones in the dilated freespace as in [57] or as seed nodes in OBPRM [6] (which attempts to 'push' them to the surface) and try to find a path on the resulting roadmap.

However, in our scenario, more efficient methods are possible because the provided paths are expected to be 'close' to a valid path. Below, we first propose methods for pushing an approximate path to the free space, and then describe an *iterative pushing* technique within the IRC paradigm. Coupled with the pushing methods, this enables one to transform a solution for a much easier version of the problem (valid in the dilated freespace [57]) into a solution for the original version.

1. Pushing Methods

In this section, we describe more specialized methods for transforming colliding path segments to the free space. Since the collided segment is expected to be near a valid segment, our main challenge is to determine the best direction in which to

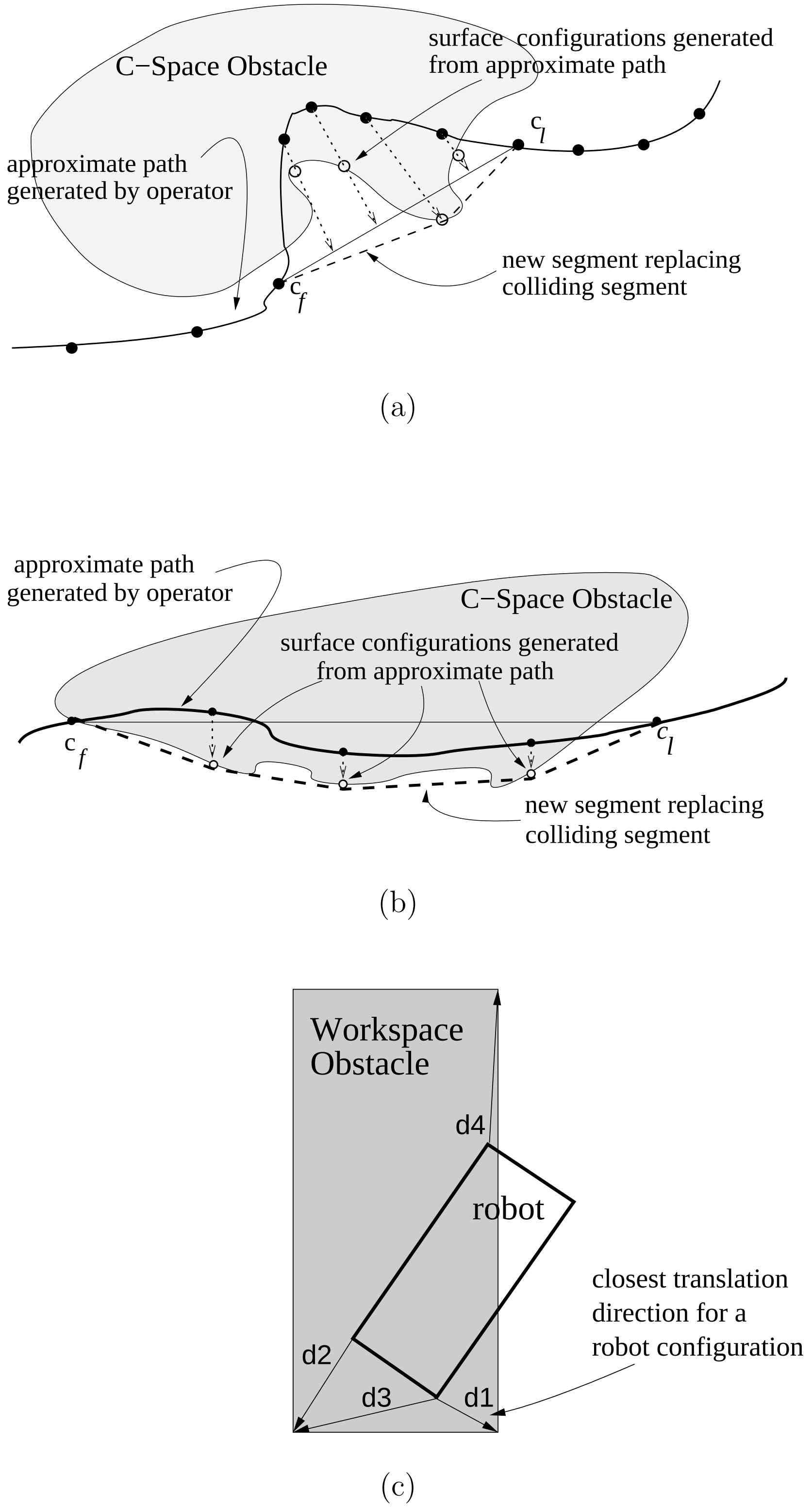

C−Space Obstacle
surface configurations generated from approximate path
approximate path generated by operator
c_l
c_f
new segment replacing colliding segment
(a)
approximate path generated by operator
C−Space Obstacle
surface configurations generated from approximate path
c_f
c_l
new segment replacing colliding segment
(b)
Workspace Obstacle
d4
robot
closest translation direction for a robot configuration
d2
d3
d1
(c)

Fig. 16. Pushing approximate paths to free space: (a) simple push ($k = 1$), (b) shortest push, (c) workspace assisted push.

'push' the colliding segment. Below, we describe several methods for performing this deformation.

The general scenario we consider is as follows. We assume there is a single colliding portion of the path; if there is more than one, each can be considered separately. We use c_f and c_l to denote the free nodes immediately preceding and following the colliding segment, and let n_c denote the number of configurations in the colliding segment. Generally, we select $n_{\text{push}} = \lfloor cn_c \rfloor$ of the colliding configurations to push to the free space, for some constant $0 < c \leq 1$.

Simple push. In this method, we consider the straight line (in C-space) between c_f and c_l, and select kn_{push} configurations evenly spaced on this line, for some constant $k \geq 1$. Each selected configuration on the colliding segment is then paired with k of these intermediate configurations, and we try to 'push' the colliding configurations in the direction of the intermediate configuration in search of a contact configuration. (See Figure 16(a)).

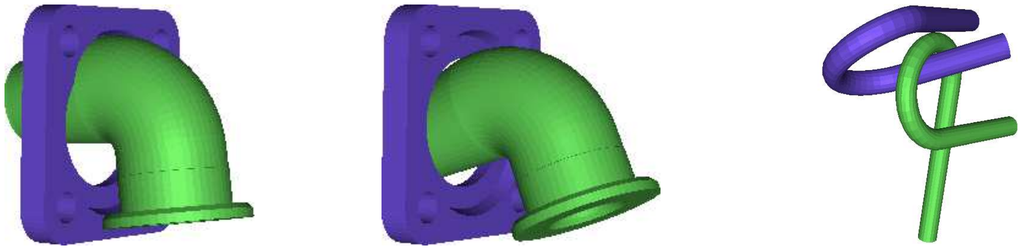
Shortest push. In this method, we attempt to find the shortest distance (in C-space) between a colliding configuration and the free space. In particular, we select a set of n_{push} configurations evenly distributed on the colliding segment, and process them in the following way. For the first node, we search in many directions in parallel until we find a "closest" free node. If we treated every node this way, this method would be slow. Instead, for each successive node, we use the previous direction found as a guide, and search a cone of directions centered around that direction. In addition to guiding our directional search, this approach also allows us to take larger steps for subsequent nodes (we choose some high fraction of the previous displacement as the step). This can dramatically reduce the running time and makes this method very efficient. Note that this method is only applicable since the input nodes are in sequence and close to their neighbors (a path) (see Figure 16(b)).

Workspace-assisted push. In this method, we try to take more explicit advantage of the fact that the user-generated colliding path is usually close to a free path. In particular, we use geometric knowledge about the workspace to select a direction to *translate* the robot. Again, we select some set of n_{push} configurations evenly distributed on the colliding segment. For each configuration selected, we find the m closest pairs of vertices (v_r, v_o), one on the robot (v_r) and one on an obstacle (v_o) (we assume objects are modeled as polygons). For each vertex pair (v_r, v_o), we choose the direction from v_r to v_o, and *translate* the robot in that direction, looking for the first free configuration (see Figure 16(c)). Note that this method will fail in cases in which rotation is required to reach a free configuration. In current work, we are considering how to augment this method to detect when such movements are required.

E. Experiments

In this section, we present the results of experiments designed to evaluate IRC. We also investigate whether a human operator and an automatic motion planner can work together to solve a motion planning query within the IRC paradigm. Our experiments were designed to determine if haptic input is useful, to evaluate our various pushing methods and to observe the behavior of the heuristic collision detection. Our experiments were structured as follows:

1. An initial roadmap is generated by OBPRM.
2. An approximate solution is found.
    (a) The human operator is shown the roadmap, and uses the PHANToM to generate approximate paths for the planner.



(a) (b) (c)

Fig. 17. The flange and alpha-puzzle problems.The flange problem is shown in (a) and (b): (a) a colliding configuration from the user-generated approximate path, and (b) the resulting collision-free configuration it was pushed to. For the alpha puzzle (c), we show a narrow passage configuration found by the workspace directed push method.

   (b) A reduced-scale version of robot is used to find approximate paths.

3. The initial roadmap is improved using the approximate paths supplied and one of the pushing methods.

1. Environments

Our experiments involved two environments: the *flange* and the *alpha puzzle.* The flange consists of a fixed rectangular part with a circular opening (the obstacle, 990 triangles) and a curved pipe (the robot, 5306 triangles) that must be inserted into the opening in the obstacle (see Figure 17(a)). The alpha puzzle consists of two identical twisted, intertwined tubes (one the obstacle and one the robot, 1008 triangles each); the objective of the puzzle is to separate the tubes (see Figure 17(c)). For the flange, we used the original version, as well as slightly easier versions of the model, which were obtained by scaling the pipe by a factor of .85 and .95. For the alpha puzzle, slightly easier versions were obtained by scaling the obstacle tube by a factor of 1.2 in one dimension, which widened the gap between the prongs of the alpha shape.

Table I
Comparison of Different Approaches

Env	Method	Running Time(secs)	
		Gen/Push	Connect
Flange .85	OBPRM	304	119
	Haptic Push	28	1.7
Flange .95	Haptic Push	416	2653
	IRC .85 Path	86	18
Flange 1.0	IRC .95 Path	586	1100
Alpha 1.5	OBPRM	30	1649
	Haptic Push	20(9.4)	309(3.6)
Alpha 1.2	Haptic Push	24(6.8)	274(3.0)
	IRC 1.5 Path	31(10)	420(5.0)

Node generation/pushing and connection times for each method to successfully solve the query in the specified environment. Values shown are complete processing times; numbers in parenthesis show only time spend processing haptic input. The label "IRC .85 Path" indicates that IRC used the solution path for the .85 version as input, and similarly for others.

2. Overall Evaluation

A summary evaluation of the methods is shown in Table I. In both environments, the haptic hints not only enabled us to solve the problems much faster, but also allowed us to solve harder problems than we were able to solve using the fully automatic method alone. We believe the reason for this is that human operator can easily spot the crucial locations while the machine must search for them, more or less, blindly.

The results for the flange are most dramatic. Our fully automatic planner was only able to solve the .85 version in a reasonable amount of time, but starting with the haptically generated path, we were able to solve the original problem quite quickly using the IRC technique. Moreover, the fast solution times for the iterative method show that using a valid solution path for an easier version of the problem as the starting point (approximate path) for a harder version is a very useful technique for the flange. An interesting observation is that the connection time for haptic pushing is less than the generation/pushing time. This indicates the method generates a good set of nodes that can be connected easily. Finally, we note that the hints (i.e., the approximate paths) given to the automatic planner not only increased the solvability of a problem but also reduced the solution time dramatically. For example, the generation of the roadmap for the flange .85 version took 423 seconds with the automatic planner, and only 30 seconds for the haptically enhanced version.

For the alpha puzzle, we observed similar results. That is, the haptic hints greatly reduced the solution times and enabled us to solve harder problems than with the fully automatic planner. One difference noted was that the iterative method works much better for the flange than for the alpha puzzle. This can be explained by the different geometrical structures of the two problems. In particular, in the alpha puzzle the tube can slide out without much rotation, while in the flange, the solution path requires continuous rotation. Therefore, for the flange, it becomes more difficult for the user to collect a good approximate path as the problem gets harder (fine rotational movements are required). As a result, the iterative method finds the solution faster since the quality of the approximate paths it uses (solutions to easier versions) are of higher quality than the user-generated path. However, for the alpha puzzle, since little rotation is required (for the versions we considered), the difficulty of collecting a good haptic path does not vary much when problem gets harder, and thus the iterative technique does not provide the same advantages as for the flange. Although it appears as if the iterative method actually took longer than the direct method, this is heavily dependent on the quality of the approximate path collected, and the results above indicate that we collected a relatively better path for the 1.2 version than for the 1.5 version, i.e., the operator happened to do a better job in the 1.2 version.

3. Evaluation of Pushing Methods

In this section, we evaluate the various pushing methods used to improve the approximate solution (Section C) in terms of their ability to generate useful free configurations, and in particular, their ability to generate so-called ‘critical’ configurations of the robot that were in narrow passages in C-space.

For both environments, all three methods were tested on the same user-generated path. However, a filtering process was used with the *simple* and *workspace assisted* push since both methods may generate outside (irrelevant) nodes because they search in only one direction for a free node. We retained a generated node only if collisions resulted when it was moved in both directions along its principal axes. Also, if there were many close configurations in C-space, we retained only one as a representative. This greatly reduced the number of configurations (and subsequent connection times), and visual inspection showed it worked very well.

Table II summarizes our results. All methods performed slightly better for the alpha-puzzle than for the flange. This is because the boundaries of the narrow passage in the flange are quite thin, and so it is quite easy to push the configurations completely outside the passage. This was not as true for the alpha puzzle. In terms of execution times, the flange had slower node generation times since we tried to generate nodes in the hole.

The *simple* and *shortest* push methods had comparable success in generating nodes in narrow passages; the *simple* push generated more configurations because each colliding configuration was paired with several intermediate configurations (whereas the *shortest* push generates only one configuration for each colliding configuration). The *workspace assisted* push had lower success rates.

The winner of the these three methods is unmistakably *shortest push* since it has

TABLE II
PUSHING METHODS

Env	Method	Passage	%Success	Time(secs)
Flange	simple	211	91%	0.51
	shortest	140	100%	0.21
	wk.asst.	68	77%	2.81
Alpha	simple	600	98%	0.24
	shortest	250	100%	0.07
	wk.asst	51	61%	1.95

The number of nodes in the passage, the success rate and the average node generation time for each method are shown above.

the fastest node generation time and the highest success rate with a small number of nodes being generated (which is an advantage in the connection phase). However, we note that the *simple push*, and especially the *workspace assisted* push, can capture critical configurations. Their main disadvantages are that *simple push* generates an excessive number of nodes (even after filtering) and hence slows down the connection phase, and *workspace assisted* push has a lower success rate.

4. Evaluation of Heuristic Collision Detection

Table III shows collision detection statistics for the haptic system we used for our virtual prototyping experiments. This table shows the percentage of collision detection results that were actually computed vs. those that were supplied by heuristic decisions. It also shows the average computation time for required collision detection in the server and the average wait time for the client after it requests for the collision detection. The importance of the heuristic function can be seen in the table since almost all collision detection calls are handled by the heuristic function. We also note that the client's (i.e., the haptic system) average wait time for a collision detection request is approximately twice as long as the average collision detection computation

Table III
Collision Detection Statistics

	Percentage		Timing (ms.)	
Env	Heuristic	Real	Wait.	Coll.Det.
Flange	98.6%	1.4%	71.47	34.61
Alpha	99.0%	1.0%	99.30	40.16

Collision detection statistics: percentage of heuristic decisions vs real results, average wait time for collision detection results by the client, and the average collision detection time on the server.

time on the server. This delay is caused by the interprocess communication (the actual collision detection computation was done on another computer). However, even with this delay, the wait time is significantly shorter than the wait time when we had computed the collision detection locally. Our experiments showed that the wait time would be around 500-700ms since our local machine had to process not only the computations but also the interface functions such as haptic interaction or visualization.

F. Conclusions and Future Work

This chapter reports on our investigation regarding a "hybrid" (manual + automatic) motion planner based on the IRC (Iterative Relaxation of Constraints) paradigm which uses haptic and visual interfaces to enable a human operator and an automatic planner to cooperatively solve a motion planning query. Our results indicate that:

- A heuristic algorithm based on the last known collision result can be used effectively to get reasonably realistic feedback for approximate path (containing collisions) collection.
- The automatic planner can effectively transform approximate paths into free paths.

- Iterative Relaxation of Constraints is a promising approach. The methods described in this chapter can be applied to the paths found by any automatic planner which solves a simplified problem, and the resulting paths then can be used as the starting point for a harder version of the problem.

- Roadmap visualization is useful since it gives the human the ability to understand the automatic planner, understand and compensate for its drawbacks, and adjust the planner parameters if required.

Our future work includes increasing the realism of the haptic feedback by reducing collision detection time (using multiple servers and faster algorithms) and improving the pushing methods by using more hints from the workspace (such as using the medial axis or the shape of the workspace objects).

CHAPTER V

LIGAND BINDING

Efficiency of a drug molecule is measured by its ability to find a specific position and orientation inside a protein (see Figure 18). This process is called binding (sometimes called docking) and the drug molecule is often referred to as a ligand. The binding configuration should satisfy some constraints based on geometry, electrostatic, and chemical reactions between the ligand and protein atoms. A good binding site should also be accessible to the ligand which must reach it from an outside location. This makes the path to the binding site important, and motivates the use of motion planning to study this problem.

Many researchers investigating automated docking methods simplify the problem by treating the ligand and protein molecules as rigid bodies. Our experiments show that this simplification negatively impacts our ability to identify the binding site. However, if the molecules are flexible, i.e., the molecules are articulated bodies, the problem becomes very high-dimensional and a deterministic search of the configuration space is practically impossible. An intermediate alternative employed in some work, including ours, is to treat the protein as rigid and the ligand as flexible. In

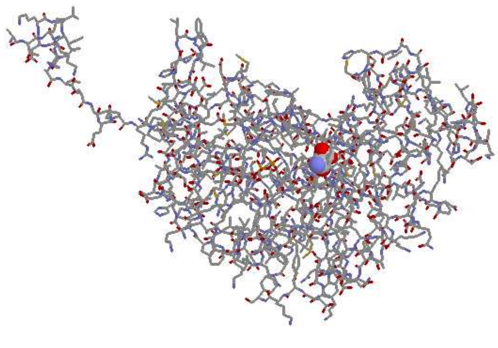


Fig. 18. A protein in wireframe with a space-fill ligand. PDB file: 1LDM

this case, a configuration is a conformation of the ligand in a particular position and orientation. As we are going to see in the experiments, our representation of ligands and proteins was sufficient to generate candidate binding sites near the real binding sites. This may be explained by the fact that for many protein-ligand complexes, the protein molecules do not make large conformational changes when interacting with the ligand atoms.

In this work, we apply the Iterative Relaxation of Constraints (IRC) technique (see Chapter III to protein-ligand binding. We found some initial configurations either automatically or manually. These initial configurations are approximate in the sense that they may have higher energy values than are acceptable. Later, we improve these configurations by moving them to nearby local minima. Our automated configuration generation methods are inspired by PRMs. As discussed below, the applicability of PRMs to protein-ligand binding was first noted by Singh *et al.* [112]. The configuration of the ligand in the binding site has low potential energy, and so the usual PRM feasibility test (collision) is replaced by a test for low potential energy. In this study, we use OBPRM [6] to generate configurations close to the protein's surface. Since our goal, the binding site, is a local minimum of the potential, we use a gradient-descent like algorithm to drive the generated configurations towards a local minimum (improving the approximate solution). Our approach was able to quickly (usually 15-60 seconds) generate conformations in the binding site in all of the 15 protein-ligand complexes tested. Two of these complexes were studied by Singh *et al.* [112], where in one case their approach failed to generate a conformation in the binding site, and twelve of the complexes were studied with FlexX [105], which obtained good predictions in all cases.

---

[2] This research is published in [23, 24].

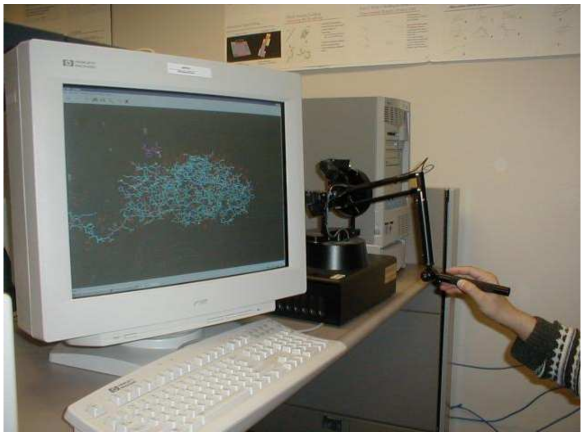


Fig. 19. Using the PHANToM haptic device for in ligand binding.

We note that our primary goal in this work is to study the potential of our IRC technique for generating conformations in the binding site. Ideally, these conformations would then be analyzed by one of the many existing techniques for scoring binding conformations (e.g., [41, 51, 74, 84, 95, 96, 126]). In this work we use a very simple potential energy function to study the feasibility of our approach. Ultimately, the system should utilize one of the available energy functions such as CHARMM [30].

The *narrow passage* problem, which refers to the difficulty of sampling important configurations in a narrow C-space region, and is the motivation for our IRC paradigm, also exists in ligand binding where the narrow passage becomes a passage through high potential areas. Similar to virtual prototyping in Chapter IV, we also investigate if we can further improve our result by incorporating human insights to the system through a haptic device. To address this problem, we add a human operator equipped with a haptic device to the system. As in virtual prototyping, a haptic device enables the user to touch and feel a virtual object (see Figure 19). The user adds human insight and the haptic device enables the user to explore the energy space. In our approach, the user selects some configurations of the ligand with the haptic device and they are passed to the planner for further processing.

Another challenge with PRMs is that it is hard for the user to visualize and understand the progress made by the planner. In molecular docking, this problem becomes even more significant since the familiar three-dimensional workspace is replaced with an energy landscape. Since feeling the potential energy surface is not sufficient for understanding the planner's progress, roadmap visualization methods similar to we have seen in Chapter IV are also employed.

Although there has been much work on automated solutions of the molecular docking problem, they have mainly concentrated on the fully automated solution of a simplified version of the problem (e.g., they often assume rigid ligands). There has been less research on man-machine interaction for this problem and there has been even less work using novel technologies like haptic interfaces.

Outline. We discuss relevant previous work in Section A. We briefly describe C-Space for the ligands and our potential energy computations in Section B Section C describes our haptic interaction system for molecular docking. We give details of our approach in Sections 1 and E, and present experimental results in Section F. Section G concludes this chapter.

Throughout this chapter we use the terms conformation and configuration interchangeably.

A. Related Work

Many automated docking algorithms have been proposed, including, for example, AutoDock [95], Dock [74, 40], FlexX [105], FLOG [92], FTDock [67], and Gold [65]. While many successes have been achieved by such algorithms, it has been stated that the success ratio is often relatively low (2-20%) [126]. Also, in most cases these algorithms make the simplifying assumption that the ligand is rigid to reduce the computation cost.

Recently, a new approach which makes use of the PRM paradigm for protein-ligand binding was proposed by Singh *et al.* [112]. Their results were very promising, and are in large part the motivation for this work. They uniformly sample configurations, and resample more densely around a small subset of low potential configurations from the original sample. In essence, they identify randomly sampled configurations with low potential energies and further explore those regions. Since the goal of the obstacle-based PRM used in our work is to generate, directly, configurations near the protein's surface, we believe that it will provide a more representative sample of the configuration on the surface than the uniform sampling approach of [112]. As will be seen in Section F, some evidence in favor of our approach is our success in generating configurations in the binding site for a test case in which this previous method failed.

Because of the relatively low success rates for fully automated algorithms, some researchers have added a human operator to the decision process. For example, Brickmann *et al.* [28] discuss new strategies for man-machine interaction in molecular modeling. STALK [81] enables the user to interact with the genetic algorithms in a virtual reality environment. An operator can move the ligand with a mouse in SCULPT [123, 124]. In SMD [80], biochemists can "tag" molecules into different shapes by specifying external forces on the graphical interface. A revolutionary approach taken by Ouh-young [100] and Ouh-young *et al.* [101] was to include a haptic interface and let the user feel the force applied on the ligand during the docking process. In Ouh-young *et al.* [102] it is shown that force feedback performed better than visual display in the docking process. In Pachter and Lupo [103], an interactive molecular docking simulator was used for interactive molecular design. It is shown by Wanger [132] that SCULPT can be improved by adding a haptic device. Meyer *et* *al.* [90] discusses using a haptic device in molecular modeling and drug design. Finally, Harvey and Gingold [53] haptically visualize the probability density functions of individual atomic orbitals.

Unfortunately, there has not been much work on improving automated planners with human input. In [21], and, in the previous chapter, we used a haptic device together with a PRM planner on motion planning problems typical of maintainability studies in mechanical CAD designs. Our results showed that human insight improved the planner's ability to solve problems involving narrow passages in C-space.

B. Preliminaries

Our motion planning system for ligand binding is very similar to that used for articulated bodies. In this section, we briefly describe a configuration space for the ligands. As described in Chapter II, the feasibility criterion for ligands is potential energy. We describe the potential energy we have used in more detail in the last part of this section.

1. C-spaces of Ligands

Proteins usually consist of thousands of atoms, while the ligand typically consists of tens. While both protein and ligand are in fact flexible, we model the protein as a rigid body and the ligand as a flexible body as was done in [112]. In particular, the ligand is modeled as an articulated body with a free base, where the torsional movement of atomic bonds is modeled by one degree of freedom (dof) revolute joints (bond angles and lengths are fixed). Thus, the base atom requires 6 dof and each additional torsional movement requires 1 dof.

The joint angle of a revolute joint takes on values in $[0, 2\pi)$, with the angle 2π

equated to 0, which is naturally associated with a unit circle in the plane, denoted by S^1. Assuming some position and orientation for one of the links (the base), the positions of each of the remaining links can be specified by the *joint angle* between the link and some adjacent link. Thus, a *configuration* of an n joint tree-like articulated object can be specified by 6 dof for the base atom, and $n - 1$ joint angles for the remaining atoms. That is, the configuration space of interest for our ligands can be expressed as:

$$\mathcal{C} = \{q \mid q \in R^3 \times S^3 \times S^{n-1}\}. \tag{5.1}$$

Note that $\mathcal{C}$ simply denotes the set of all possible configurations, but says nothing about their feasibility. The validity of a point in $\mathcal{C}$ will be determined by potential energy computations.

An example configuration for the ligand *Napap (MID)* is shown in Figure 20. In the figure, the nitrogen (N), carbon (C), oxygen (O), and sulfur (S) atoms are shown in different shades and textures. There is a link between two atoms if only the rotation between two atoms changes the shape of the ligand. For example, if the angle between C1 and C2 is changed, the upper (or bottom) part of the ligand will rotate and the shape will change. So there has to be a link between C1 and C2. However, if the angle between S and O changes, only O will rotate around itself and the overall shape will be the same, so there is no link between S and O. Also if two atoms are part of a ring such as C3 and C4 there are no links between them, because they cannot actually rotate since the connections to their neighbors would prevent that, i.e., we treat rings as rigid. The links of MID are shown with dashed lines in Figure 20. There are 11 links for the torsional angles and 6 for the base orientation, so MID has 17 dof.

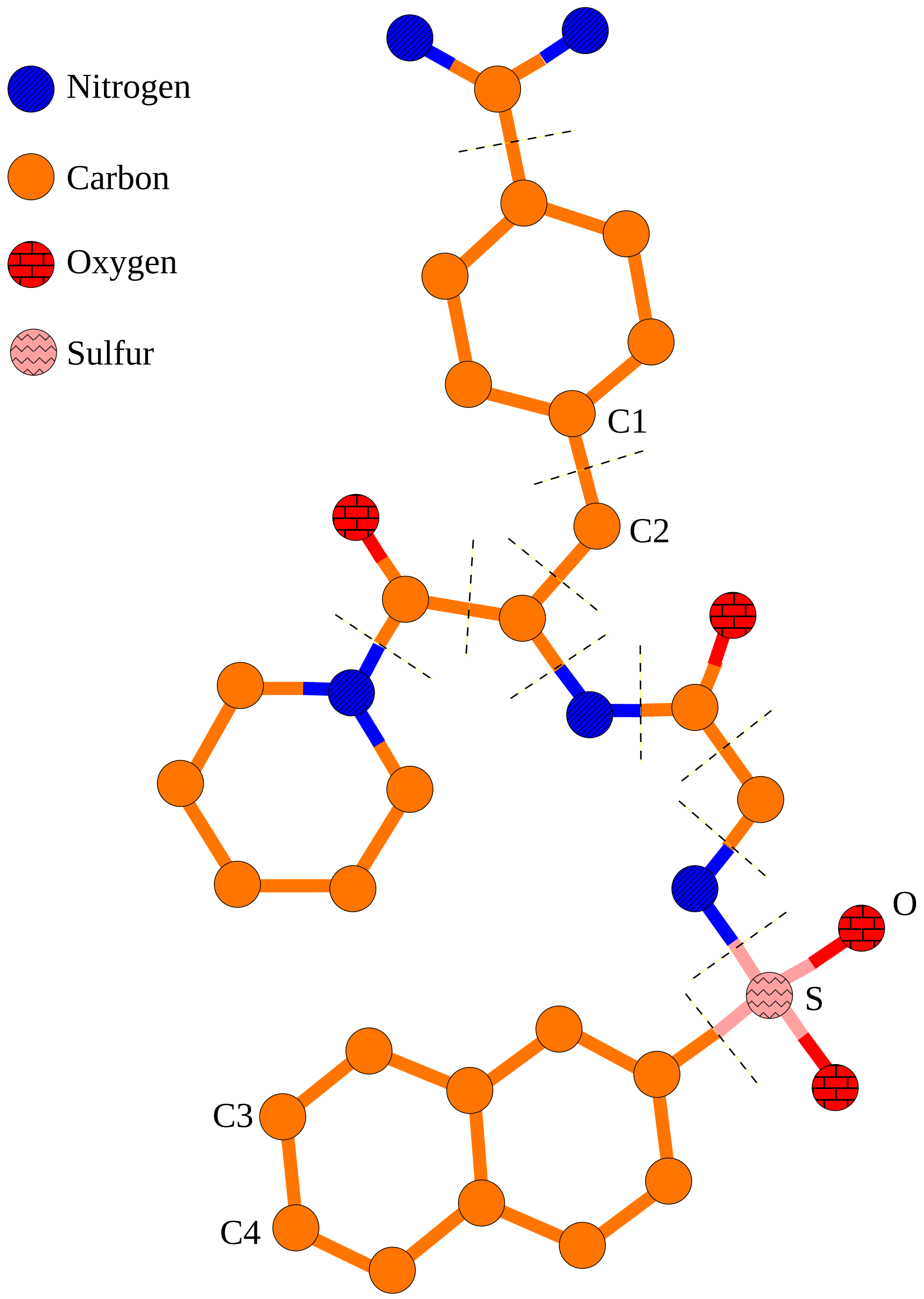


Fig. 20. Configuration of Napap (MID). The dashed lines show where the torsional links are located.

2. Potential Energy

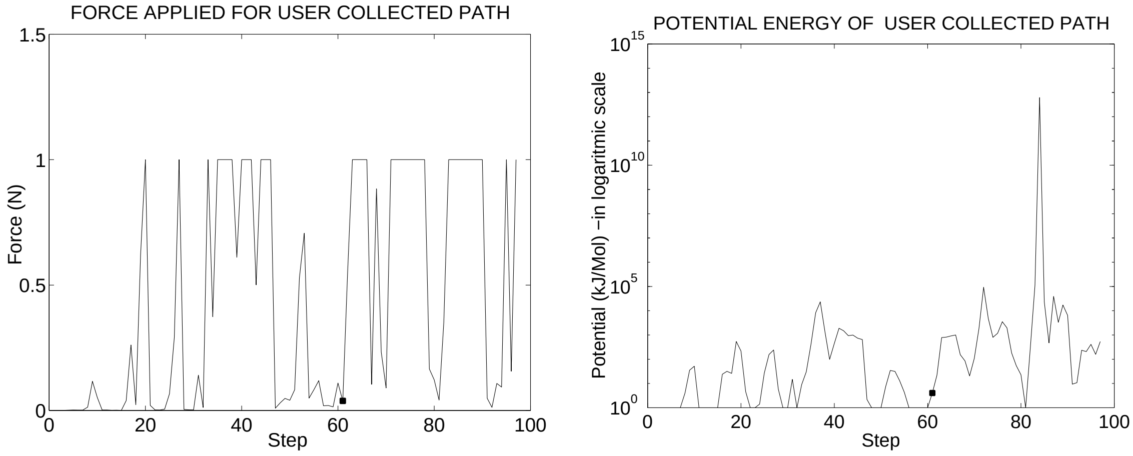
A key requirement in simulating the ligand binding process is to have a good model of the interaction between a ligand and its receptor, a protein. During binding, the ligand can be viewed as moving in the potential field created by the protein's atoms, searching for a stable low potential configuration. The potential has many components such as van der Waals potential, Coulomb potential, etc. Things are further complicated by the solvent effect due to the surrounding fluid (mainly water), which is difficult to model efficiently. Thus, a complete, accurate model of the potential is not feasible. Instead, we approximate the potential by the van der Waals terms only. While this is an extreme simplification, the results obtained are quite impressive indicating that perhaps the van der Waals interaction plays the main role in the binding process. In particular, the potential we use is

$$U = \sum_{atom\ pair\ i,j} (A/r_{ij}^{12} - B/r_{ij}^{6}) \tag{5.2}$$

where r_{ij} is the distance between protein atom i and ligand atom j, and the parameters are taken from [82].

Even with our simplified potential, computing the potential energy of a ligand configuration is expensive since it includes a contribution for each atom pair. In particular, the time complexity for a potential energy computation is $O(mn)$, where m and n are the number of atoms in the protein and ligand, respectively. Recall that m and n are on the order of 1000 and 10, respectively.

There are a few approaches to optimize this computation. One is parallelization, which can reduce the complexity to $O(mn/p)$, where p is the number of processors, as the potential computation is embarrassingly parallel since the contribution from each atom-pair can be computed independently.



(a) (b)

Fig. 21. Correlation between (a) user-felt force and (b) potential energy. The binding conformation is shown as a solid square.

Another approach is to use a grid-based energy calculation. We use this approach to optimize the potential energy computation. In this approach, the protein is covered by a three-dimensional grid, and each grid point stores the potential applied by all protein atoms at that point [104]. During potential energy calculations, the value stored at the closest grid point to each ligand atom is used as an approximation of the potential for that atom. The accuracy of this method directly depends on the resolution of the grid. Better accuracy can be achieved by using smaller resolutions. However, the smaller grid size, the more it costs to compute and store it. Our experiments indicate that a 0.5 Å grid is sufficient for realistic force feedback for our haptic device.

C. Overview

Our prototype system consists of a PHANToM haptic device (6 dof) output) [89], and a Xeon 550Mhz NT Workstation. The operator uses the PHANToM to manipulate a *rigid* ligand around a virtual protein.

A challenge for molecular docking is to compute the potential energies of the configurations fast enough for haptic interaction. To achieve this, we used a grid-based force calculation as described above, which was also used in [100]. Figure 21(a) shows the force felt by the user while collecting a path of ligand configurations. To stay in the PHANToM's force feedback limits, any computed force greater than 1N was reduced to 1N without changing its direction. This resulted in a repulsive force in the correct direction. Figure 21(b) shows the real potential energy at the sampled configurations (y-axis is logarithmic). The correlation between the figures shows that the user feels low repulsive forces when the real potential energy is low. Note that at the binding configuration, the potential energy is low and the user felt almost no force, hence with visual and haptic feedback the user can estimate the position of a binding site. A notable observation is that there are some low potential configurations near the binding configuration which actually represent a corridor that the ligand must pass through to reach the binding site. Most other configurations with small potentials were not near the protein's surface.

Our haptic system also lets the user investigate a path or roadmap generated by the automated planner. The user can visualize a path, select a configuration in the path, and investigate the potential field around that configuration. Similarly, the user can visualize a roadmap. Roadmap configurations are represented by points in the three-dimensional workspace and roadmap edges by lines connecting them [22]. The user can investigate a specific roadmap configuration by moving the haptic pointer close to a roadmap configuration which causes the configuration to pop up visually. To aid location of roadmap configurations, there is an option which pulls the user to the nearest roadmap configuration.

D. Approximate Solution Generation

Good binding conformations are thought to be conformations at local minima in the potential landscape. Additionally, they may be more stable if they are surrounded by conformations with higher potentials so that the ligand will tend to stay in the site once it gets there [112].

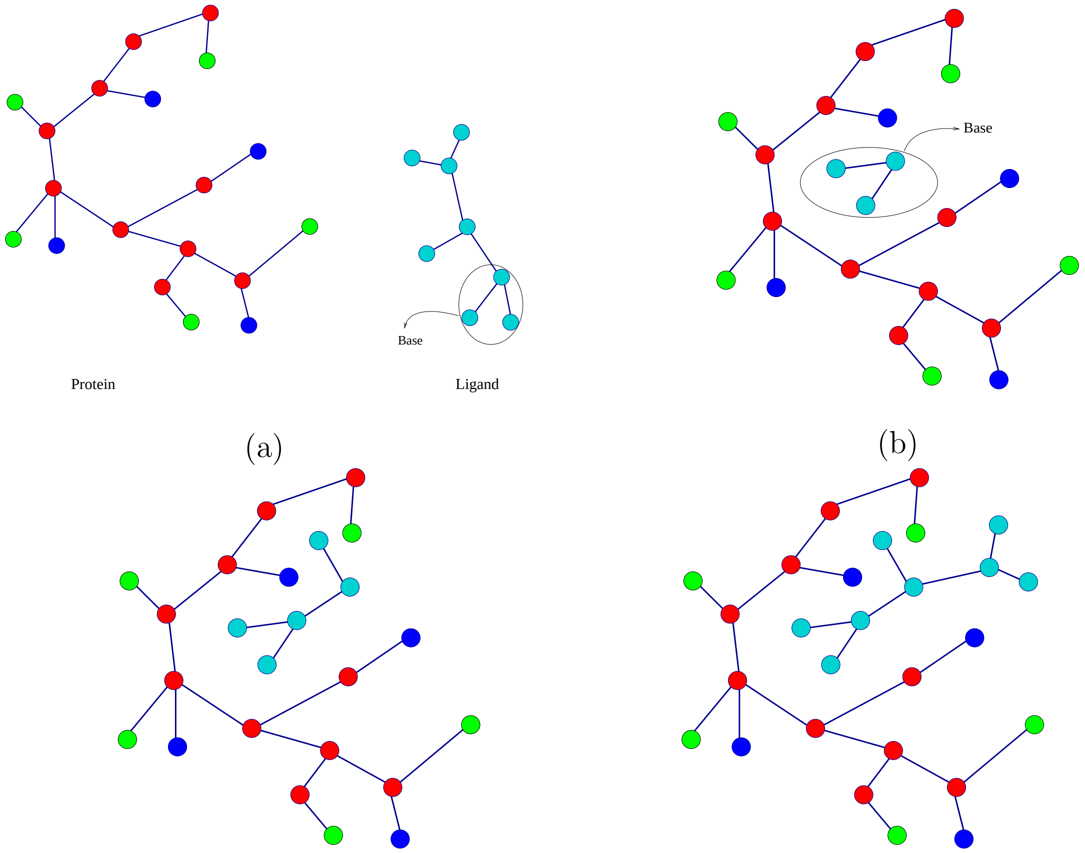
Our goal is to automatically generate such conformations of the ligand. Within our *IRC* paradigm, we first find an approximate solution by generating nodes near the protein's surface, and then improve that solution by pushing the generated nodes to their local minima. We have investigated several methods for generating the initial ligand conformations:

- an OBPRM-like method utilizing geometrical information only,
- an OBPRM-like method utilizing potential energy information, and
- a haptics-based user input method.

Details of each step are discussed below.

1. Automatically Generated Conformations Using Geometrical Information

Although the initial conformation could be selected randomly, the immense size of the search space means it is difficult to find conformations close to the binding site, or even the protein's surface, in this manner. Our OBPRM, obstacle-based PRM, is designed to exploit known geometrical information to generate conformations near the surface. Figure 22 shows how this works. We model each protein atom as a unit cube (1 Å), and each ligand atom as a sphere with 1 Å radius. The reason a cubic representation is selected for protein atoms is to improve the efficiency of OBPRM by using the



(a) (b)

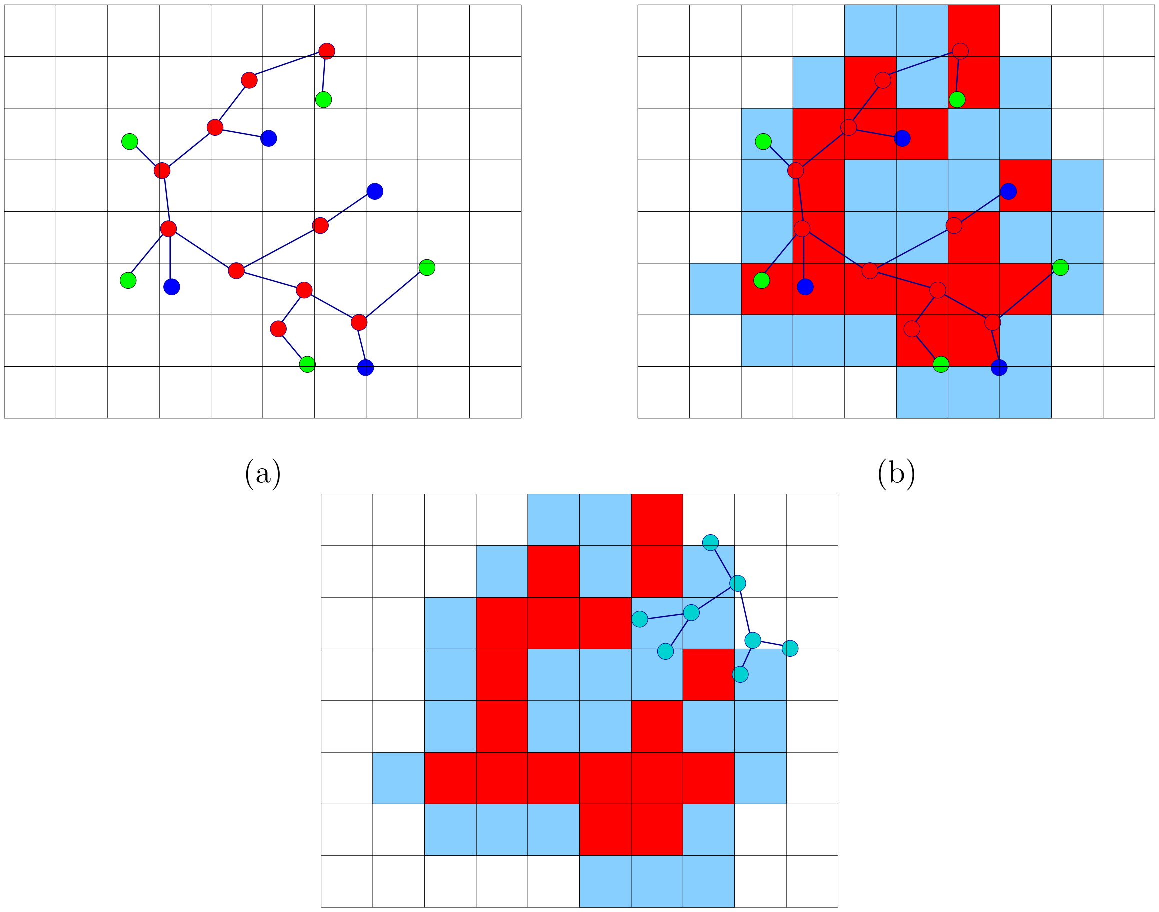
(c) (d)

Fig. 22. Automatically generating conformations using geometrical information. (a) A Protein and a ligand. (b) The ligand's base is put a collision free position and orientation close to protein atoms. (c), (d) The remaining links are generated randomly while satisfying collision free constrains.

surface normals of the cubes to direct our configuration generation. Considering only the base atoms of the articulated ligand model, we generate a collision-free (geometrically) node for the base (Figure 22(b)). The link which would be the base can be arbitrarily selected among all the ligand's links, since it won't have a significant effect due to the random placement of the other links. We have selected the atoms of the first links in the PDB file as the base. Then, we assign random values for the joint angles (Figures 22(c) and 22(d)). Next, we evaluate the potential of the ligand's configuration and keep the node only if its potential is less than some fixed value $E_{\max}$. A fairly high value of 5,000 KJ/Mol was chosen for $E_{\max}$. However, when we improve the solution, most remaining nodes will have low potential. Although better than uniform sampling in the conformation space, the effectiveness of this approach is limited since its initial placement of the base of the ligand is guided purely by geometrical constraints and does not use information regarding the potential energy landscape.

2. Automatically Generated Conformations Using Potential Energy Information

Our next method utilizes the already calculated potential grid (Section B.2) to concentrate the generation of conformations in low energy regions (See Figure 23). Recall that we use a 0.5Å grid around the protein in which each grid cell stores the potential applied by all protein atoms at that point. In our implementation, we first choose a low potential cell at random from all low potential cells of the grid. We then translate the base of the ligand to that cell (i.e., we choose a position within the cell) and select a random orientation. Then, we check if the atoms of the base are placed in acceptable (i.e., low-energy) cells. If not, we repeat this procedure until we find an acceptable configuration for base. Then, we choose a random angle (orientation)



(a)

(b)

(c)

Fig. 23. Automatically generating conformations using potential energy information. (a) Grid is calculated. (b) High (darkest area), low (dark area) and very low energy (white area) regions are identified. (c) The ligand is placed in low energy regions.

for the next link and check if the atoms of the link are placed in acceptable cells. This procedure is repeated until a satisfactory configuration is found for the link. If no satisfactory configuration for the link is found after some maximum number of attempts, we move one step back and generate another configuration for the previous link. This process is repeated until the entire ligand is processed.

Ideally, all ligand atoms will be placed in low potential energy cells. However, this is not always possible. So, we allow some of the atoms to be placed in high(er) potential cells provided that these cells are adjacent to low potential neighbors. This actually enables the generated configurations to lie near potential energy boundaries. We also favor the low energy cells with high energy neighbors (since they are also near the boundary).

Note that this approach is similar to OBPRM since the nodes that are generated will be close to constraint surfaces in the potential landscape. Also, it implicitly utilizes geometrical information, since the geometry determines the potentials.

3. Haptics-based User Generated Conformations

We also experimented with operator collected nodes using a visual and haptic interface. As we have seen in the previous chapter, significant improvement can be achieved in CAD/CAM models when user input is given to the automatic planner. A user would collect a node when he/she felt like it was a local minimum in the potential and also saw it was in a promising position on the protein. The ligand was restricted to a rigid body for the haptic interaction.

E. Improving the Approximate Solution

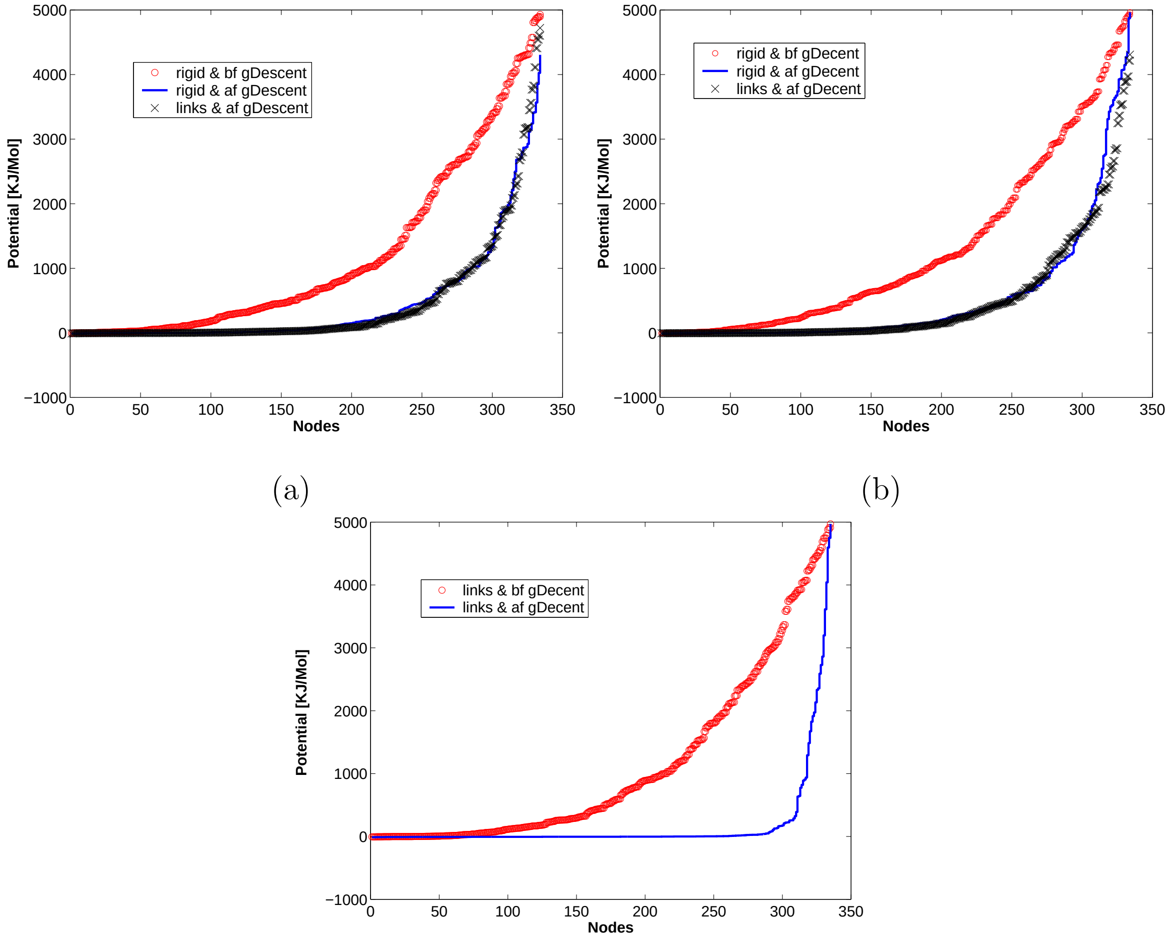
Since a binding configuration is commonly thought to be a local minimum surrounded by higher potentials, it is natural to push our generated nodes to local minimum. In particular, we perform an approximate gradient decent as described below. Next, the generated configurations will be connected to form the roadmap.

1. Approximate Gradient Decent

For each generated node, we uniformly sample n nearby nodes, and select the lowest potential new node for iteration. We chose $n = 2d$, where d is the number of degrees of freedom for the ligand. This process is repeated for K iterations, or until a local minimum is reached (when all sampled nodes have higher potential). A local minimum was usually obtained in 2-4 iterations. Figure 24 shows the energy values before and after gradient descent for some ligand-protein complexes (with rigid and flexible ligand representations).

2. Node Connection

After gradient decent is performed, all local minimum nodes are added to the roadmap (with the previously generated nodes). Then, we perform node connection as is normally done for PRMs, i.e., we attempt to connect each node to its k nearest neighbors along the straight-line in C-space between them (we use $k = 10$). An edge between the node and its neighbor is added to the roadmap if it passes some validity check. The only difference here is that our validity check uses potential evaluation instead of collision detection. As we will see later, we use this roadmap to identify the most promising candidates.



(a)

(b)

(c)

Fig. 24. The (sorted) potential distribution of all roadmap nodes before and after gradient descent for three protein-ligand complexes (given by pdb identifier): (a) 1LDM, (b) 1A5Z and (c) 1STP.

Table IV
Statistics for the Different Protein-Ligand Complexes Studied

Pair ID	***Protein***		***Ligand***			
Protein Sym.	**Name**	**# Atom**	**Name**	**Sym.**	**#Atom**	**dof**
5TIM	Triosepho. Isomer.	4063	1,4-Dithiothreitol	DTT	8	6
1ULB	Pur. Nucle. Phospho.	2279	Guanine	GUN	11	6
3TPI	Trypsinogen	2262	Sulfate Ion	SO4	5	6
1A5Z	*L*-Lact. Dehydroge.	2416	Oxamate	OXM	6	7
1LDM	M_4-Lact. Dehydroge.	2544	Oxamate	OXM	6	7
2PHH	PHBH-ADPR-POHB	3201	P-Hydroxybenz.	PHB	10	7
3PTB	β-Trypsin	1701	Benzamidine	BEN	9	7
6RSA	Ribonuclease A	1535	Uridine Vanadate	UVC	21	8
2CTC	Carboxypeptidase	2636	L-Phenyl lactate	LOF	12	9
4TLN	Thermolysin	2597	L-Leu.-Hydroxyl.	LNO	10	10
1STP	Streptavidin	1001	Biotin	BTN	16	11
4DFR	Dihydrofo. Reduct.	3005	Methotrexate	MTX	33	16
1DWC	α-Thrombin	2437	MD-805	MIT	35	17
1DWD	α-Thrombin	2445	Napap	MID	37	17
4PHV	Hiv-1 Protease	1716	N/A	VAC	36	20

F. Experiments

We tested our approach on 15 different protein-ligand complexes obtained from the Protein Data Bank at *http://www.rcsb.org/pdb/*, which provides atom positions for the protein and the binding configuration of the ligand. Two of these complexes, 1LDM and 1STP, were studied in [112]. 1A5Z, was chosen based on the complexity of protein's potential field which created many isolated areas. The remaining complexes were selected from those studied in [105]. Some details of these complexes are shown in Table IV.

In our experiments, we investigated the following questions:

- How does the purely geometry-based generation method compare with the grid-based potential energy generation method?
- How does the rigid approximation of a flexible ligand affect our ability to generate conformations in the binding site?
- Can a user provide helpful information to an automated planner? Does the haptic interface aid the user's understanding?

To answer these questions we performed three different types of experiments.

In the first experiment, we investigated the behavior of the automated planner when the ligand was flexible, i.e., articulated. In this case we used both automatic generation methods, i.e., using geometrical information or using the grid-based potential energy information.

In the second experiment, the ligand was treated as a rigid body. The planner generated configurations using geometrical information, and found potential binding sites using these configurations.

In the third experiment, the user collected configurations using the PHANToM by manipulating a rigid ligand. The automated planner then fine-tuned these configurations using approximate gradient descent, now treating them as flexible molecules.

1. Scoring Binding Sites

There exist as many scoring functions as docking algorithms, including empirical functions [41, 95, 74], knowledge based functions [96], and force field functions [84, 51]. Our goal is to generate good nodes that can be scored by these existing techniques. Nonetheless, as it may be useful to help characterize the conformations we find, we have implemented a very basic scoring method. However, as with our potential energy function, we emphasize that our scores should not be given much significance, as one should ultimately utilize previously existing techniques.

Our basic scoring method can be described as follows. We first uniformly sample n nodes within a fixed distance r from the candidate node, and then define the weight, or score, of the candidate as the average potential of all the sampled nodes.

$$Weight(c) = \frac{1}{n}\sum_{i=0}^{n} P(c_i) \tag{5.3}$$

where c_i is a random configuration sampled near c. The potential, $P(c_i)$, is truncated

Table V

Results for obprm-like Generation Methods for Flexible Ligands

		Energy-Based Generation			Geometry-Based Generation	
Complex	dof	*RMSD*	*Time (min)*	*Rank(%)*	*RMSD*	*Time(min)*
5TIM	6	6.0 (6.7)	0.48	96	1.5 (2.7)	0.68
1ULB	6	0.8 (2.1)	0.31	1	0.8 (3.7)	0.43
3TPI	6	1.1 (1.4)	0.18	18	0.5 (2.1)	0.42
1A5Z	7	3.3 (3.9)	0.32	1	0.8 (1.4)	0.44
1LDM	7	0.7 (1.9)	0.32	11	1.0 (2.6)	0.29
2PHH	7	2.3 (3.5)	0.59	49	1.4 (3.1)	1.3
3PTB	7	1.8 (2.5)	0.51	48	1.4 (2.9)	1.2
6RSA	8	1.7 (3.3)	9.20	51	1.6 (5.9)	1.8
2CTC	9	1.9 (4.6)	1.70	69	1.2 (4.5)	2.2
4TLN	10	2.2 (3.5)	1.10	30	1.4 (1.7)	1.6
1STP	11	2.1 (3.2)	2.34	1	1.6 (2.7)	1.2
4DFR	16	2.8 (5.9)	10.75	20	4.6 (5.4)	5.6
1DWC	17	4.4 (5.6)	4.00	54	3.7 (5.6)	1.9
1DWD	17	4.7 (7.8)	7.90	88	4.1 (5.9)	8.1
4PHV	20	3.5 (7.5)	4.80	10	5.0 (5.7)	37

Results for obprm-like generation methods for flexible ligands. For each complex, we report the generated conformation that was closest to the binding site in terms of RMSD distance, where the RMSD was obtained after applying the best rotation to the ligand conformation. In the RMSD column, the smaller value was computed using rotation, and the one in parenthesis is the RMSD without any rotation. The rank column indicates the percentage among all generated conformations, a low number implies a high (good) rank.

at 5,000 KJ/Mol if it has a larger value.

2. Results

All experiments were performed on a Pentium III 550 MHz PC. A summary of our results for the 15 complexes is shown in Tables V and VI. In these tables, the most significant values reported are (i) the RMSD distances between the generated conformations and (ii) the binding conformation and the computation times. While we also report ranks, they should be given less importance since we have used very simplistic potential and scoring functions. In an effort to somewhat compensate for our simplistic potential, we applied the same gradient descent operation to the binding conformation (taken from the pdb file) as was applied to all other conformations before computing the RMSD distances.

Table VI

Comparison of Different Node Generation Strategies

Different Node Generation Strategies									
Complex		Geometry-Based			Energy-Based		Haptic-Based User Input		
Protein	Ligand (dof)	Rigid	Flex.	Time[min]	Flex.	Time[min]	Rigid/Flex.	Time[min]	#Cfgs
5TIM	DTT(6)	1.4	N/A	0.68	6.0	0.48	4.45	0.20	54
1A5Z	OXM(7)	4.1	0.8	0.44	3.3	0.32	1.94	0.32	129
1LDM	OXM(7)	1.5	1.0	0.29	0.7	0.32	1.52	0.01	37
1STP	BTN(11)	4.2	1.6	1.2	2.1	2.34	2.56	0.47	92
1DWC	MIT(17)	N/A	3.7	1.9	4.4	4.00	5.66	0.14	178

Comparison of different node generation strategies: geometry-based and energy-based OBPRM-like methods, and haptics-based user input. The values shown in the Rigid/Flex columns are RMSD from the binding conformation. The geometry-based method was tested using both flexible and rigid approximations of the ligand. The energy-based method used only flexible ligands, while the haptics-based method used rigid approximations for the user collection, but a flexible model for the gradient descent. The #Cfgs value denotes the number of conformations collected by the user.

As can be seen in Table V, both the geometry-based and the energy-based methods perform well. They are both able to generate nodes close to the binding site, within 1-2 Å in most cases. Moreover, they are both quite efficient, in many cases requiring only 1-2 minutes. And more importantly, these results were obtained without utilizing any prior knowledge of the protein, which makes our method applicable to any protein/ligand complex, including previously studied and newly discovered structures. In general, the geometry-based method generates conformations that are closer to the binding conformation, while the energy-based method is a bit faster. Indeed, it is interesting to note how well a method which relies solely on geometrical information for the initial placement of the ligand's base does in generating conformations in the binding site.

The performance of both methods decreases as the dof of the ligand increases. Since the geometry-based method works by fitting the base 'geometrically' to the protein and then randomly generating joint angles for the remaining links, its performance degenerates since randomly finding a conformation fitting the physical space becomes harder as the dof increases. However, we would not expect the energy-based method to be as sensitive to the dof since it works by laying the ligand down, link by link, and can shape the ligand according to the energy surface (as determined by the grid-based potential). This is most noticeable for the 20 dof 4PHV.

A few remarks are in order regarding the consequences of our simplistic choices for the potential function and the scoring method. First, although the conformations in all methods are pushed to a local minimum as determined by the potential function, we expect that the performance of the energy-based method is most affected by inaccuracies in the energy function. We believe this is at least part of the explanation for the slight edge the geometry-based method appears to have over the energy-based method. In fact, we believe that the energy-based method would likely out perform the geometry-based method, especially for high dof ligands, if the grid were constructed using a better potential energy function.

Another factor to bear in mind is that our scoring is affected by the potential function. This is part of the reason why in some cases conformations with small RMSD to the binding conformation have a very poor rank. In addition, though, our scoring method is very naive and even with a more accurate potential would not be expected to rank all close conformations well. Nonetheless, we believe the main contribution of this work is to show that the IRC method can be successful in quickly generating conformations in the binding site, even given a rather naive potential function. We would expect improved performance with a more accurate potential.

a. Rigid vs. Flexible Ligand Models

When we compare the automated results for the rigid body and articulated representations in Table VI it is clear that better results were obtained using the flexible representations. In general, since there are many local minima in the protein's potential field, movement is hard for a rigid ligand. However, an articulated ligand can move in these areas by taking on the shape of the surrounding potential surface. Our comparison of the rigid and articulated representation is potentially relevant for other docking methods, since, as mentioned in Section A, representing the ligand as a rigid body is a commonly used simplification.

b. Haptic-based User Input

An advantage of user input is that, as shown in Table VI, it was noticeably faster than the OBPRM-based fully automated methods because the user provided a much smaller sample of nodes. (Here, however, we have not included the time the user spent collecting the nodes manually.) Fewer nodes were provided since the user only collected conformations in what appeared to be promising regions of the protein. A disadvantage is that the user input suffers the users' bias. It is difficult and tedious for the user to explore every corner of the protein, and therefore he/she may miss the true binding site. In addition, a user may be misled by some 'decoy' sites based on visual and/or force feedback. Both of these factors contributed to the relatively poor RMSDs of the user collected configurations as compared to the fully automated methods. However, haptic collection gives a unique experience to the user which the automated planners lack. This experience may help the user to better understand the problem and perhaps modify the solution parameters to obtain better results with the automated methods.

The results for 5TIM in Tables V and VI are quite interesting. Since the ligand has only 6 dof, we would expect this to be a relatively easy complex. However, both the user collected and the energy-based methods performed poorly whereas the geometry-based method found a conformation with an RMSD within 1.4 Å of the binding conformation. This can be explained by the fact that the binding site of this complex is in the outer region. Since we sample from the boundary cells in the energy-based method, most of the configurations generated were in more central parts of the protein. Similarly, the user is biased towards the more crowded areas. However, the geometry-based method samples more uniformly around the protein's surface and so is more likely to find the binding site.

G. Conclusion

Our results show that our approach to molecular docking is promising. In the examples we studied, we were able to find conformations in the binding site. However, further processing, perhaps with a more accurate potential function, will be needed to find an exact binding configuration. For example, candidates identified by our method might be used as input for other docking programs that perform detailed simulations, such as molecular dynamics methods. We also saw that better results were obtained using an articulated representation for the ligand, as opposed to the commonly used rigid body simplification. User input was seen to improve efficiency, and moreover, haptic feedback was observed to help the user better understand the problem.

Our future work includes incorporating existing potential energy functions and scoring techniques, implementing a more flexible protein representation than the rigid proteins we have now, and improving the haptic interface by allowing the user to collect configurations for a flexible ligand.

CHAPTER VI

DEFORMABLE OBJECTS

As we have seen so far, it is common practice to represent a robot as a set of rigid objects connected by joints. Although this representation covers many common situations, such as most industrial robots, there are many cases where a more flexible representation of the robot would be desirable.

Any problem that requires a robot to change its surface or shape is almost impossible for most existing planners. Yet such problems arise naturally in many common situations. Moving a long rod through corridors may require the rod to bend when passing around corners, or a flexible sack being carried or pulled by a robot may require the sack to deform when passing through narrow openings or passages. There are many situations in which the other objects in the environment are most appropriately modeled as deformable objects. For example, in a surgical simulation, the environment contains rigid bones, somewhat deformable cartilage, and highly deformable tissue. Figure 25 shows an example deformation of a brick.

Deformable objects are commonly used in computer modeling and animation. Nevertheless, even for an expert, generating deformation animations by hand, especially for complicated models, is always a very difficult and time-consuming task. Moreover, natural-looking deformed models usually require significant additional time for the animator to manually adjust the control points and parameters until the deformation 'looks right.'

Representing the deformable object with a very large number of dof (in the limit, one for each deformable point on its surface) could work in principle for most prob-

---

[3] This research is published in [17].

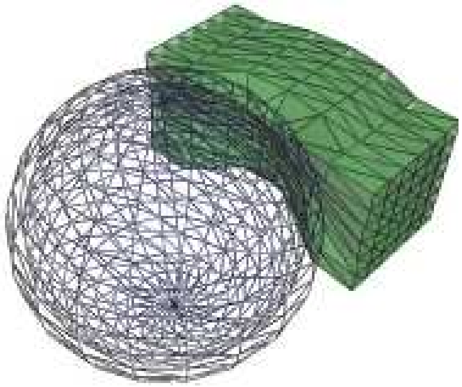


Fig. 25. A brick deformed by a spherical obstacle.

lems. However, this approach has the obvious disadvantage of increasing the dof to intractable levels and it still does not provide for a convenient way of enforcing physically realistic behavior. Thus, finding an acceptable deformation within reasonable time limits is the objective of most algorithms. The acceptability criteria can be defined by constraints on the topology or physical properties which define the realism of the deformation. Unfortunately, physically-correct deformation and speed are usually mutually exclusive. While a physically correct deformation is desirable, it might require several hours to produce. On the other hand, deformations which do not respect physical properties might be quickly computable, but generally produce unrealistic results. Thus, a trade–off between the time required to find a path with a realistic deformation and the degree of realism required must be made.

Our goal is to find an acceptable deformation within a reasonable time limit (real-time in some cases). Our approach is based on the Iterative Relaxation of Constraints (IRC) paradigm. We concentrate on problems where the robot is deformable and the other objects are rigid. However, our method can easily be generalized to problems where everything is deformable.

As in previous chapters, we first find an approximate solution and then improve that solution. In this case, the approximate solution is a path in which the robot might collide with the obstacles. Such paths can be found by using a reduced-scale version of the robot during the initial motion planning or by allowing the robot to penetrate inside an obstacle to some degree. As a result, the initial path may contain collisions for the rigid version of the robot. After finding such a path, we identify the colliding portions and deform the robot in these regions to avoid collisions. This is the improving the approximate solution stage. Note that although this approach is similar to that used for the virtual prototyping applications in Chapter IV, in virtual prototyping the planner deformed the path but not the robot, which is the opposite of the approach taken here. We currently employ two different methods for producing the deformations. One is a hierarchical approach which first deforms the robot's bounding box, and then the robot itself based on the the deformation of the bounding box. The second is a more direct approach which directly deforms the robot's geometry (but not topology).

This chapter is outlined as follows. We first discuss related work (Section A). Section B gives an overview of our approach. Section C describes how we find 'approximate' paths for a deformable object, and we discuss our deformation methods in Section D. Our experimental results are presented in Section E and Section F concludes this chapter.

A. Related Work

Deformable modeling and manipulation are essential in computer graphics. Deformable models are often used to design complicated objects. As in [47], these deformation methods can be classified into non-physical methods and those based on mechanic properties. Deformations which do not address physical properties include functional deformations (scale, bend, twist) [16] and free-form deformation [108]. Mass-spring and finite element methods are the most commonly used strategies for building deformable objects with physical properties. Deformation methods without constraints are very useful in interactive shape editing since they are fast and deform objects smoothly. However, physically–based deformations, which respond to applied external forces and constraints, are often required for simulation and animation, such as deformable tissue [45, 46], blood cells, creatures [91, 127], sand, mud, and snow [131], and cloth [56].

Although work on the motion planning problem has produced many practical planners for rigid robots, not as much work has been done in motion planning for deformable objects. The f-PRM framework [13, 68] has been proposed for planning for models using physically correct deformations. Due to expensive operations for solving mechanical models and generating collision detection data structures, their planner is not suitable for real-time use and can only deal with simple objects, such as a sheet of metal or a pipe-like robot. In [44], deformed distance fields were used to control the amount of penetration between non-penetrating flexible bodies.

B. Overview

As previously mentioned, our approach is based on the Iterative Relaxation of Constraints (IRC) paradigm.

Our implementation is similar to a traditional PRM. In our case we modify the feasibility requirements used during the roadmap construction stage so that our roadmaps may contain paths having some collisions. Later, during query processing, if a path containing collisions is extracted from the roadmap, we attempt to deform the robot to avoid those collisions. An example of such a deformation is shown in Figure 25, where the brick robot is deformed to avoid a collision with a spherical obstacle. If we do not succeed, we remove the corresponding edge from the roadmap and search for a new path.

MOTION PLANNING FOR DEFORMABLE OBJECTS

```
Build a feasible roadmap adhering to relaxed constraints
while (a valid path is not found)
    find a path in the roadmap
    foreach path configuration
        if (there is collision)
            deform robot
            if (not deformable)
                remove path segment from roadmap
endwhile
```

The way we have implemented this strategy is to use a rigid robot during roadmap construction, and a soft (deformable) robot during query processing. The key to our approach is to define appropriate feasibility metrics which enable one to construct useful roadmaps with rigid robots. The two feasibility tests we have studied thus far are:

1. In the sampled configuration, a reduced-scale version of the rigid robot (in some nominal shape configuration) is collision-free.
2. In the sampled configuration, an original-scale version of the rigid robot (in some nominal shape configuration) penetrates an obstacle only within some acceptable limit.

Clearly, both tests can accept colliding configurations of the original robot. During roadmap generation, we use heuristics to assign edge weights to denote the difficulty of deforming that edge, and during query processing a minimum weight path is extracted.

This philosophy is similar to Fuzzy PRM [98], Lazy PRM [47], or Customizable PRM [119] where the roadmap nodes and/or edges are not validated, or are only partially validated, during roadmap construction and are validated completely only at query time. These methods have been shown to greatly decrease the roadmap construction costs, while only slightly increasing the query costs. Moreover, as noted in [119], there are actually some other benefits to this approach, such as enabling the same roadmap to be used for different robots, or for queries with different requirements.

Outline. In the following sections we describe our algorithm in more detail. Section C discusses the methods to generate an initial approximate path. Our deformation methods are described in Section D. We present experimental results in Section E. Section F concludes this chapter.

C. Approximate Solution Generation

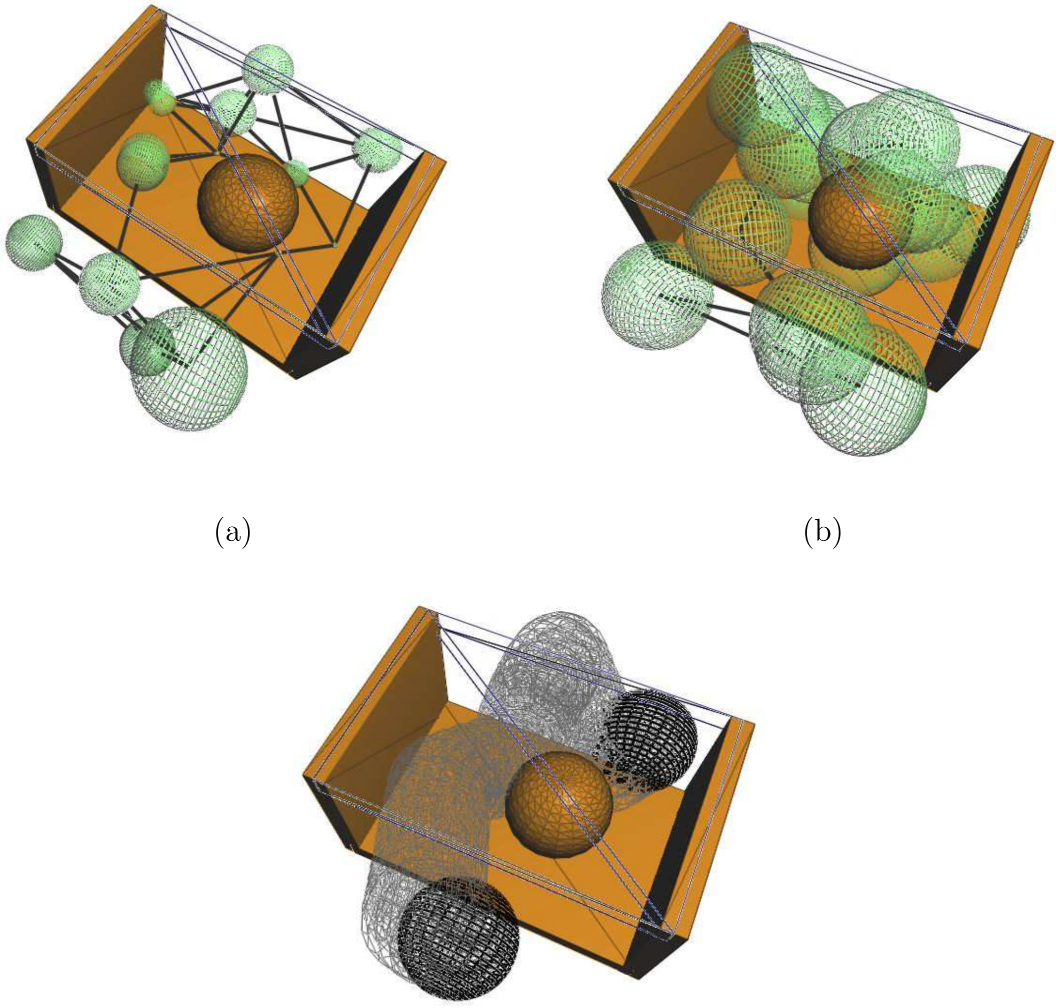
As discussed in Section B, our strategy is to build roadmaps which may include some colliding configurations. To increase the probability that we can in fact deform the colliding configurations to free configurations during the query phase, we place some feasibility requirements on the acceptable colliding configurations. We also attempt to weight our roadmap edges to denote the expected difficulty (cost and feasibility) of deforming that edge if necessary. That is, the weights are selected to denote the expected *deformation energy* required to deform the edge. Since this energy is not known *a priori*, we use heuristic methods of assigning the weights that are related to the parameters of the feasibility metrics used to define acceptable configurations.

In this section, we describe in detail the two strategies for constructing the roadmap mentioned above. In the first, roadmap nodes are accepted if the configuration corresponding to a reduced-scale version of the rigid robot is collision free. In the second, the configuration is accepted if the amount of penetration by the original robot is less than some acceptable bound. In both cases, our goal is to build a roadmap containing paths that are either already valid, or that can be made valid by deforming the robot.

1. Scaled Robots

We assume that the volume of the robot/obstacle intersection is related to the required deformation energy. We would like to weight roadmap edges with some estimation of these intersection volumes. While such volumes are difficult to compute, the ‘scale’ factor required to obtain a free robot in a particular configuration does offer some indication of the deformation energy and is at least loosely correlated with the intersection volume. Moreover, these scale factors can be computed relatively inexpensively if we have pre-computed a set of reduced-scale models, which can be viewed as a “uniformly deformed robot.” The scaled robot models are constructed in a straightforward manner by scaling all vertices defining the robot model, and maintaining the robot topology. These scaled rigid models are used for collision detection in node generation and connection.

Node Generation and Connection. During node generation, for each node, we begin with the largest robot and progressively reduce its scale, while maintaining the configuration, until a collision free robot size is found. After the roadmap nodes have been generated, the roadmap is connected as with a traditional PRM. In particular, any of the normal local planners or connection methods could be employed [6]. An edge weight is set as the sum of the scale factors of the two scaled models that define the (potential) edge’s endpoints. (A larger (smaller) robot model has a smaller (larger) scale factor.)



(a)

(b)

(c)

Fig. 26. Roadmaps for different generation methods and an example query. (a) A roadmap generated with scaled robots; roadmap configurations are the wire-frame spheres with a radius proportional to their scale factor. (b) A roadmap generated with penetrating robots; roadmap configurations are the wire-frame spheres. (c) A path found from a start to a goal configuration. Note that it is not the shortest path but one which is sensitive to the amount of required deformation.

Figure 26(a) shows a roadmap generated with a spherical scalable robot. The smaller (larger) roadmap nodes represent smaller (larger) robot models. One clearly sees the smaller nodes surrounding the obstacles, which should be avoided by the deformable object if possible.

2. Penetration

As discussed previously, one indication of ”deformability” is penetration depth. The deeper the robot penetrates inside an obstacle the harder it will be to deform the robot into a collision free shape. Unfortunately, penetration is hard to measure in most cases and is not provided by collision detection libraries (except for special cases such as convex objects [94]).

Hence, we propose using C-space penetration instead of workspace penetration. While C-space penetration is not easier to compute, we use a probabilistic approach to compute an estimate. Let c_0 be our initial configuration of interest. We generate a constant number of random C-space vectors which originate at c_0. These vectors have random directions and lengths in some predetermined range (our allowable penetration depth). Note that the size of a vector depends on the distance metric selected and can be defined as $|c_i - c_0|$ where c_i is a configuration representing the ith vector. Adding these vectors to c_0 returns configurations within the acceptable penetration distance. If any of them is collision free, then we accept configuration c_0 (Figure 27). The minimum penetration depth found can be used to provide edge weights in the roadmap.

Node Generation and Connection. In this method, collision detection is replaced by a test for acceptable penetration, i.e., if the penetration is acceptable, a configuration which is in collision is accepted into the roadmap. The advantage of this approach is that we can use a standard PRM, including all node generation

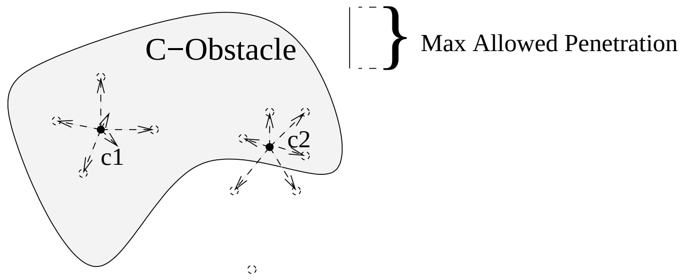


Fig. 27. Finding penetration of a configuration in C-Space. First a set of random vectors whose sizes are within the maximum allowed penetration are generated. Those vectors represent search directions and the depth of penetration. Since by adding at least two of such vectors, c_2 can reach collision free configurations, its penetration is acceptable. However c_1 does not have an acceptable penetration.

and connection methods. For example, the intermediate configurations that a local planner tests during the connection phase can be accepted based on their penetration. See Figure 26(b).

3. Query

While the roadmap was constructed using rigid bodies, a deformable version of the robot is used in the query phase. Since roadmap nodes and/or edges may be in collision, the query process must check them for collision and then, if collisions are found, call on some deformation method to try to deform the offending configurations into collision-free configurations. The query process below applies to roadmaps generated by either of the methods described above.

```
Path Query
Test if robot could be deformed in start and goal.
Connect start and goal to same CC
Connected=false;
while( !Connected )
    Find a path in CC connecting start and goal
    Sort edges from largest weight to lowest.
    for each edge
        if( test failed ) Delete this edge and break;
        endif
    endfor
    Connected=true;
endwhile
```

In the query, the deformation test essentially replaces the collision detection test for traditional motion planning. Now, a configuration will be accepted if its deformation energy is less than some threshold, which can be user-defined to give the user some control over the deformations.

Because deformation is an expensive operation, we need a quick way to determine if a path edge cannot be deformed. To do this, we sort path edges according to their weights (which are estimates of deformation cost), and test the edges with highest weights first as they are the most likely to fail. If a test fails, the edge is deleted from the roadmap and the process repeats. This approach is similar to Fuzzy PRM [99].

Figure 26(c) shows a path found by a query. Although the shortest path is on the right side of ball, our planner selected a longer path which had a smaller deformation energy.

D. Improving the Approximate Solution

We improve the approximate solution (i.e., a path for the rigid version of the robot with collisions) by deforming the robot. To construct large numbers of deformable models on–line (at query time), we should have fast deformation methods. However, it is also important to consider the physical properties of the model. Although we consider physical properties, our approach is not physically accurate, since our goal is to produce realistic *looking* deformations fast. For example, we do not attempt to precisely compute energy, as in [13].

Of the two objects participating in a deformation, one is the *deformable object* and the other is called the *deformer*. The deformer pushes (a portion of) the deformable object towards a collision-free state, and deformable object then changes shape according to these external forces. We implemented two deformation methods: bounding box deformation and geometric deformation.

1. Bounding-Box Deformation

The *deformable object* should react to external forces (pushes from the deformer) fast and properly. To achieve this, the *bounding box deformation* deforms the given model hierarchically. First, the top-level bounding box is deformed, and then the bottom-level internal model is deformed according to the deformation computed for the bounding box.

Our approach combines strategies of the ChainMail Deformation [45] and Free-Form Deformation (FFD) [108] to create real-time deformation which is sensitive to physical properties. The model is deformed by viewing the bounding box as a ChainMail model, and then deforming the ChainMail bounding box with respect to the deformer (workspace obstacle). This deformed bounding box defines an FFD lattice, which is used to deform the enclosed model using FFD. We next give a brief overview of ChainMail and FFD.

The ChainMail deformation was created to model tissue in surgical simulations [45]. Briefly, each vertex in the 3D ChainMail is connected to six neighboring vertices, and the maximum and minimum distances to these neighbors are set according to the material property. The user deforms models by moving a vertex, and the deformation is completed by adjusting the other vertices within the sphere of influence of the deformed vertex until all distance constraints are satisfied. In this work, we have developed a modified version of ChainMail that attempts to optimize performance. Breadth-first-search (BFS) is used to propagate vertex movements (i.e., adjust inter-

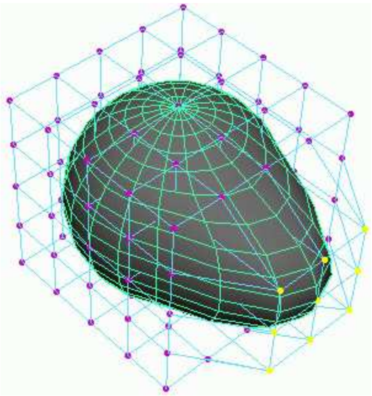


Fig. 28. FFD deformation with a 5 × 5 × 5 lattice.

vertex distances and constraints). To reject infeasible deformations quickly, vertices are processed in reverse order of the number of BFS trees in which they participate (since vertices contained in the largest number of BFS trees have higher failure rates). Figure 25 shows a revised ChainMail deformation.

Free Form Deformation, or FFD [108], is a well known interactive modeling tool which uses affine transformations to map local coordinates to deformed global coordinates. Figure 28 shows a sphere deformed when control points on the FFD lattice are moved. While FFD is extremely fast, it is generally difficult to obtain physically-based deformations since it requires tweaking many control points.

There are several benefits of this hybrid, hierarchical bounding box deformation. First, ChainMail provides physical properties, such as preservation of inter-vertex distances and elastic properties, and later FFD smoothly deforms the real model. One of the best aspects is that the motion planner can use only ChainMail, since it only requires the deformation energy, and utilize FFD only during rendering. Thus, the motion planning cost is independent of the complexity of the model since only the ChainMail bounding box is deformed during planning. The display cost, which uses FFD, will still be dependent on the model complexity.

After the object is deformed, the deformation energy can be calculated as the summation of the displacement of each vertex from its initial position to its deformed position.

The *deformer* pushes the deformable object to a collision free condition. Since the deformable object is converted to a bounding box, the implementation of the deformer is quite simple. The center of the bounding box is the guide for pushing all points. First, we check if the center of the bounding box is inside or outside the deformer (it is always outside the obstacles if scalable robots are used, but may be internal for the penetration method). For each point on the surface of the bounding box, we shoot a ray to the center point. If the ray intersects the deformer, the point is pushed along the ray until no intersection exists.

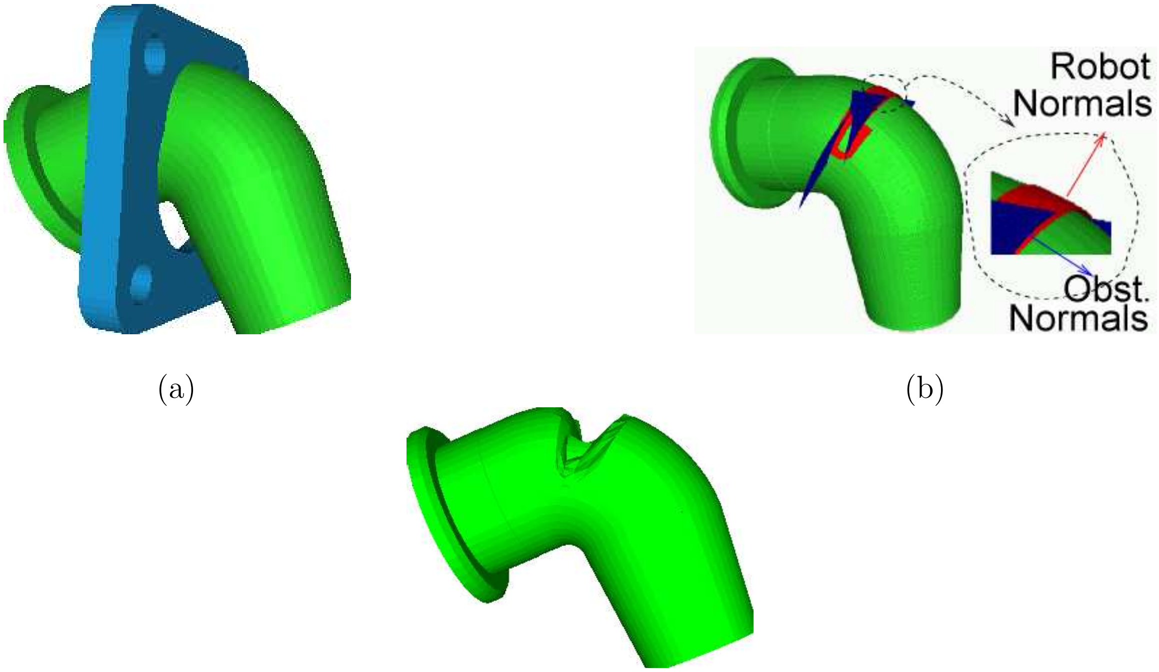
2. Geometric Deformation

In this approach we again use the obstacle as the deformer, and directly deform the colliding portion of the robot. If we knew the colliding volumes of the robot and obstacle, it would be relatively easy to move the colliding part of the robot out of collision. Unfortunately, for the complex models we consider, collision detection routines only return the intersecting polygons of the models.

Our algorithm continuously moves the colliding polygons of the robot until they are outside the obstacle. The most challenging part of this algorithm is to decide which direction to move intersecting polygons. An intuitive approach would be to move the robot polygons in the direction of the obstacle polygons' outward normals. Unfortunately, there are some important cases where this approach does not produce the desired results. For example, if most of the robot is on the 'other' side of the obstacle from the colliding polygons, we would prefer to deform the robot in the opposite direction suggested by this intuitive approach. To address this, we employ a sampling strategy to identify the correct direction. If we select directions randomly, we expect that directions toward larger external portions of the robot should become collision free faster than other directions. To improve efficiency, we restricted our sampling to the following directions: (i) normals of colliding polygons on the robot, (ii) normals of the colliding polygons (or average of several colliding neighbors) on the obstacle, (iii) some uniformly selected directions, and (iv) the inverse of the directions in (i-iii).

Figure 29 shows an example. The robot is the elbow shaped object and the obstacle is the hole shown in a colliding configuration in Figure 29(a). Some colliding polygons are shown in 29(b) After we choose a correct direction (from the types i-iv mentioned above), we move the robot's colliding polygons in that direction until we reach the collision free shape shown in 29(c).

If the robot does not fit in a narrow passage, then this method will fail to find a pushing direction unless we use a reduced-scale version of the robot. When the smaller robot is free, the colliding polygons of the original robot are moved towards the corresponding polygons on the smaller robot. Another problem with this method is that it can be distracted by remote collisions that occur when deforming some colliding region of the robot. To avoid this, we define a neighborhood function, which eliminates the influence of the remote collisions. Finally, this method can produce unrealistic looking deformations, because the resulting polygons may be quite large. In some cases we may even have topological changes. To address such issues, we added a constraint which states that none of the deformed edges can be shorter or longer than some percentage of the original edge. After the deformation a smoothing operation is applied to enforce this constraint. However, after smoothing we may end up with a shape that is in collision. If this occurs, we simply continue deforming, and then smoothing, until either a free deformation or the maximum number of iterations



(a) (b)

(c)

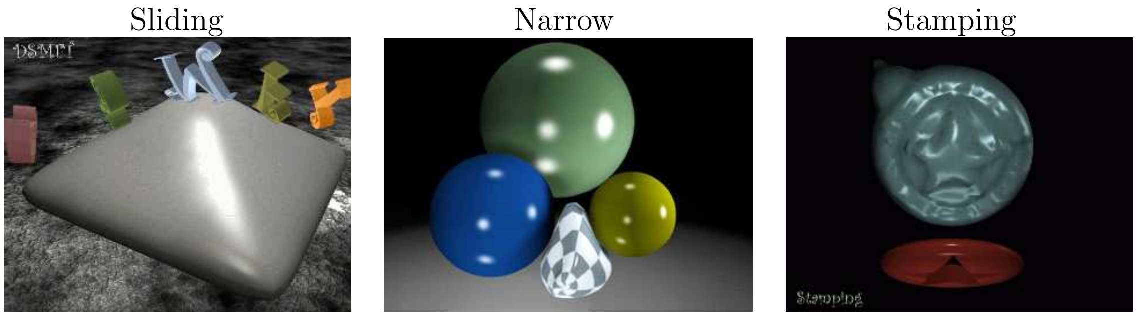
Fig. 29. An example geometric deformation. (a) A colliding configuration in the environment. The robot is the elbow and the obstacle is the hole. (b) Intersecting polygons of the robot and the obstacle and the normals to try. (c) Deformed version.

is reached.

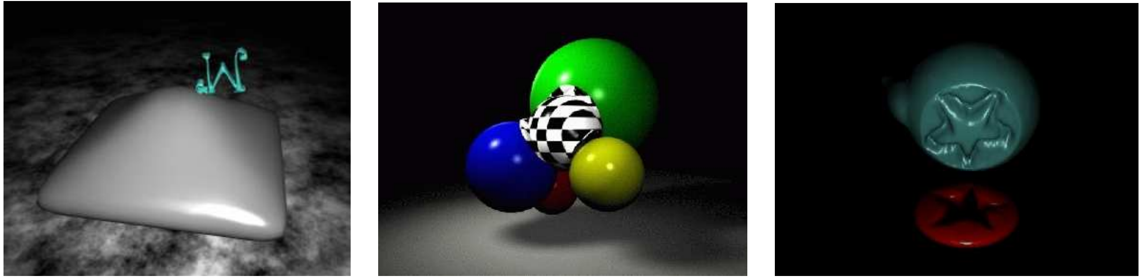
E. Experiments

In this section we report on experiments designed to study the scalable robot and penetration techniques for roadmap construction (Section C) and the bounding box (BBox) and geometric deformation techniques (Section D). All experiments were implemented on a PC with Athlon 1.33 Mhz CPU and 256MB RAM.

Our experiments investigate the performance of our algorithm in three situations requiring diverse deformations (Figure 30). In the *sliding* experiment, the letters "DSMFT" simulate multiple robots following a path passing very close to an obstacle. The path is found for one letter and then applied to the others. In the *narrow passage*


Sliding Narrow Stamping

(a)Bounding Box Deformation



(b) Geometric Deformation

Fig. 30. Snapshots of experiments. The first column is for the sliding environment, the middle, the second column is the narrow environment and the last column is stamping environment. The first row shows the bounding box deformation while the last row shows geometric deformation.

experiment, a deformable sphere (robot) passes through a narrow passage bounded by four different-sized rigid spheres (obstacles) contained in a bounding sphere (not shown for visibility). The *stamping* experiment enables us to see the effect of pushing the teapot (robot) against a complex seal (the obstacle).

Our first set of experiments studies the BBox and geometric deformation techniques. As reported in Table VII, both deformation techniques were applied to paths extracted from roadmaps constructed using the scaled robot and the penetration methods for all three environments. Snapshots of the deformed robots are shown in Figure 30, and movies can be found at http://parasol.tamu.edu/people/burchanb/. Our results show the BBox deformation is several hundred times faster than the

Table VII
Roadmap and Deformation Statistics

	Roadmap Generation				Average Deformatation Times(s)	
	Scaled Robot		Penetration			
Environment	#CC	Total Time (s) (gen.+con.)	#CC	Total Time(s) (gen.+con.)	Bounding Box	Geometric
Sliding	65	5.4(2.7+2.7)	16	16.5(8.4+8.1)	0.04	13.3
Narrow	87	16.2(0.5+15.7)	35	116.9(25.9+91)	0.40	86
Stamping	9	5.2(0.1+5.1)	1	124.2(50.54+73.77)	0.86	536

Roadmap and Deformation statistics. Total time includes node generation time and connection time for 1000 nodes. We use OBPRM for Sliding and PRM for Narrow and Stamping. Deformation time includes time for deformation, time for relaxation (for BBox only), and time for energy calculation. The average deformation time is for the successful deformations. All times are in seconds.

geometric deformation. This is due to the high cost of the search for the pushing directions and because collision detection costs are higher than in the BBox deformation where only the bounding box is tested for collision (as opposed to the entire model in the geometric deformation). The BBox method also produces smoother deformations, which is as expected because ChainMail directly propagates a deformation towards the neighboring points, whereas the geometric deformation propagates deformations recursively (i.e., indirectly). This difference in the methods can be mitigated by a good neighborhood function, perhaps determined experimentally for each environment. We note that the average cost of the deformations varies with the complexity of the environment and the deformations. In the BBox deformation, this is also affected by the number of lattice points in the bounding box.

Our second set of experiments studies the effects of the parameters for the roadmap generation techniques in the narrow passage environment (spheres). For the scalable robot method, roadmaps were constructed using two and six scaled robots (models). For the penetration method, we consider two penetration depths, $15r$ (small) and $25r$ (large), where r is the environmental resolution set for collision

TABLE VIII
COMPARISON OF ROADMAP GENERATION AND DEFORMATION TECHNIQUES

	ROADMAP GENERATION		BOUNDING BOX				GEOMETRIC			
			Time(s)				Time(min)			
GEN. METHOD	#CC	Time(s)	Query	Deform	#Deform	#Failed	Query	Deform	#Deform	#Failed
Scaled (2 models)	22	2.8	619	617	1403	25	N/A	N/A	N/A	N/A
Scaled (6 models)	21	4.3	82	81	199	0	322	322	212	1
Penetration (large)	3	14.9	144	144	328	4	514	514	344	3
Penetration (small)	14	25.6	75	75	162	2	255	255	174	1

Comparison of roadmap generation (350 nodes) and deformation methods for the narrow passage environment. The table shows query and total deformation times as well as the number of deformations and the number of the failed deformations. These values are for different roadmap generation methods; using 2 scaled robots and 6 scaled robots, enabling small and large penetration. Deform time is total deformation time in the query phase. The times for Bounding Box deformation are in seconds while the times for Geometric deformation are in minutes.

detection. As previously discussed, these parameters should approximate the deformation energy. The results in Table VIII show that both the scale factor and the penetration value are good indications of deformation energy. As expected, if a larger number of reduced-scale robots is used, or a smaller penetration is allowed, the resulting roadmap will be more expensive to compute, but will contain paths that are easier to deform. We also note that most (99% to 100%) of the query time is spent on deformation processing. (The geometric deformation did not find a solution within acceptable time limits for the roadmap generated by the two scaled robots.)

Although the main disadvantage of the geometric deformation is speed, it was also observed to sometimes alter the topology of the model. This could be solved by including self-collision checks as the object is deformed. Nevertheless, this method is suitable for some types of situations, e.g., narrow passage problems where the robot must assume the shape of the surrounding environment.

F. Conclusion and Future Work

This chapter considers motion planning for deformable objects. Withing the IRC paradigm, we have suggested two different methods for constructing and querying roadmaps (approximate solution), and have presented two deformation techniques that can be applied to the resulting path (improving the approximate solution). This approach decouples motion planning and deformation, and enables hybrid methods utilizing multiple methods in a “plug and play” fashion. Our future work includes optimizing the geometric deformation method and applying our algorithm to particle systems where several robots deform independently.

CHAPTER VII

CONCLUSIONS AND FUTURE WORK

In this thesis, we proposed a new paradigm, *Iterative Relaxation of Constraints (IRC)* to solve motion planning problems. In this method, a problem is relaxed by reducing the feasibility constraints of the problem. The solution found for this relaxed version (approximate solution) guides the automated planner towards a solution to the original problem (e.g., improves the approximate solution). As we have seen, IRC increases the automated planner's performance in most cases. An advantage of IRC is it is very flexible. The various constraint definitions, and different methods to generate and improve approximate solutions can be plugged into the system with ease.

In this dissertation, we have applied IRC to three different domains: (i) virtual prototyping (Chapter IV), (ii) ligand binding (Chapter V), and, (iii) deformable objects (Chapter VI). We have experimented with different relaxation criteria. We have shown methods to generate approximate solutions that are valid for the relaxed constraints and we have developed techniques to improve the approximate solutions. While generating the approximate solutions we have used not only automated methods but also manual methods. We have suggested different feedback methods (such as visual or haptic feedback) for users to understand the planner's process.

In virtual prototyping our goal was to test if a part would be accessible from the outside. In this domain, we relaxed the collision requirement. Our initial path was found by an automated planer using a scaled version of the part or selected by the user using a force-feedback device. In order to achieve a realistic sense of touch, we developed a heuristic method to determine the force that should be applied by the force feedback device. Our heuristic was successful in delivering a realistic touch sense in previously infeasible complex environments. We also relaxed the constraints by reducing the size of the parts we were checking for accessibility. We have described a novel roadmap visualization technique so that the users can understand the process of the planner and improve it by selecting critical paths. We would like to continue our work in this domain to increase the realism of the haptic feedback by distributing the heuristic process to multiple servers. We also want to develop more methods to improve approximate solutions.

In ligand binding, our goal was to generate candidate binding sites. Our initial solution was generated by relaxing the low energy requirement, e.g., our approximate solution contained ligand conformations with high energy . We have improved our initial solution by moving such conformations to nearby local minima of the potential. In generating the approximate solution, similar to virtual prototyping we experimented with both automated and manual (with a force feedback device) conformation generation. Our results showed that our approach to this domain is promising. We would like to continue our work in this domain by incorporating existing potential energy functions and scoring techniques. We are also interested in adding flexibility to the protein atoms and improving the haptic interface by allowing the user to feel flexible ligands.

For deformable objects, we deformed a robot to avoid collision while following a path. Our approximate solution was the path for a rigid version of the robot. We relaxed the collision free requirement by (a) using scaled versions of the robot, and (b) enabling penetration. To improve the approximate solution, we suggested two deformation methods. We would like to continue our research in this domain by improving the deformation methods and applying our approach to systems where several robots deform independently.

In addition to continuing our research in these domains, we are also interested in applying our IRC paradigm to new domains, developing new relaxation criteria, finding new approximate solution methods and implementing new improvement techniques.

REFERENCES

[1] J. M. Ahuactzin and K. Gupta, “A motion planning based approach for inverse kinematics of redundant robots: The kinematic roadmap,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’97)*, 1997, pp. 3609–3614.

[2] N. Allinger, “Conformational analysis 130. mm2. a hydrocarbon force field utilizing v1 and v2 torsional terms,” *J. Am. Chem. Soc.*, vol. 99, pp. 8127–8134, 1977.

[3] N. Allinger, K. Chen, and J.-H. Lii, “An improved force field (mm4) for saturated hydrocarbons,” *J. Comp. Chem.*, vol. 17, pp. 642–668, 1996.

[4] N. Allinger, Y. H. Yuh, and J.-H. Lii, “Molecular mechanics. The mm3 force field for hydrocarbons.,” *J. Am. Chem. Soc.*, vol. 111, pp. 8551–8565, 1989.

[5] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, “Choosing good distance metrics and local planners for probabilistic roadmap methods,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’98)*, 1998, pp. 630–637.

[6] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, “OBPRM: An obstacle-based PRM for 3D workspaces,” in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR’98)*, 1998, pp. 155–168.

[7] N. M. Amato, O. B. Bayazit, K. Kim, W. Son, and G. Song, “Cooperative motion planning: Providing hints to automatic motion planners,” in *Proc. Workshop on Motion Support in Virtual Prototyping, Stanford University*, 1999.

[8] N. M. Amato, O. B. Bayazit, and G. Song, “Providing haptic ‘hints’ to automatic motion planners,” in *Proc. Phantom User’s Group Workshop*, 1999.

[9] N. M. Amato, K. A. Dill, and G. Song, “Using motion planning to map protein folding landscapes and analyze folding kinetics of known native structures,” in *Proc. Int. Conf. Computational Molecular Biology (RECOMB’02)*, 2002, pp. 2–11.

[10] N. M. Amato, C. V. Jones, and D. Vallejo, “An adaptive framework for ‘single shot’ motion planning,” Dept. Comput. Sci., Texas A&M Univ., Tech. Rep. 98-025, Nov 1998.

[11] N. M. Amato and G. Song, “Using motion planning to study protein folding pathways,” *J. Comp. Biol.*, vol. 9, no. 2, pp. 149–168, 2002. *RECOMB’01* special issue.

[12] E. Anshelevich, S. Owens, F. Lamiraux, and L. Kavraki, “Deformable volumes in path planning applications,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’00)*, 2000, pp. 2290–2295.

[13] E. Anshelevich, S. Owens, F. Lamiraux, and L. Kavraki, “Deformable volumes in path planning applications,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’00)*, 2000, pp. 2290–2295.

[14] M. Apaydin, A. Singh, D. Brutlag, and J.-C. Latombe, “Capturing molecular energy landscapes with probabilistic conformational roadmaps,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’01)*, 2001, pp. 932–939.

[15] R. Avila and L. Sobierajski, “A haptic interaction method for volume visualization,” in *Proc. IEEE Conf. Visualization*, 1996, pp. 197–204.

[16] A. H. Barr, “Global and local deformations of solid primitives,” in *Proc. ACM SIGGRAPH*, 1984, pp. 21–30.

[17] O. B. Bayazit, J.-M. Lien, and N. M. Amato, “Probabilistic roadmap motion planning for deformable objects,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’02)*, 2002, pp. 2126–2133.

[18] O. B. Bayazit, J.-M. Lien, and N. M. Amato, “Better flocking behaviors using rule-based roadmaps,” in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR’02)*, Dec 2002. To appear.

[19] O. B. Bayazit, J.-M. Lien, and N. M. Amato, “Better group behaviors in complex environments using global roadmaps,” in *Proc. Artificial Life*, 2002, pp. 362–370.

[20] O. B. Bayazit, J.-M. Lien, and N. M. Amato, “Roadmap-based flocking for complex environments,” in *Proc. Pacific Graphics*, 2002, pp. 104–113.

[21] O. B. Bayazit, G. Song, and N. M. Amato, “Enhancing randomized motion planners: Exploring with haptic hints,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’00)*, 2000, pp. 529–536.

[22] O. B. Bayazit, G. Song, and N. M. Amato, “Enhancing randomized motion planners: Exploring with haptic hints,” *Autonomous Robots, Special Issue on Personal Robotics*, vol. 10, no. 2, pp. 163–174, 2001. Preliminary version appeared in *ICRA’00*, pp. 529–536.

[23] O. B. Bayazit, G. Song, and N. M. Amato, “Ligand binding with OBPRM and haptic user input: Enhancing automatic motion planning with virtual touch,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’01)*, 2001, pp. 954–959. This work was also presented as a poster at *RECOMB’01*.

[24] O. B. Bayazit, G. Song, and N. M. Amato, “Ligand binding with obprm and

haptic user input: Enhancing automatic motion planning with virtual touch," in *Currents in Computational Molecular Biology* (N. El-Mabrouk, T. Lengauer, and D. Sankoff, eds.), pp. 81–82, Les Publications CRM, Montreal, Canada, 2001. Book includes short papers from The Fifth ACM Int. Conf. on Computational Molecular Biology (*RECOMB'01*), Montreal, Canada, April 2001.

[25] P. Berkelman, R. Hollis, and S. Salcudean, "Interacting with virtual environments using a magnetic levitation haptic interface," *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS'95)*, 1995. pp. 117–122.

[26] R. Bohlin and L. E. Kavraki, "Path planning using Lazy PRM," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'00)*, 2000, pp. 521–528.

[27] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'99)*, 1999, pp. 1018–1023.

[28] J. Brickmann, W. Heiden, H. Vollhardt, and C.-D. Zachmann, "New man-machine communication strategies in molecular modeling," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'95)*, 1995, pp. 273–282.

[29] O. Brock and L. Kavraki, "Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration places," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'01)*, 2001, pp. 1469–1474.

[30] B. Brooks, R. Bruccoleri, B. Olafson, D. States, S. Swaminathan, and M. Karplus, "Charmm: a program for macromolecular energy, minimization and dynamics calculations," *J. Comp. Chem.*, vol. 4, pp. 187–217, 1983.

[31] J. Canny, "On detecting collisions between moving polyhedra," *IEEE Trans. Patt. Analy. Mach. Intell.*, vol. 8, no. 2, pp. 200–209, 1986.

[32] J. Canny, "A fast algorithm for incremental distance calculation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'91)*, 1991, pp. 1008–1014.

[33] H. Chang and T. Y. Li, "Assembly maintainability study with motion planning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'95)*, 1995, pp. 1012–1019.

[34] H. Choi and S. Lee, "Force display system for manipulating thin flexible objects," in *Proc. IEEE Int. Workshop on Robot and Human Communication*, 1996, pp. 507–512.

[35] M. Clark, R. D. C. III, and N. van Opdenhosch, "Validation of the general purpose tripose 5.2 force field," *J. Comp. Chem.*, vol. 95, pp. 8005–8025, 1989.

[36] J. Colgate, M. Stanley, and J. Brown, "Issues in the haptic display of tool use," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS'95)*, 1995, pp. 140–145.

[37] H. Das, "Shared real-time human-computer control of redundant link manipulator ", Sc.D. dissertation, MIT, Cambridge, MA 1989.

[38] P. Dauber-Osguthorpe, V. A. Roberts, D. O. J. Wolff, M. Genest, and T. A. Hagler, "Structure and energetics of ligand binding to proteins: E. coli dihydrofolate reductase- trimethoprim, a drug-receptor system," *Proteins: Structure, Function and Genetics*, vol. 4, pp. 31–47, 1989.

[39] A. P. del Pobil, M. A. Serna, and J. Llovet, "A new representation for collision avoidence and detection," in *Proc. IEEE Int. Conf. Robotics and Automation*

(ICRA'92), 1992, pp. 246–251.

[40] R. DesJarlais, R. Sheridan, R. Nilakatan, K. Haraki, N. Bauman, and R. Venkataraghavan, "A shape- and chemistry-based docking method and its use in the design of HIV-1 protease inhibitors," *J. Comp.-Aid. Mol. Des.*, vol. 8, pp. 231–242, 1994.

[41] M. Eldridge, C. Murray, T. Auton, G. Paolini, and R. Mee, "Empirical scoring functions: I. the development of a fast empirical scoring function to estimate the binding affinity of ligands in receptor complexes," *J. Comp.-Aid. Mol. Des.*, vol. 11, pp. 425–445, 1997.

[42] S. Everett and R. Dubey, "Model-based variable position mapping for telerobotic assistance in a cylindrical environment," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'99)*, 1999, pp. 2197–2202.

[43] L. Fabiani, G. Burdea, N. Langrana, and D. Gomez, "Human interface using the rutgers master ii force feedback interface," in *Proc. IEEE Virtual Reality Annual Int. Symp.*, 1996, pp. 54–59.

[44] S. Fisher and M. C. Lin, "Fast penetration depth estimation for elastic bodies using deformed distance fields," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS'01)*, 2001, pp. 330-336.

[45] S. Gibson, "3D Chainmail: A fast algorithm for deforming volumetric objects," in *Proc. Symp. Interactive 3D Graphics*, 1997, pp. 149–154.

[46] S. Gibson, "Using linked volumes to model object collision, deformation cutting, carving, and joining," in *IEEE Visualization and Computer Graphics*, pp. 169–177, 1999.

[47] S. Gibson and B. Mirtich, “A survey of deformable modeling in computer grapics,” MERL, Cambridge, MA, Tech. Rep. TR-97-19, 1997.

[48] A. Gregory, M. Lin, S. Gottschalk, and R. Taylor, “A framework for fast and accurate collision detection for haptic interaction,” in *Proc. IEEE Virtual Reality Conf.*, 1999, pp. 38–45.

[49] C. E. Guestrin and D. Ormoneit, “Robust combination of local controllers,” in *Uncertainty in Artificial Intelligence*, pp. 178–185, 2001.

[50] C. Guo, T. Tarn, N. Xi, and A. Bejczy, “Fusion of human and machine intelligence for telerobotic systems,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’95)*, 1995, pp. 3110–3115.

[51] T. Halgren, “Merck molecular force field. i. basis, form, scope, parameterization, and performance of MMFF94,” *J. Comp. Chem.*, vol. 17, pp. 490–519, 1996.

[52] L. Han and N. M. Amato, “A kinematics-based probabilistic roadmap method for closed chain systems,” in *Algorithmic and Computational Robotics – New Directions (WAFR 2000)*, pp. 233–246, 2000.

[53] E. Harvey and C. Gingold, “Haptic representation of the atom,” in *Proc. IEEE Int. Conf. Information Visualization*, 2000, pp. 232–235.

[54] M. Hernando, E. Gambao, E. Pinto, and A. Barrientos, “Collision control in teleoperation by vritual force reflection,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’99)*, 1999, pp. 565–570.

[55] C. Holleman and L. Kavraki, “A framework for using the workspace medial axis in PRM planners,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’00)*, 2000, pp. 1408–1413.

[56] D. House, R. DeVaul, and D. Breen, "Towards simulating cloth dynamics using interacting particles," *Int. J. Cloth. Sci. Tech.*, vol. 8, no. 3, pp. 75–94, 1996.

[57] D. Hsu, L. Kavraki, J.-C. Latombe, R. Motwani, and S. Sorkin, "On finding narrow passages with probabilistic roadmap planners," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR'98)*, 1998, pp. 141–153.

[58] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized Kinodynamic Motion Planning with Moving Obstacles," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR'00)*, 2000, pp. SA1–SA18.

[59] D. Hsu, J.-C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'97)*, 1997, pp. 2719–2726.

[60] Y. K. Hwang and N. Ahuja, "Gross motion planning – a survey," *ACM Computing Surveys*, vol. 24, no. 3, pp. 219–291, 1992.

[61] R. M. T. II, J. Chen, S. Okimoto, N. Llopis-Artime, V. L. Chi, F. P. B. Jr., M. Falvo, S. Paulson, P. Thiansanthaporn, D. Glick, S. Washburn, and R. Superfine, "Pearls found on the way to the ideal interface for scanned-probe microscopes," in *Proc. IEEE Conf. Visualization*, 1997, pp. 467–470.

[62] M. Jägersand, "Image based predictive display for tele-manipulation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'99)*, 1999, pp. 550–556.

[63] J.D.Cohen, M.C.Lin, D.Manocha, and M. Ponamgi, "I-collide: An interactive and exact collision detection system for large-scale environment," in *Proc. Symp. on Interactive 3D Graphics*, 1995, pp. 189–196.

[64] D. Johnson and E. Cohen, “A framework for efficient minimum distance computations,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’98)*, 1998, pp. 678–3684.

[65] G. Jones, P. Willett, R. C. Glen, A. R. L. Leach, and R. Taylor, “Development and validation of a genetic algorithm for flexible docking,” *J. Mol. Biol.*, vol. 245, pp. 43–53, 1985.

[66] D. Jung and K. Gupta, “Octree-based hierarchical distance maps for collision detection,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’96)*, 1996, pp. 454–459.

[67] E. Katchalski-Katzir, I. Shariv, M. Eisenstein, A. Friesem, and C. A. I. Vakser, “Molecular surface recognition: Determination of geometric fit between proteins and their ligands by correlation techniques,” in *Natl. Acad. Sci. USA*, vol. 89, pp. 2195–2199, 1992.

[68] L. Kavraki, F. Lamiraux, and C. Holleman, “Towards planning for elastic objects,” in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR’98)*, 1998, pp. 313–325.

[69] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Trans. Robot. Automat.*, vol. 12, pp. 566–580, August 1996.

[70] D. Kim, L. Guibas, and S. Shin, “Fast collision detection among multiple moving spheres,” in *Proc. ACM Symp. on Computational Geometry (SoCG’97)*, 1997, pp. 373–375.

[71] R. Kindel, D. Hsu, J.-C. Latombe, and S. Rock, “Kinodynamic motion planning

amidst moving obstacles," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'00)*, 2000, pp. 537–543.

[72] J. Klosowski, M. Held, J. Mitchell, H. Sowizral, and K. Zikan, "Efficient collision detection using bounding volume hierarchies of k-DOPs," *IEEE Trans. Visual. Comp. Graph.*, vol. 4, no. 1, 1998.

[73] Y. Kunii and H. Hashimoto, "Tele-teaching by human demonstration in virtual environment for robotic network system," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'97)*, 1997, pp. 405–410.

[74] I. Kuntz, "Structure-based strategies for drug design and discovery," *Science*, vol. 257, pp. 1078–1082, 1992.

[75] A. Ladd and L. Kavraki, "Generalizing the analysis of PRM," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'02)*, 2002, pp. 2120–2125.

[76] F. Lamiraux and L. Kavraki, "Planning paths for elastic objects under manipulation constraints," *Int. J. Robot. Res.*, vol. 20, no. 3, pp. 188–208, 2001.

[77] J. C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers, 1991.

[78] S. LaValle, P. Finn, L. Kavraki and J.-C. Latombe, "Efficient database screening for rational drug design using pharmacophore-constrained conformational search," *J. Comp. Chem.*, vol. 21, no. 9, pp. 731–747, 2000.

[79] S. LaValle, J. Yakey, and L. Kavraki, "A probabilistic roadmap approach for systems with closed kinematic chains," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'99)*, 1999, pp. 1671–1677.

[80] J. Leech, J. F. Prins, and J. Hermans, “Smd: Virtual steering of molecular dynamics for protein design,” *IEEE Computational Science and Engineering*, vol. 3, pp. 38–45, 1996.

[81] D. Levine, M. Facello, P. Hallstorm, G. Reeder, B. Walenz, and F. Stevens, “Stalk: An interactive system for virtual molecular docking,” *IEEE Computational Science & Engineering*, vol. 4, pp. 55–66, 1997.

[82] M. Levitt, “Protein folding by restrained energy minimization and molecular dynamics,” *J. Mol. Biol.*, vol. 170, pp. 723–764, 1983.

[83] J.-M. Lien, M. Morales, and N. M. Amato, “Neuron PRM: A framework for constructing cortical networks,” PARASOL Lab, Dept. Comput. Sci., Texas A&M Univ., Tech. Rep. 01-002, Oct 2001.

[84] M. Liu and S. Wang, “MCDOCK: A Monte Carlo simulation approach to the molecular docking problem,” *J Comp.-Aid. Mol. Des.*, vol. 13, no. 5, pp. 435–451, 1999.

[85] G. Luecke and J. Edwards, “Virtual cooperating manipulators as a virtual reality haptic interface,” in *Proc. the Third Annual Symp. on Human Interaction with Complex Systems (HICS’96)*, 1996, pp. 133–140.

[86] H. Maekawa and J. M. Hollerbach, “Haptic display for object grasping and manipulating in virtual environment,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’98)*, 1998, pp. 2566–2573.

[87] T. Maneewarn and B. Hannaford, “Haptic feedback of kinematic conditioning for telerobotic applications,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS’98)*, 1998, pp. 1260–1265.

[88] B. Martínez-Salvador, A. del Pobil, and M. Pérez-Francisco, "Very fast collision detection for practical motion planning part i: The spatial representation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'98)*, 1998, pp. 625–629.

[89] T. H. Massie and J. K. Salisbury, "The PHANToM haptic interface: A device for probing virtual objects," in *Int. Mechanical Engineering Exposition and Congress, DSC 55-1*, (Chicago), pp. 295–302, C. J. Radcliffe, ed., ASME, 1994.

[90] E. Meyer, S. Swansan, and J. Williams, "Molecular modeling and drug design," *Pharmacology & Therapeutics*, vol. 85, pp. 113–121, 2000.

[91] G. S. P. Miller, "The motion dynamics of snakes and worms," in *Proc. ACM SIGGRAPH*, 1988, pp. 169–177.

[92] M. Miller, S. Kearsley, D. Underwood, and R. Sheridan, "Flog: A system to select 'quasi flexible' ligands complementary to a receptor if known three-dimensional structure," *J. Comp.-Aid. Mol. Des.*, vol. 8, pp. 153–174, 1994.

[93] B. Mirtich, "V-clip: Fast and robust polyhedral collision detection," *ACM Trans. Graphics*, vol. 17, no. 3, pp. 177–208, 1996.

[94] B. Mirtich, "V-clip: Fast and robust polyhedral collision detection," MERL, Cambridge, MA, Tech. Rep. TR97-05, 1997.

[95] G. Morris, D. Goodsell, R. Halliday, R. Huey, W. Hart, R. Belew, and A. Olson, "Automated docking using a lamarckian genetic algorithm and empirical binding free energy function," *J. Comp. Chem.*, vol. 19, pp. 1639–1662, 1998.

[96] I. Muegge, Y. Martins, P. Hajduk, and S. Fesik, "Evaluation of pmf scoring in docking weak ligands to the fk506 binding protein," *J. Medicinal Chemistry*, vol. 42, no. 14, pp. 2498–2503, 1999.

[97] A. Nahvi, D. D. Nelson, J. M. Hollerbach, and D. E. Johnson, "Haptic manipulation of virtual mechanisms from mechanical cad designs," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'98)*, 1998, pp. 375–380.

[98] C. L. Nielsen and L. E. Kavraki, "A two level fuzzy PRM for manipulation planning,"Comput. Sci., Rice Univ., Houston, TX, Tech. Rep. TR2000-365, 2000.

[99] C. L. Nielsen and L. E. Kavraki, "A two lovel fuzzy PRM for manipulation planning," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS'00)*, 2000, pp. 1716–1721.

[100] M. Ouh-young, "Force display in molecular docking, tr90-004," Ph.D. dissertation, Univ. of North Carolina at Chapel Hill, Chapel Hill, NC, 1990.

[101] M. Ouh-Young, M. Pique, J. Hughes, N. Srinivisan, and F. P. Brooks, Jr., "Using a manipulator for force display in molecular docking," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'88)*, 1988, pp. 1824–1829.

[102] M. Ouh-young, D. V. Beard, and F. P. Brooks, Jr., "Force display performs better than visual display in a simple 6-d docking task," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'89)*, 1989, pp. 1462–1466.

[103] R. Pachter and J. A. Lupo, "Virtual reality for materials design," in *Proc. The 14th Annual Dayton Section Symposium AESS/IEEE*, 1997, pp. 51–56.

[104] N. Pattabiraman, M. Levitt, T. E. Ferrin, and R. Langridge, "Computer graphics in real time docking with energy calculation and minimization," *J. Comp. Chem.*, vol. 6, pp. 432–436, 1985.

[105] M. Rarey, B. Kramer, T. Lengauer, and G. Klebe, "A fast flexible docking method using an incremental construction algorithm," *J. Mol. Biol.*, vol. 261, pp. 470–489, 1996.

[106] J. Reif, "Complexity of the piano mover's problem and generalizations," in *Proc. IEEE Symp. Foundations of Computer Science (FOCS'79)*, 1979, pp. 421–427.

[107] P. Richard, G. Burdea, G. Birebent, D. Gomez, N. Langrana, and P. Coiffet, "Effect of frame rate and force feedback on virtual object manipulation," *Presence - Teleoperators and Virtual Environments*, vol. 5, no. 1, pp. 95–108, 1996.

[108] T. Sederberg and S. Parry, "Free-form deformation of solid gemetic models," in *Proc. ACM SIGGRAPH*, 1986, pp. 151–160.

[109] *GHOST Software Developer's Toolkit Programmer's Guide Version 2.* Cambridge, MA: Sensable Technologies, 1998.

[110] T. Sheridan, "Telerobotics," *Automatica*, vol. 25, no. 4, pp. 487–507, 1989.

[111] M. K. A. Shomper, "Haptic rendering," Virtual Environment, 3D Medical Imaging, and Computer Graphics Laboratory, Air Force Institute of Technology, Wright-Patterson Air Force Base, Tech. Rep., 1997. Presented at the 1997 AESS Dayton Symposium.

[112] A. Singh, J. Latombe, and D. Brutlag, "A motion planning approach to flexible ligand binding," in *7th Int. Conf. Intelligent Systems for Molecular Biology (ISMB)*, pp. 252–261, 1999.

[113] M. Sitti and H. Hashimoto, "Tele-nanorobotics using atomic force microscope," in *Proc. IEEE/RSJ Intl. Conf. Intelligent Robots and Systems (IROS'98)*,

pp. 1739–1746, 1998.

[114] G. Song and N. M. Amato, "A motion planning approach to folding: From paper craft to protein folding," Dept. Comput. Sci., Texas A&M University, Tech. Rep. TR00-017, July 2000.

[115] G. Song and N. M. Amato, "A motion planning approach to folding: From paper craft to protein folding," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'01)*, 2001, pp. 948–953.

[116] G. Song and N. M. Amato, "Randomized motion planning for car-like robots with c-prm," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS'01)*, 2001, pp. 37–42.

[117] G. Song and N. M. Amato, "Using motion planning to study protein folding pathways," in *Proc. Int. Conf. Computational Molecular Biology (RECOMB'01)*, 2001, pp. 287–296.

[118] G. Song and N. M. Amato, "A motion planning approach to folding: From paper craft to protein folding," *IEEE Trans. Robot. and Automat.*, 2002. Conditionally accepted. Preliminary version appeared in *ICRA'01* , pp. 948-953.

[119] G. Song, S. L. Miller, and N. M. Amato, "Customizing PRM roadmaps at query time," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'01)*, 2001, pp. 1500–1505.

[120] G. Song, S. L. Thomas, and N. M. Amato, "A general framework for PRM motion planning," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA'03)*, 2003. Under submission.

[121] G. Song, S. Thomas, K. Dill, J. Scholtz, and N. Amato, “A path planning-based study of protein folding with a case study of hairpin formation in protein G and L,” in *Proc. Pacific Symposium of Biocomputing (PSB’03)*, 2003. To appear.

[122] S. Sundaram, I. Remmler, and N. Amato, “Disassembly sequencing using a motion planning approach,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’01)*, 2001, pp. 1475–1480.

[123] M. C. Surles, “Techniques for interactive manipulation of graphical protein models,” Ph.D. dissertation, Univ. of North Carolina at Chapel Hill, Chapel Hill, NC, 1992.

[124] M. Surles, J. Richardson, D. Richardson, and F. Brooks, Jr., “Sculpting proteins interactively: Continual energy minimization embedded in a graphical modeling system,” *Protein Sci.*, vol. 3, pp. 198–210, 1994.

[125] P. Taylor and S. Rankov, “A visual/tactual virtual display,” in *Proc. IEEE Int. Conf. Control*, 1998, pp. 745–750.

[126] M. Totrov and R. Abagyan, “Derivation of sensitive discrimination potential for virtual ligand screening,” in *Third Annual Int. Conf. on Computational Molecular Biology*, 1999, pp. 312–320.

[127] X. Tu and D. Terzopoulos, “Artifical fishes: physics, locomotion, perception, behavior,” in *Proc. ACM SIGGRAPH*, 1994, pp. 43–50.

[128] D. Vallejo, C. Jones, and N. Amato, “An adaptive framework for ‘single shot’ motion planning,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS’00)*, 2000, pp. 1722–1727.

[129] D. Vallejo, I. Remmler, and N. Amato, “An adaptive framework for ‘single shot’ motion planning: A self-tuning system for rigid and articulated robots,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’01)*, 2001, pp. 21–26.

[130] W. van Gunsteren and H. J. C. Berendsen, *Groningen molecular simulation (GROMOS) library manual*. Groningen, Holland: Biomos, 1987.

[131] R. Vunner, J. O’Brien, and J. Hodgins, “Animating sand, mud, and snow,” in *Computer Graphics Forum*, vol. 18, pp. 17–26, 1999.

[132] L. Wanger, “Haptically enhanced molecular modeling: A case study,” in *Proc. the Third PHANTOM Users Group Workshop*, 1998.

[133] S. Weiner, P. Kollman, D. Case, U. Singh, G. Alagona, S. P. Jr., and P. Weiner, “A new force field for the molecular mechanical simulation of nucleic acids and proteins,” *J. Am. Chem. Soc.*, vol. 106, pp. 765–784, 1984.

[134] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, “MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA’99)*, 1999, pp. 1024–1031.

[135] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, “Motion planning for a rigid body using random networks on the medial axis of the free space,” in *Proc. ACM Symp. on Computational Geometry (SoCG’99)*, 1999, pp. 173–180.

[136] P. G. Xavier and R. A. LaFarge, “A configuration space toolkit for automated spatial reasoning: Technical results and ldrd project final report,” Sandia National Laboratories, Albuquerque, NM, Tech. Rep. SAND97-0366, Sandia National Laboratories, 1997.

VITA

Osman Burçhan Bayazıt was born on October 21, 1971, in Adana, Turkey. He graduated from Susurluk Lycee in 1988 and received his B.S. in Computer Engineering at Middle East Technical University, Ankara, Turkey, in 1994. He continued his graduate study at Middle East Technical University until he received a sponsorship from the Turkish Ministry of Education in 1996 to pursue graduate studies abroad. From 1994 to 1996 he worked as a teaching and research assistant at Middle East Technical University. From 1996, until now he has been a student and graduate assistant in the Department of Computer Science at Texas A&M University, where he received an M.S. degree in Computer Science in 1998.

Mr. Bayazıt's permanent address is Yeni Mah. İstasyon Cad. Demirel Apt. 59/3, 10600 Susurluk/BALIKESİR, Turkey.