Thèse

présentée au

Laboratoire d'Analyse et d'Architecture des Systèmes

en vue de l'obtention du

Doctorat de l'*Institut National Polytechnique de Toulouse*
Ecole Doctorale Systèmes

Spécialité : **Systèmes Automatiques / Robotique**

par **Juan Cortés**

Algorithmes pour la Planification de Mouvements de Mécanismes Articulés avec Chaînes Cinématiques Fermées

---

Motion Planning Algorithms for General Closed-Chain Mechanisms

---

Soutenue le **16 Décember 2003**

devant le Jury composé de :

Lydia E. Kavraki	Rice University, Houston	***Rapporteurs***
Steven M. LaValle	University of Illinois, Urbana	
Raja Chatila	LAAS-CNRS, Toulouse	***Examinateurs***
Jean-Paul Laumond	LAAS-CNRS, Toulouse	
Jean-Pierre Merlet	INRIA, Sophia Antipolis	
Pierre Monsan	INSA, Toulouse	***Membre invité***
Thierry Siméon	LAAS-CNRS, Toulouse	***Directeur de thèse***

Ojalá que esta tesis pueda aportar a la Ciencia al menos una pequeña parte de lo que su realización me ha aportado a mi.

In memory of my grandmother Cristina.
Dead the 25^{th} of June 2003.

Acknowledgments

Three years. Three full years. Three years of experiences. Three years of learning. Three years of contrasts: hope and despair, happiness and sadness, satisfaction and dislike, harmony and discord. But, in the balance, three very positive years, during which I have matured as a person, and, I hope, as a scientist. For this reason, I have to express my sincere gratitude to the persons who made it possible and to all the people who have helped me and have accompanied me during this period.

First of all, I have to thank to the main responsibles of this adventure. Thanks to Thierry Siméon for being much more than an excellent advisor. Thanks to Jean-Paul Laumond for being always there, ready to give a bit of geniality. I don't want to forget to Luis Montano, who made me meet them.

My gratitude to the directors of the LAAS-CNRS, Jean-Claude Laprie and Malik Ghallab, and to the head of the group RIA, Raja Chatila, for the material support. And thanks to people of group RIA and Sysadmin for their frequent helps.

I also thank Lydia Kavraki and Steven LaValle for the review of this thesis. Their comments and suggestions have inestimable value for me.

A very special thanks to Lluís Ros and Angel Sappa. To see their passion for research helped me decide to start a PhD. And also thanks to other people with whom I have had the pleasure of collaborating or interacting, particularly: Gwénaëlle André, Paul Bates, Patrick Danes, Fabien Gravot, Didier Henrion, Leonard Jaillet, Jean-Pierre Merlet, Magalie Remaud-Siméon, Marc Renaud, Vicente Ruiz, Anis Sahbani, Federico Thomas and Vinh Tran.

Finally, I have to mention my family and friends, my main source of motivation. Specially, thanks to Christophe for all the beautiful moments we have shared, and thanks to Odri for the nice illustrations of this thesis, but above all, thanks for loving me.

Contents

Introduction

Motion Planning is a fundamental issue in Robotics. An intelligent machine must be able to decide how to move in its environment. The motion planning problem consists in finding feasible paths for a mobile system, the robot, in a physical world. The notion of *configuration-space* [Lozano-Pérez 83] allows a geometrical formulation of the problem such that it is reduced to find the connectivity of subsets of a n-dimensional manifold. A detailed formulation of the problem and a collection of basic techniques can be found in [Latombe 91]. This same formulation can be adopted for problems in other very different domains [Latombe 99]. For instance, if virtual actors are modeled like robots, then motion planning techniques become tools for graphic animation [Koga 95, Kuffner 99]. Questions involving motion of objects often rise during the design of prototypes of products and their manufacturing using CAD/CAM systems. Motion planners, as integral components of these software packages, are capable to answer these questions [Gottschlich 92, Ferré 03]. In industrial logistics, motion planing algorithms have an application to assist difficult maintenance operations [Siméon 01c] or the transport of large objects [Lamiraux 03]. In medical applications, finding a minimally-invasive path of a surgical tool, given a 3D model of the patient's body, can also be seen as a motion planning problem [Tombropoulos 99]. There is a clear resemblance between the structural representation of robots an molecules [Parsons 94, Kavraki 97]. Motion planing strategies can be used to solve complex problems in computational Chemistry and Biology [Finn 98, LaValle 00, Apaydin 02]. With slight variations, the formulation of the motion planning problem can be enlarged to more complex problems such as manipulation planning [Alami 95, Siméon 03] and kinodymamic planning [Donald 93, Fraichard 99, LaValle 01b]. Then, extensions of motion planners can be studied to solve them.

The motion planning problem was first formulated for a single rigid objects whose only motion constraints arise from the presence of static obstacles. It is usually referred to as the *piano mover's problem* [Schwartz 83, Schwartz 84]. Solving this basic instance is already a challenging task which has attracted the interest of many scientists over the last two decades. The development of motion planning algorithms began with this particular case, and some planners have been proposed that provide an exact solution (e.g. [Canny 88]). Nevertheless, most of the above mentioned practical applications require to deal with much more complex instances of the problem that such exact methods are unable to treat. The difficulty of motion planning depends on the complexity of the mobile system and on the intrinsic and extrinsic motion constraints. Practical motion planning algorithms, able to handle such difficulties, have been developed in the last ten years (e.g. [Kavraki 96, LaValle 98]). These planners use sampling techniques combined with efficient geometric tools to construct data structures that encode the connectivity of the feasible subsets in the search-space. The efficacy and generality of sampling-based motion planning algorithms have been demonstrated by the several research groups working on

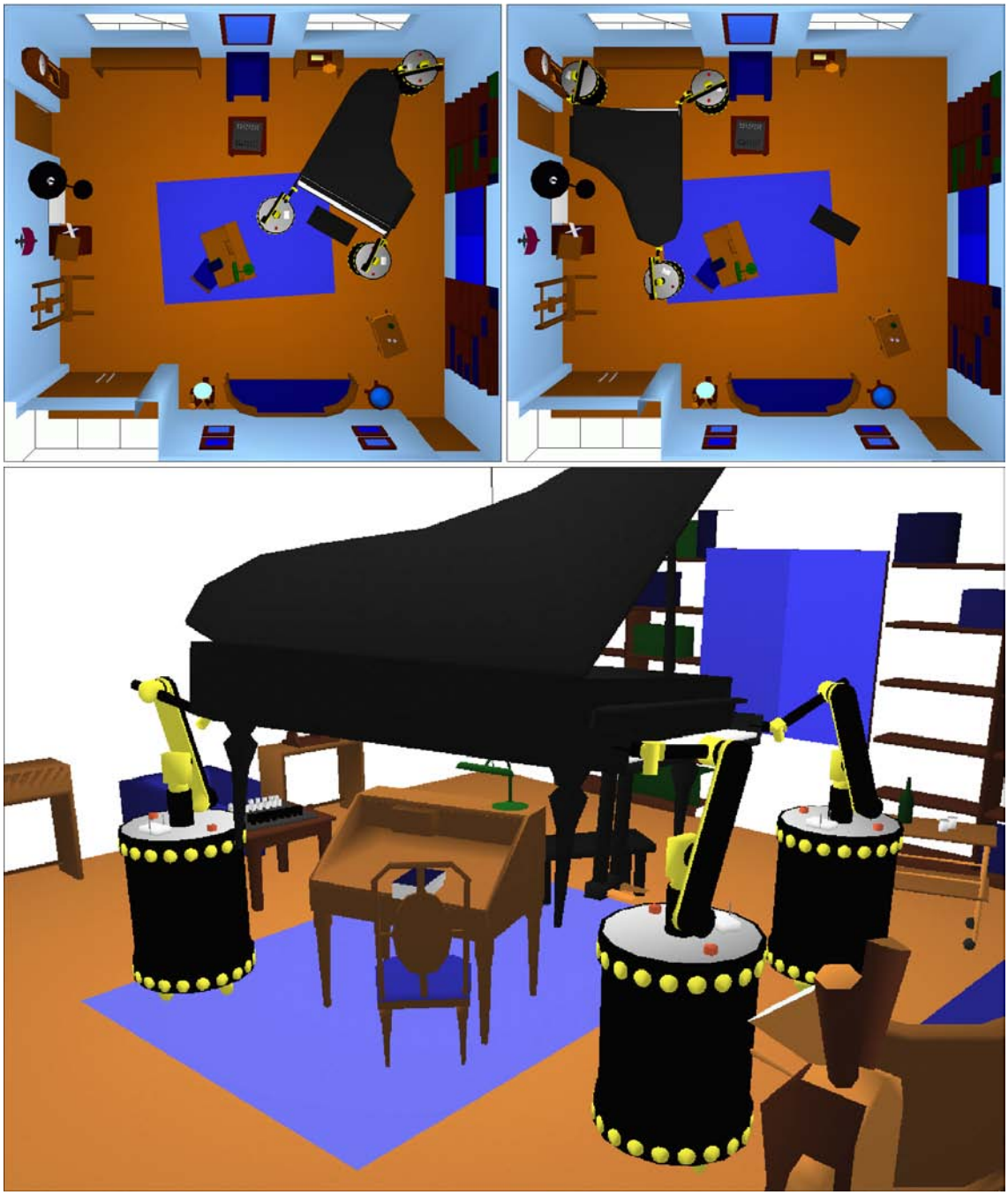


Figure 1: A “robotized” version of the piano mover’s problem. The situation of the piano in the room is changed by cooperating mobile manipulators.

them. Although they are becoming well-known techniques, some questions still remain, and theoretical development must continue in this direction.

A particular kind of constraint that notably increases the difficulty of the motion planning problem is the closure of kinematic chains. This thesis deals with motion planning under such constraints.

Motion Planning for Closed-Chain Mechanisms

We address articulated mechanism containing closed kinematic chains. Such a structural particularity induces intrinsic motion restrictions. Some of the variables in the motion planning problem are related by non-linear equations that express loop closure constraints. In general, a complete representation of the solution of loop closure equations is not available. Thus, even in absence of other constraints, we do not know a priori which are the feasible arrangements of the mobile elements in the world. Motion planning cannot be performed without this information, and sampling-based algorithms have to be extended to deal with this kind of constraints.

Figure 1 illustrates a motion planning problem under kinematic loop closure constraints. It is a "robotized" version of the piano mover's problem, where the piano is handled by mobile robotic manipulators. The top images represent the initial and the goal situations of the piano in the room. The bottom image is an snapshot of the piano's motion transported by the robots. Virtual closed kinematic chains are created when several manipulators grasp the piano.

The above example represents one of the applications of motion planning for closed-chain mechanisms: manipulation planning with several coordinated robots [Koga 94]. But there are many other applications. Indeed, closed kinematic chains appear in all the domains where motion planning algorithms are applied. In Robotics, parallel manipulators [Merlet 00] are themselves closed-chain structures in which the end-effector is connected to the base by at least two independent kinematic chains. In the area of mobile robots, legged robots [Boissonnat 00] also create kinematic loops whenever two or more legs contact with the ground. Other application in Robotics with increasing interest is motion planning for modular reconfigurable robots [Yim 03], which usually form closed chains. As well, constraints on motions of a robotic manipulator maintaining contact between the manipulated object and another object in the workspace [Xiao 01] can be translated as loop closure constraints. In the application of motion planning to graphic animation, closed kinematic chains are formed, for example, when human-like characters act in contact with each other. The design of prototypes involving closed kinematic chains with CAD/CAM system could clearly benefit from the incorporation of such extended algorithms for motion planning. In this same context, some manufacturing constraints [Garber 02] could be expressed as virtual loop closure constraints. In the industry, large objects are often transported by several handling devices, so that closure constraints are induced [Cortés 02a]. The conformational analysis of cyclic molecules [Gō 70] requires to deal with articulated closed-chain structures. Closure constraints are also imposed onto a molecular chain when

the relative position of certain atoms is fixed [Moult 86].

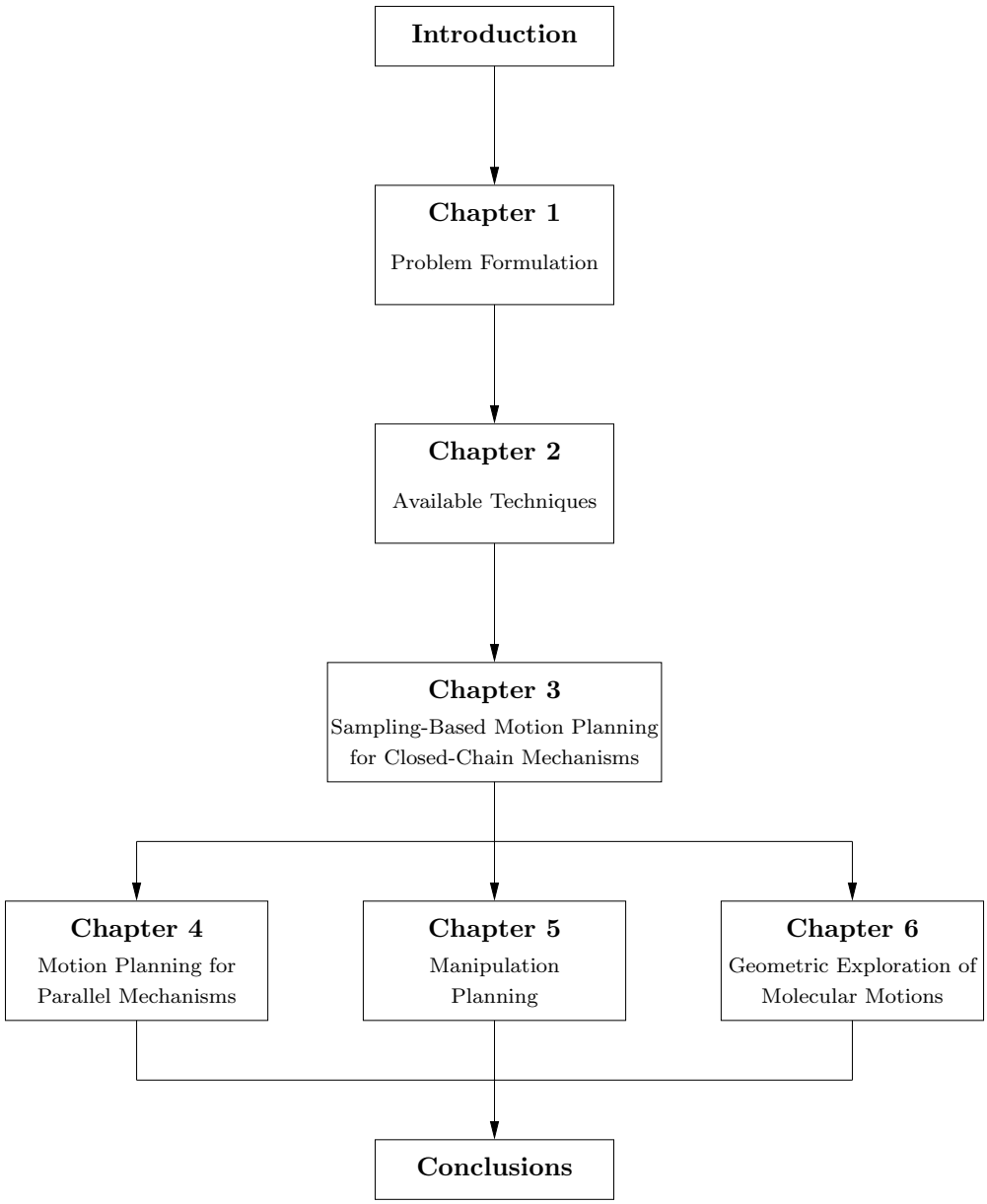
Despite the above commented interest, this instance of the motion planning problem raises difficult issues and has been rarely addressed. Only a few approaches [LaValle 99, Han 01], based on randomized sampling, are today potentially applicable to problems involving simple closed-chain mechanisms. Several issues remained to properly extend sampling-based planners to more complex mechanisms under closure constraints. Working on such extensions was the motivation of this thesis.

Contributions and Thesis Organization

The first part of this thesis contains our theoretical and technical work. The general problem is formulated in Chapter 1. We extend the formulation of the motion planning problem to the case of closed-chain mechanisms. Chapter 1 also explains some important notions used in the rest of the document. Then, Chapter 2 provides an overview of related works. We present a scope of techniques for solving motion planning problems and loop closure equations. Our approach is detailed in Chapter 3. We introduce sampling-based algorithms into our formulation of the motion planning problem in presence of kinematic closure constraints. The most important technical contribution in this thesis is also described in this chapter. We have developed a general and simple geometric algorithm called Random Loop Generator (RLG) for sampling random configurations satisfying loop closure constraints. RLG overcomes the most challenging aspect for extending sampling-based motion planning algorithms to closed chains. Some results shown in this chapter demonstrate the qualities of the approach.

The second part of this thesis deals with the different fields of application that we have investigated. In Chapter 4, we discuss the application of motion planning algorithms to parallel mechanisms. These mechanisms can be real structures, such as Gough-Stewart-like platforms, or virtual kinematic loops formed by several manipulators grasping the same object. Applied to parallel robots, our algorithms can help designers of these mechanisms, or can provide useful data for real-time trajectory planning. The same algorithms can also be used as components of manipulation planning techniques involving several coordinated robots. Chapter 5 regards manipulation planning for a robot and a movable object. An algorithm for planning the motions of a single-loop closed kinematic chain is used as a key component of a manipulation planning approach able to treat continuous sets in the definition of the manipulation task. This planner admits continuous sets for modeling both the possible grasps and the stable placements of the movable object, rather than discrete sets generally assumed by the existing planners. Then, intermediate grasp/ungrasp operations required to solve the problem are automatically identified. Finally, in Chapter 6, an interesting application is addressed out of the field of Robotics. We propose to use motion planing algorithms as efficient filters for the conformational exploration of protein loops. The structural analysis of protein loops is a very active area of research in Computational Biology. Our geometric algorithms can relieve conformational exploration approaches of a part of the heavy energetic treatment, and thus, improve their performance.

Interdependence of Chapters

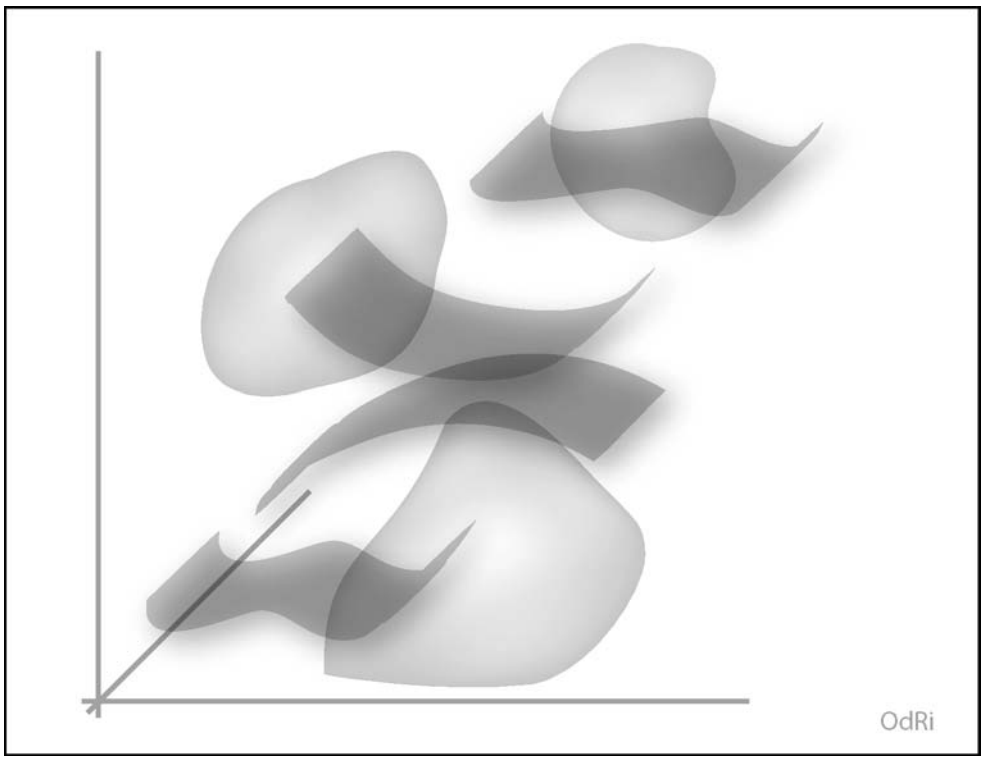


Part I

Theory & Techniques

Chapter 1

Problem Formulation



In most scientific areas, problems are formulated and solved for models of real systems. Motion planning problems treated in this document consider geometric models of physical objects. The first section of this chapter deals with the geometric modeling of rigid bodies and articulated mechanisms. Our work focus on the motion of mechanisms affected by kinematic closure constraints. Section 1.2 explains how these constraints are mathematically expressed on the geometric model of the mechanism.

The motion planning problem is formulated using the notion of configuration-space, presented in Section 1.3. Using this basic tool, the problem consists in exploring a space where kinematic constraints have a geometrical interpretation. In Section 1.4 we present different instances of the problem, centering the explanations on the issues tackled in next chapters.

1.1 Mechanical Modeling

This section presents notions related with the modeling of objects and mechanical structures. These notions are commonly used in Robotics, so readers initiated in the domain will not discover new definitions. However, for beginners or readers in other fields, we have considered interesting to briefly explain here the basic notions and to give pointers to books and articles for people who want to go deeply into some concepts.

1.1.1 Rigid Body Model

Elements of the world where motion takes place, often called *workspace* $\mathcal{W}$ in Robotics, are considered to be rigid objects (motion planning problems for deformable objects [Holleman 98, Lamiraux 01a] are not considered in this thesis). $\mathcal{W}$ is normally represented as the three-dimensional (3D) Euclidean space $\mathbb{R}^3$ and then a rigid object is a closed subset $\mathcal{O} \subseteq \mathbb{R}^3$. A rigid object can be represented in many different ways, using for example: polygonal and polyhedral models, semi-algebraic models or sets of triangles (see [Hoffmann 89, Mortenson 97] for explanations on some solid modeling methods). Characterizing $\mathcal{O}$ in $\mathcal{W}$ requires this representation and a set of parameters p defining its spatial location (i.e. position and orientation). For this, a Cartesian coordinate frame $F_{\mathcal{W}}$ is fixed in the world and another frame $F_{\mathcal{O}}$ is attached to the object. The relative position of these frames can be given by the vector $\{x_o, y_o, z_o\}$ of the coordinates of the origin of $F_{\mathcal{O}}$ in $F_{\mathcal{W}}$. The relative orientation can be expressed as a 3×3 matrix whose columns are the direction cosines of the axes of $F_{\mathcal{O}}$ in $F_{\mathcal{W}}$. Such a matrix belongs to the so called Special Orthogonal Group $SO(3)$. Thus, the position-orientation space in a 3D world can be defined as: $SE(3) = \mathbb{R}^3 \times SO(3)$. The parameterization of $\mathbb{R}^3$ is trivial: the elements of the vector of Cartesian coordinates. Different parameterizations of $SO(3)$ exist. They only require three parameters, for instance, consecutive rotations γ, β, α around the axes of the reference frame (i.e. parameters $\{$*roll, pitch, yaw*$\}$). Figure 1.1 illustrates this parameterization. Hence, p contains six independent parameters (three for the position and

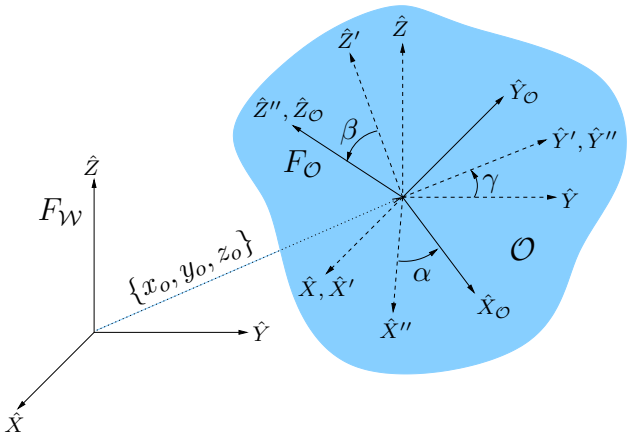


Figure 1.1: Six parameters $\{x_o, y_o, z_o, \gamma, \beta, \alpha\}$ defining the position and orientation of a solid.

three for the orientation).

The coordinates transformation from $F_{\mathcal{W}}$ to $F_{\mathcal{O}}$ can be expressed in a compact way using a homogeneous transformation matrix. This matrix is built from the parameters in p as follows:

$$ {}^{\mathcal{W}}T_{\mathcal{O}} = \begin{pmatrix} \cos\beta\cos\alpha & \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha & \cos\gamma\sin\beta\cos\alpha + \sin\gamma\sin\alpha & x_o \\ \cos\beta\sin\alpha & \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & y_o \\ -\sin\beta & \sin\gamma\cos\beta & \cos\gamma\cos\beta & z_o \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1.1} $$

More details about parameterizations of $SE(3)$ and about coordinate transformations can be found in basic textbooks on Robotics (e.g. [Hunt 78, Paul 81, Gorla 84, Craig 89, Sciavicco 00, Angeles 03]).

There are two kinds of object in the workspace: mobile objects and static objects. The former are the *links* of the articulated mechanism (defined next) that we call the *robot* [1]. The latter are the *obstacles* which compose what we call the *environment*. We designate the set of links by $\mathcal{A}$ and the set of obstacles by $\mathcal{B}$.

1.1.2 Articulated Mechanism

An articulated mechanism is a set of partially connected rigid bodies, called links in the mechanical terminology. We next explain some concepts related with such mobile systems.

[1] Even if the mobile system is composed of several robots and other mobile objects.

Degrees of Freedom: The term *degrees of freedom* (d.o.f.) in this context makes reference to the motion of objects. A rigid body freely moving in the world has six d.o.f., which are the parameters required for defining its location.

Joints: The connection between two neighboring links is called *joint*, J. A joint limits the relative mobility of the two objects. Thus, the number of d.o.f. between these links is reduced. These d.o.f. are called *joint variables*. Normally, the value of joint variables is limited by *mechanical stops*. Thus, each d.o.f. can take values within an interval bounded by the joint limits.

In Mechanics, the constrained motion between two connected links is normally characterized by two sliding surfaces. The term *lower pair* is used to describe joints making this kind of connection. There exist lower pair joints allowing two d.o.f. (*cylindrical joints* and *universal joints*) and three d.o.f. (*planar joints* and *spherical joints*). However, due to mechanical design considerations, articulated mechanisms are generally constructed with *revolute joints* and *prismatic joints* with only one d.o.f.. A detailed classification of joints is given in [Gorla 84].

Spatial Description: The 3D model of a mechanism is defined by the spatial location of all its rigid bodies. This information is given by the transform matrix between $F_{\mathcal{W}}$ and each one of the frames attached to links. The only condition for attaching a frame to a solid is that each point of the solid has fixed coordinates in it. Several conventions issued from the field of Robot Kinematics give rules for associating frames to connected links so that parameters describing the geometry of the mechanism are directly obtained from them.

We have adopted the widely used *modified Denavit-Hartenberg* (mDH) convention described in [Craig 89] for articulated mechanisms made up with revolute and prismatic joints. Let us call $F_{\mathcal{A}_{i-1}}$ and $F_{\mathcal{A}_i}$ the frames attached to two links $\mathcal{A}_{i-1}$ and $\mathcal{A}_i$ connected by a joint J_i. Following this convention, the location of $F_{\mathcal{A}_i}$ relative to $F_{\mathcal{A}_{i-1}}$ is given by the homogeneous transformation matrix:

$$ {}^{i-1}T_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i\sin\alpha_{i-1} \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i\cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1.2} $$

Two of the parameters in this matrix only depend on the geometry of $\mathcal{A}_{i-1}$ (i.e. they are constant): a_{i-1} is called the *link length* and α_{i-1} is the *link twist*. The other two parameters are related to the interconnection of the links: d_i, the *link offset*, and θ_i, the *joint angle*. These two parameters are variable depending on the joint type. For a revolute joint, θ_i is variable and d_i is a constant parameter, and the inverse for a prismatic joint. Thus, the only variable parameters in this relationship correspond to joint variables.

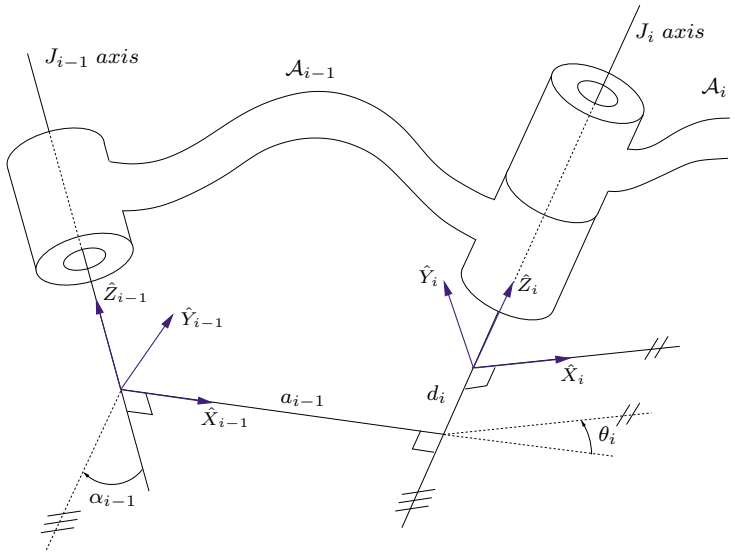


Figure 1.2: The mDH parameters defining the relative location of two links connected by a one-d.o.f. joint (following the convention in [Craig 89]).

When a mechanism is built with a joint J_i allowing n_{dof} d.o.f., it can be modeled as n_{dof} joints of one d.o.f. (revolute or prismatic) connecting $n_{dof} - 1$ fictive links of zero length. For instance, a spherical joints can be modeled by three consecutive revolute joints rotating around three orthogonal axes. We use the notation $J_{i.j}$, with $j = 1 \dots n_{dof}$, for the elementary joints issued from this decomposition. The matrix ${}^{i-1}T_i$ is obtained as the product of the n_{dof} individual transform matrices.

A sequence of $n+1$ links $\mathcal{A}_0 \dots \mathcal{A}_n$ connected by n joints $J_1 \dots J_n$ is called a *kinematic chain* ${}^1\mathcal{K}_n$. The location of frames attached to links $\mathcal{A}_1 \dots \mathcal{A}_n$ relative to the *base-frame* (attached to $\mathcal{A}_0$) is obtained from the sequence of local transformations. For instance, the transformation to the *end-frame* (attached to $\mathcal{A}_n$) is:

$$ {}^0T_n = {}^0T_1 {}^1T_2 \dots {}^{n-1}T_n \tag{1.3} $$

Therefore, only joint variables are required for defining the spatial arrangement of a kinematic chain. Since articulated mechanisms are composed of kinematic chains, the last sentence is true for any articulated structure.

Kinematic Diagram: Kinematic diagrams are often used to define the connectivity of links (also called topology) of articulated mechanism [Erdman 91]. Such diagrams are graphs of connections where nodes correspond to links and edges represent joints. Some authors have proposed rules to build such graphs and for the indexing of links and joints (e.g. [Hervé 78, Gosselin 88]).

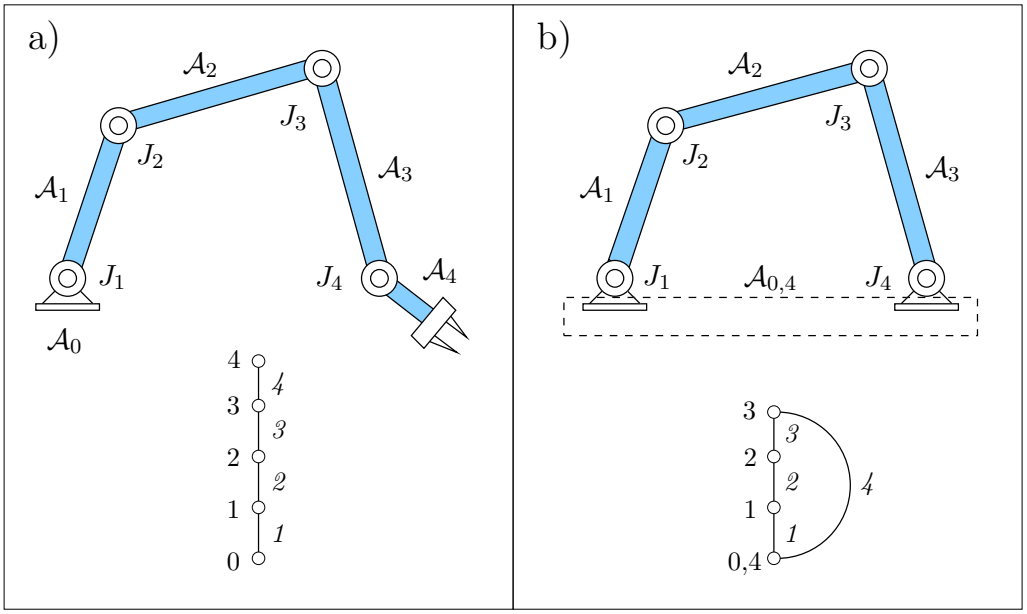


Figure 1.3: Planar mechanisms with revolute joints and their kinematic diagrams: (a) $4R$ planar manipulator (open kinematic chain); (b) the equivalent closed mechanisms, called $4R$ planar linkage.

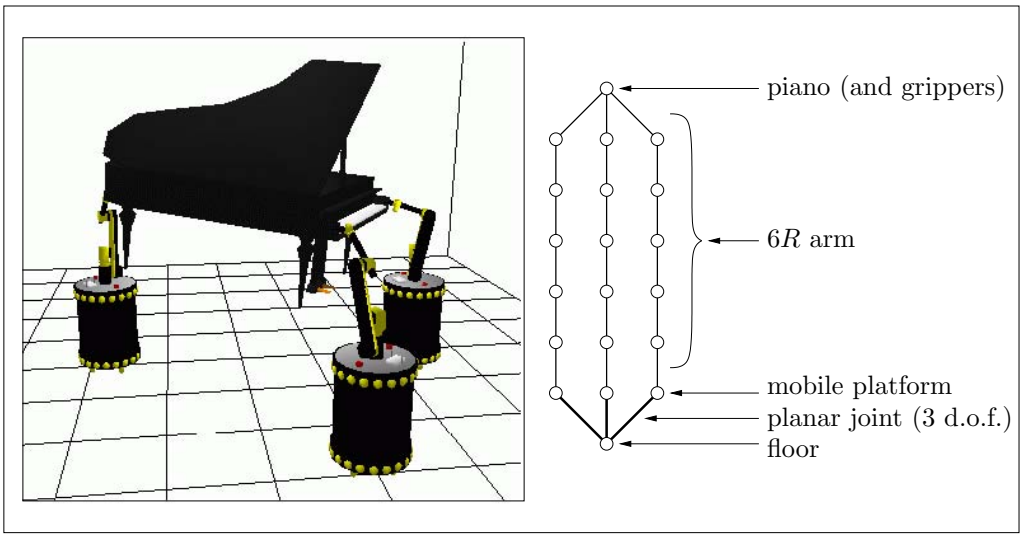


Figure 1.4: Model of complex mechanism composed of three mobile manipulators handling a piano. The kinematic diagram shows two loops.

A kinematic chain is said to be *closed* when it corresponds to a loop in the topological graph. Otherwise, the kinematic chain is called *open*. Figure 1.3 shows simple mechanisms with their topological graphs. An open chain (in Figure 1.3.a) becomes a closed mechanism (in Figure 1.3.b) if the relative location of the extreme links is fixed. In such case, the base-frames and the end-frame are attached to a same (fictive) solid $\mathcal{A}_{0,n}$. The kinematic diagram corresponding to the model of the mechanism composed by the three mobile manipulators handling a piano, that we presented in the introductory chapter, is shown in Figure 1.4. The model of the mobile platforms is simplified. The wheels are not considered, and contact with the floor is modeled by a planar joint. The kinematic diagram shows two loops which compose a multiple closed kinematic chain, called a *multi-loop*.

Mobility, Redundancy and Parallelism: We introduce here concepts that are mainly used to characterize robotic manipulators. However, they can be applied to any articulated device.

The *degree of mobility* M of an articulated mechanism is the number of independent variable parameters of the geometric model. For a mechanism with only open kinematic chains, all the joint variables are independent. Thus, M is equal to the sum of the number of d.o.f. allowed by joints. When the mechanism contains closed chains, *closure constraints* (that will be explained in Section 1.2) imply a relationship between joint variables. Several formulas have been proposed for determining the degree of mobility of general mechanisms from a topological analysis of connections (e.g. [Artobolevski 77, Hunt 78]). For instance, the next expression is called the *general mobility criterion* [Hunt 78]:

$$M = \sum_{i=1}^{n_{joint}} n_{dof_i} - d\, n_{loop} = m - d\, n_{loop} \tag{1.4}$$

with n_{dof_i} the number of d.o.f. of a joint J_i, n_{loop} the number of single loops and $d = 3$ for planar mechanisms (i.e. moving in 2D) or $d = 6$ for spatial mechanisms (i.e. moving in 3D). However, this simple formula (as any of the existent) may fail for mechanisms with special geometry (e.g. the Bennett's linkage [Bennett 03] and the molecule of cyclohexane [Crippen 92]) and the study must be completed by other geometric techniques (e.g. based on differential analysis [Angeles 88, Gosselin 88]).

If we apply the general mobility criterion to the planar open chain in Figure 1.3.a we obtain: $M = 4 - 3 \cdot 0 = 4$, that corresponds to the number of joint variables. However, $M = 1$ for the closed linkage in Figure 1.3.b, thus, only one joint variable is independent and the other three must satisfy equations issued from closure constraints. For the complex mechanism formed by the mobile manipulators and the piano (in Figure 1.4), $M = 27 - 6 \cdot 2 = 15$.

When talking about robot manipulators, the last link of the kinematic chain is called *end-effector*. The number E of absolute d.o.f. (i.e. with respect to the base-frame) of the end-effector is a very important characteristic of the manipulator, determining the kind of

tasks it can perform. When $M > E$, the manipulator has higher mobility than necessary. It is said to be *kinematically redundant*, with *degree of redundancy*: $\rho = M - E$. For a single-loop closed chain the end-effector has a fixed location with respect to the base, thus $E = 0$ and $\rho = M$.

The end-effector of the planar manipulator in Figure 1.3.a has three d.o.f. (three parameters are sufficient for defining the location of a solid in 2D). Thus this manipulator is redundant with $\rho = 1$. Note that this corresponds to the degree of mobility of the equivalent closed chain. If we consider that the piano is the "end-effector" of the complex system in Figure 1.4, then $E = 6$. Thus, this mechanism has a degree of redundancy $\rho = 9$.

Some authors (e.g. [Gosselin 88]) have also introduced the notion of *degree of parallelism* as:

$$P = \begin{cases} \frac{n_{loop}}{M-1} & \text{if } M \neq 1 \\ 1 & \text{if } M = 1 \text{ and } n_{loop} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{1.5}$$

An articulated mechanism is called *fully-parallel* if $P = 1$ and *partially-parallel* if $0 \leq P \leq 1$. Then, the closed chain in Figure 1.3.b can be seen as a single-loop closed chain or as an elementary fully-parallel mechanism because $M = 1$ and $n_{loop} = 1$. The mobile manipulators with the piano form a partially-parallel structure with $P = \frac{1}{7}$.

Singularities: The three above mentioned concepts (mobility, redundancy and parallelism) are global characteristics of an articulated mechanism. However, there are particular values called *critical points* of the set of joint variables for which these properties change. For these critical points, the mechanisms is *kinematically singular*. There are different kinds of singularities that yield to "abnormal" behaviors of articulated mechanisms. For serial manipulators (i.e. open-chain mechanisms), singular configurations are found by differential analysis of the joint variables. A classification of the different types of singular configurations for redundant serial manipulators is given in [Bedrossian 90]. For mechanisms involving closed kinematic chains (e.g. parallel mechanisms), the identification and the classification of singularities is more complex. There are singularities similar to those of serial manipulators, called *local singularities*, but also *architecture singularities* due to geometric particularities of the articulated structure [Gosselin 88, Charentus 90, Ma 91, Merlet 92].

The study of singularities has a great importance for the validation of trajectories from a point of view of controllability [Merlet 94, Sciavicco 00]. Nevertheless, singularities are not treated in this thesis, so we are not going into more details about them.

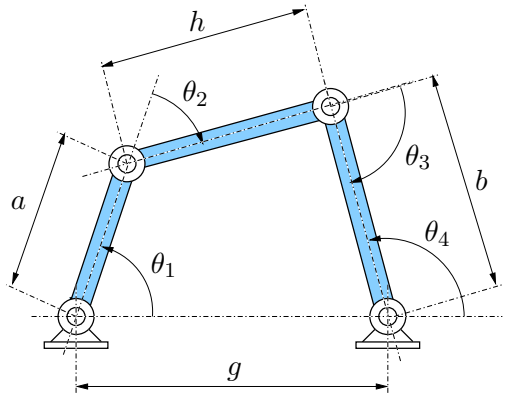


Figure 1.5: Dimensions characterizing a planar $4R$ linkage.

1.2 Loop Closure Constraints

When an articulated mechanism contains closed kinematic chains (i.e. there are loops in its kinematic diagram), joint variables involved in these chains are related by the so called *loop closure equations*. There are different ways to obtain such equations. For a $4R$ planar linkage, loop closure equations are obtained by a simple geometric analysis of the structure [McCarthy 00]. Next equations give the relationship between angles corresponding to joint variables for the linkage dimensions represented in Figure 1.5:

$$\theta_4(\theta_1) = \arctan\left(\frac{2ab\sin\theta_1}{2ab\cos\theta_1 - 2gb}\right) \pm \arccos\left(\frac{g^2 + a^2 + b^2 - h^2 - 2ag\cos\theta_1}{\sqrt{(2ab\cos\theta_1 - 2gb)^2 + (2ab\sin\theta_1)^2}}\right) + \pi \tag{1.6}$$

$$\theta_2(\theta_1, \theta_4) = \arctan\left(\frac{b\sin\theta_4 - a\sin\theta_1}{g + b\cos\theta_4 - a\cos\theta_1}\right) - \theta_1 \tag{1.7}$$

$$\theta_3(\theta_1, \theta_2, \theta_4) = \theta_1 - \theta_2 - \theta_4 + \pi \tag{1.8}$$

For a general closed kinematic chain ${}^1\mathcal{K}_n$, the loop closure equations can be obtained from the transformations between frames attached to links. Closure imposes the location of the $F_{\mathcal{A}_n}$ relative to $F_{\mathcal{A}_0}$ to be fixed. Then, the transform matrix ${}^0T_n$ is constant and known. ${}^0T_n$ can also be obtained from the sequence of local transformations (1.3). This matrix equality can be decomposed in scalar equations, leading to a system of 12 trigonometric equations. Normally, trigonometric equations are reduced to polynomial equations using the tangent-of-the-half-angle substitution [Kovács 93], although the drawback of this substitution (its singularity at $\pm\pi$) yields some scientists to search better alternatives.

When there are multi-loops, as in the example of Figure 1.4, the closure equations

are more complex. Joint variables involved in several loops must satisfy all the closure constraints simultaneously. Therefore, there is a dependence between the closure equations of the individual loops that compose the multi-loop. In the mentioned example, the location of the frame attached to the piano (relative to the floor) obtained traversing the kinematic chains corresponding to each one of the three mobile manipulators must be the same. The loop closure equations can be obtained from this multiple equality.

Hence, closure constraints are in general mathematically expressed as a system of multi-variable non-linear equations which relates the joint variables. This expression can be written as:

$$f(\mathcal{Q}\ ) = \mathbf{I} \tag{1.9}$$

where $\mathcal{Q}$ is the set of the joint variables, called the *joint-space*, and $\mathbf{I}$ is the identity displacement (i.e. the identity matrix when the equations are expressed in matrix notation).

The Inverse Kinematics Problem

In the case of open kinematic chains, loop closure equations must be solved to obtain the value of joint variables for a given location of the end-frame. That is called the *inverse kinematics problem* which is an important and widely studied problem in Robotics [Nielsen 97]. Some authors call this instance the *existence problem* making a distinction with other formulations, the *tracking problem* and the *point-to-point problem* [Siciliano 90, Ahuactzin 99], involving trajectories.

1.3 The Notion of Configuration-Space

The notion of *configuration-space*, $\mathcal{C}$, was introduced in the field of Robotics by Lozano-Pérez [Lozano-Pérez 83]. It is a key tool for the formulation of the motion planning problem (see Section 1.4). A *configuration* q is a minimal set of parameters defining the location of a mobile system in the world. $\mathcal{C}$ is the set of all the configurations q.

For a *free-flying robot* $\mathcal{A}$ (i.e. a rigid object whose motion is not limited by any kinematic or dynamic constraint), q specifies the location of $F_{\mathcal{A}}$ relative to $F_{\mathcal{W}}$. Therefore, $q = p$ and $\mathcal{C} = SE(3)$, as explained in Section 1.1. For a set of n free-flying objects $\mathcal{A}_1 \ldots \mathcal{A}_n$, the specification of the location of all the frames $F_{\mathcal{A}_i}$ is given by the vector: $q = \{p_1, \ldots, p_n\}$. Thus, the configuration-space would be the composite space:

$$\mathcal{C}\ = \underbrace{SE(3) \times \cdots \times SE(3)}_{n}$$

However, for an *articulated mechanism*, constraints imposed by joints reduce the number of variable parameters required to locate bodies. As explained in Section 1.1, these parameters correspond to the d.o.f. allowed by joints. A configuration q is then given by the

array of the joint variables, and the configuration-space corresponds to the joint-space, $\mathcal{Q}$. Let us assume that the mechanism does not contain loops and that joint have no mechanical stops. If r rotational d.o.f. and s translational d.o.f. are allowed by joints, then the configuration-space (considering the mechanism has a base-link fixed in the world) is:

$$\mathcal{C} = \mathcal{Q} = \underbrace{S^1 \times \cdots \times S^1}_{r} \times \underbrace{\mathbb{R} \times \cdots \times \mathbb{R}}_{s}$$

where S^1, the unit circle, can be parameterized as a single value θ in the interval $[0, 2\pi)$, with modulo 2π arithmetic. Topologically, $\mathcal{Q}$ is a m-dimensional smooth manifold, $m = r + s$, with a simple representation. For the $4R$ open chain in Figure 1.3.a, $\mathcal{Q}$ can be represented by a 4-torus. There is a one-to-one correspondence between each point in the 4-torus manifold and distinct configuration of this planar manipulator. See [Burdick 88] and [Latombe 91] for more details about the topology and representation of $\mathcal{Q}$.

The presence of joint limits does not affect the topological characteristic of $\mathcal{Q}$ (i.e. it is a smooth manifold) but modifies its geometric representation. For the $4R$ planar manipulator with joint limits, $\mathcal{Q}$ is represented by a 4-dimensional hypercube rather than a 4-torus.

If the articulated mechanism contains closed kinematic chains, then joint variables must satisfy loop closure constraints. Such constraints are normally expressed as a system of algebraic equations (1.9). Thus, $\mathcal{C}$ is an algebraic variety embedded in $\mathcal{Q}$. An algebraic variety is in general difficult to characterize topologically and even more difficult to represent [Mishra 97, Bochnak 98]. The representation is associated with the method used to solve the system of equations (see Section 2.2). Next, we comment some characteristics that are important for the formulation of the motion planning problem.

The Variety of the Configurations Satisfying Closure

Let us go back to the example of the $4R$ planar linkage (Figure 1.5). We have seen that this mechanism has one degree of mobility and we have written equations (1.6), (1.7) and (1.8) which give the value of joint variables θ_2, θ_3 and θ_4 as a function of θ_1. However, the closure constraint is not satisfied for all the values of θ_1. For a solution to exist, the argument of the arc-cosine function in equation (1.6) must be within the range $[-1, 1]$. This feasibility condition depends on the linkage dimensions. See for example Figure 1.6. For the linkage to the left (a), all values of θ_1 are feasible (we suppose that mechanical stops limit the allowed values of θ_1 to the interval $[0, \pi]$). On the contrary, an upper limit θ_1^{max} is determined by the intersection of circles with radii a and $b + h$ for the linkage to the right (b). It could also be a lower limit θ_1^{min} for a linkage whose dimensions make the circles with radii a and $b - h$ intersect. We call the *closure range* to the set of values of one joint variable satisfying the loop closure equations. In general, the closure range is composed of several real intervals that we call *closure intervals*.

Note that equation (1.6) gives two possible values of θ_2, and that this variable is in-

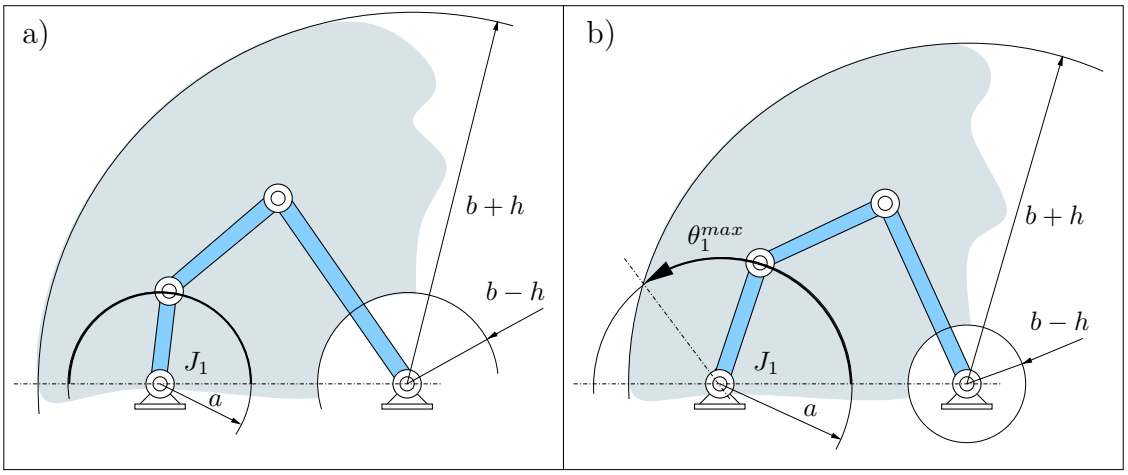


Figure 1.6: Two $4R$ linkages with different dimensions. The feasible range of values for θ_1 depends on these dimensions.

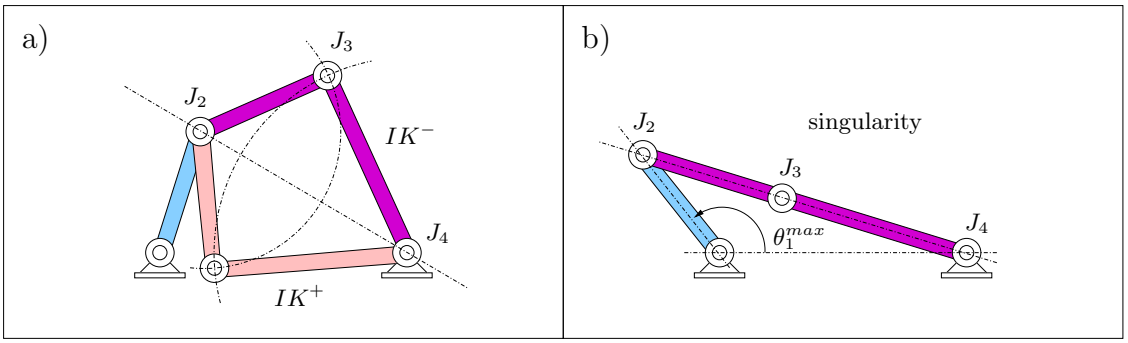


Figure 1.7: (a) The two configurations of the $4R$ linkage satisfying the closure constraint for a given value of the parameter θ_1. (b) For the value θ_1^{max}, the subchain involving joints J_2, J_3 and J_4 is at a singular configuration.

volved in equations (1.7) and (1.8). Therefore, two possible sets of joint angles $IK^+ = \{\theta_2^+, \theta_3^+, \theta_4^+\}$ and $IK^- = \{\theta_2^-, \theta_3^-, \theta_4^-\}$ satisfy the loop closure equations for a given valid value of θ_1. If θ_1 is swept through its closure range, the loop closure equations will generate two 1-dimensional manifolds embedded in $\mathcal{Q}$ (the 4-dimensional hypercube representing the joint-space of the mechanism without closure constraints). For the linkage in Figure 1.6.a, these manifolds remain separate for all the values of θ_1. If the subchain involving joints J_2, J_3 and J_4 is seen as a non-redundant $3R$ manipulator, the two manifolds physically correspond to motions where this subchain has a configuration "up elbow" (for IK^-) or "down elbow" (for IK^+ in Figure 1.7.a). In fact, IK^- and IK^+ correspond to the two possible solutions to the inverse kinematics problem for the $3R$ planar manipulator. However, for the linkage in Figure 1.6.b, the two manifolds meet at one point. This point corresponds to a singular configuration of the non-redundant $3R$ manipulator, illustrated in Figure 1.7.b.

Results of the above analysis of the $4R$ linkage can be extrapolated to more complex closed-chain mechanisms. Several works exist on the topological characterization of the variety of the configurations satisfying closure constraints (e.g. [Burdick 89, Thomas 93, Lück 93, Milgram 02]). Next, we summarize some of the conclusions obtained in these works which mostly focus on the characterization of the set of solutions to the inverse kinematics problem for redundant manipulators.

For redundant manipulators, there is an infinite number of configurations that lead to the same location of the end-effector (i.e. that solve the inverse kinematics problem). These configurations can be grouped into disjoint sets. Such sets are called *self-motion sets* because any trajectory inside one of them corresponds to a continuous motion of the manipulator maintaining a fixed location of the end-effector. In other words, the virtual closed kinematic chain created for a fixed location of the end-effector is not broken when the configuration changes inside a self-motion set. Self-motion sets can be seen as smooth hypersurfaces of dimension ρ intersecting themselves, with ρ the degree of redundancy of the manipulator (remember that $\rho = M$ if we consider the virtual closed kinematic chain). The stratification of these sets leads to n_{sm} ρ-dimensional manifolds $\mathcal{M}_i$ which can be connected through sets of lower dimension $\mathcal{S}_k$. The former are called *self-motion manifolds* and the latter are sets of singular configurations. Thus, the configuration-space of a closed-chain mechanism can be expressed as:

$$\mathcal{C} = \bigcup_{i}^{n_{sm}} \mathcal{M}_i \tag{1.10}$$

If we exclude the singular sets $\mathcal{S}_k$, then the sets of regular configurations in the self-motion manifolds $\mathcal{M}'_i$ are disjoint (i.e. $\mathcal{M}'_i \cap \mathcal{M}'_j = \emptyset$, for $i \neq j$).

Bounds on the number of disjoint self-motion sets have also been studied. The conclusion is that a redundant kinematic chain can have no more self-motion sets than the maximum number of inverse kinematic solutions of a non-redundant kinematic chain of the same class. Therefore, a planar closed kinematic chain can have at most two disjoint self-motion sets and a spatial chain a maximum of sixteen [2].

Some authors have worked on a more detailed characterization of the configuration-space of closed-chain mechanisms. Notions such as *mechanical aspects* and *assembly modes* have been introduced (e.g. [Borrel 86, Wenger 98, Chablat 98]). However, results are limited to particular classes of articulated structures. Therefore, the only general assertion is that the configuration-space $\mathcal{C}$ is the union of disjoint lower-dimensional sets embedded in the joint-space $\mathcal{Q}$. The topological structure of these sets (when including singular configurations) can be completely general. Indeed, it as been proved that there is a linkage whose configuration-space is homeomorphic to an arbitrary compact real algebraic variety with Euclidean topology [Jordan 99].

[2] The bound to the number of inverse kinematic solutions of a spatial $6R$ manipulator with general geometry was proved in [Primrose 86] and a solution method was provided in [Lee 88b, Lee 88a].

1.4 The Motion Planning Problem

The basic motion planning problem for a free-flying robot $\mathcal{A}$ in a static environment composed by n_{obst} obstacles $\mathcal{B}_1 \ldots \mathcal{B}_{n_{obst}}$ is formulated as follows: given an initial location and a goal location of $\mathcal{A}$ in $\mathcal{W}$, generate a *path* τ specifying a continuous sequence of locations of $\mathcal{A}$ avoiding *collisions* with the $\mathcal{B}_i$, or report failure if no such path exists. This problem is often referred to as the *piano mover's problem* [Schwartz 83, Schwartz 84].

Using the notion of configuration-space [Lozano-Pérez 83], $\mathcal{A}$ at a particular location becomes a point $q \in \mathcal{C}$. A path τ is then defined as a continuous map $\tau : [0,1] \longrightarrow \mathcal{C}$ such that $\tau(0) = q_{init}$ and $\tau(1) = q_{goal}$, with q_{init} the initial configuration and q_{goal} the goal configuration. For a solution to be feasible, the collision avoidance condition must be maintained along τ. The notion of *C-obstacles* is then introduced in order to characterize the subset of admissible (collision-free) configurations. Each obstacle $\mathcal{B}_i$ is mapped to $\mathcal{C}$ as a region:

$$\mathcal{C}_{\mathcal{B}_i} = \{q \in \mathcal{C} \mid \mathcal{A}(q) \cap \mathcal{B}_i \neq \emptyset\}$$

where $\mathcal{A}(q)$ is the subset of $\mathcal{W}$ occupied by $\mathcal{A}$ at a configuration q. Then, the *C-obstacle region* and the *free-space* are defined as:

$$\mathcal{C}_{obst} = \bigcup_{i \in [1, n_{obst}]} \mathcal{C}_{\mathcal{B}_i}$$

$$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obst}$$

Remark that $\mathcal{C}_{obst}$ is a closed subset of $\mathcal{C}$ and thus $\mathcal{C}_{free}$ is open. Therefore, configurations where $\mathcal{A}$ is in contact with the obstacles are contained in $\mathcal{C}_{obst}$. The *contact-space*, $\mathcal{C}_{contact}$, is another subset of $\mathcal{C}$ that can be defined in a simplistic way as the boundary of $\mathcal{C}_{free}$. τ is said to be a *free path* if it is completely contained in $\mathcal{C}_{free}$ and a *valid path* if it is contained in $\mathcal{C}_{valid} = \mathcal{C}_{free} \cup \mathcal{C}_{contact}$. In the following, we assume that only free paths are admissible.

Solving the basic motion planning problem formulated in this way consist in exploring the connectivity of the subset $\mathcal{C}_{free}$. A solution exists if and only if q_{init} and q_{goal} are in the same connected component of this subset. The key difficulty involved in such a formulation is the representation of the C-obstacles. How we will see in Chapter 2, some classes of motion planning algorithms evade the complexity of an explicit representation and implicitly represent them using 3D models of the robot and the obstacles combined with collision detection algorithms.

Reference books in the domain [Canny 88, Latombe 91, Laumond 98a, Gupta 98] give more details about this formulation of the basic motion planning problem and some of the extensions commented next.

1.4.1 Motion Planning under Holonomic Constraints

Our aim is to solve more complex instances of the motion planning problem than the basic one. Indeed, we tackle the *generalized mover's problem* [Reif 79], where the mobile system (the robot) is not a single rigid body but an articulated mechanism composed of n_{link} links $\mathcal{A}_1 \dots \mathcal{A}_{n_{link}}$. The relative motion of these links is constrained by the presence of joints. Restrictions imposed by joints are called ***holonomic equality constraints*** [Latombe 91] and, as we have seen in Section 1.3, they reduce the dimension of the configuration-space. Some authors seen them as a kind of *placement constraints* [Laumond 01] because they limit the set of points where objects can be *placed* (located) in the world.

The existence of several mobile objects modifies the definition of the C-obstacle region. In this case, two types of collisions may occur: those due to the intersection of a link $\mathcal{A}_i$ and an obstacle $\mathcal{B}_j$, and those due to the intersection of two links $\mathcal{A}_i$ and $\mathcal{A}_j$, called *self-collisions*. Thus, the C-obstacle region is defined as:

$$\mathcal{C}_{obst} = \left(\bigcup_{\substack{i\in[1,n_{link}]\\ j\in[1,n_{obst}]}} \{q \in \mathcal{C} \mid \mathcal{A}_i(q) \cap \mathcal{B}_j \neq \emptyset\} \right) \bigcup \left(\bigcup_{[i,j]\in CP} \{q \in \mathcal{C} \mid \mathcal{A}_i(q) \cap \mathcal{A}_j(q) \neq \emptyset\} \right) \tag{1.11}$$

where CP denotes the *collision pairs* $[i, j]$ such that $i \in [1, n_{link} - 1]$ and $j \in [i + 1, n_{link}]$.

Therefore, with a proper parameterization of $\mathcal{C}$ and the new definition of $\mathcal{C}_{obst}$, the formulation of the generalized problem is basically the same than for the piano mover's problem.

Closure Constraints

Closure constraints are a particular case of holonomic constraints that notably increase the difficulty of the motion planning problem. A first difficulty consists in computing $\mathcal{C}$, since a system of non-linear equations must be solved (see Section 1.2). Then, an extra difficulty arises from the topology of $\mathcal{C}$ that is not a smooth manifold as for mechanism without kinematic loops (see Section 1.3). In the general case, $\mathcal{C}$ is composed of several disjoint subsets (the self-motion sets) with a complex structure. Thus, the connectivity of $\mathcal{C}_{free}$ depends firstly on the connectivity of $\mathcal{C}$ and then on the presence of C-obstacles. Furthermore, self-motion sets may involve subsets of different dimension. Therefore, computing $\mathcal{C}_{free}$ requires to cover several subspaces.

Let us suppose a fictive example with three joint variables $\{\theta_1, \theta_2, \theta_3\}$ illustrated in Figure 1.8. Let us consider a function of the form $f(\theta_1, \theta_2, \theta_3) = 0$, representing a closure constraint. This function maps to several surfaces embedded in the joint-space $\mathcal{Q}$. Such surfaces are the different self-motion manifolds $\mathcal{M}_i$. In this example, $\mathcal{M}_1$ and $\mathcal{M}_2$ intersect at a singular set $\mathcal{S}$, forming a self-motion set. We have represented obstacles in the joint-space, that is why we call them $\mathcal{Q}_{obst}$. This *Q-obstacle region* is simply defined by replacing

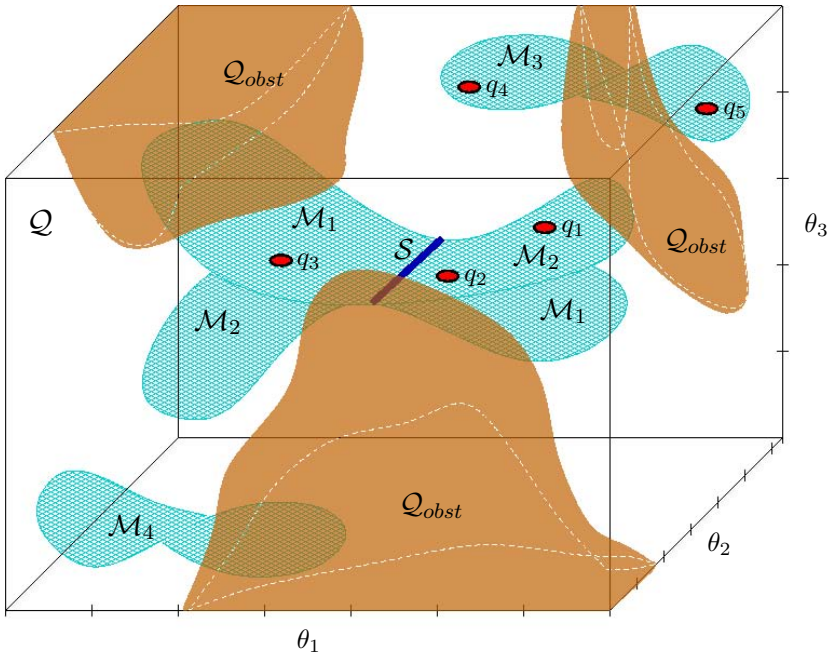


Figure 1.8: Illustration for the formulation of the motion planning problem in presence of closure constraints.

$\mathcal{C}$ by $\mathcal{Q}$ in equation (1.11). $\mathcal{C}_{free}$ is the intersection of $\mathcal{Q}_{free}$ with the different $\mathcal{M}_i$. Several situations can rise in motion planning queries within this example. The best case is a query for a path between q_1 and q_2. These configurations lie in the same self-motion manifold and on the same connected component of $\mathcal{C}_{free}$, thus there is a free path between them. A path is also feasible between q_1 and q_3, even if it contains singular configurations. However, for q_4 and q_5, the presence of obstacles makes these configurations cannot be connected by a free path. Finally, a case that does not appear for open kinematic chains can rise under closure constraints: q_1 and q_4 lie in the same connected component of $\mathcal{Q}_{free}$ but in different components of $\mathcal{C}$!

Manipulation Constraints

The *manipulation planning problem* addressed in Chapter 5 can be seen as a more complex instance of the motion planning problem. In its simplest version, this problem involves a robot, a *movable object* and the obstacles of the environment. The configuration-space is the composite space of the two mobile systems: the robot and the movable object. Particular holonomic constraints affect this problem. The movable object cannot move by itself. Either it is transported by the robot, or it stays at a stable placement. Such manipulation constraints determine subspaces of feasible configurations for the robot and the movable object. We will make a detailed formulation of this problem in Chapter 5.

Holonomic Inequality Constraints

Holonomic inequality constraints determine a subset of allowed locations of the robot having in general the same dimension than $\mathcal{C}$ [Latombe 91]. The restrictions imposed by the collision with obstacles and self-collisions are of this type. Indeed, any smooth function F with non-zero derivative applied onto the configuration parameters such that:

$$F(q) < 0 \quad \text{or} \quad F(q) \leq 0$$

implies a holonomic inequality constraint. For instance, if a weight is associated with each configuration, the admissible subset of $\mathcal{C}$ could be restricted to configuration over or under a certain value of this weight. In robotic applications, a weight obtained from the environment model could represent a suitability or a danger index for robots interacting with humans [Chatila 02] or moving in unknown areas [Chatila 95]. Another interesting example arises in the application of motion planning techniques to Structural Biology (see Chapter 6). Feasible conformations of a molecule (a conformation for a molecule is the equivalent to a configuration for a robot) are under an energetic level that depends on this conformation [Leach 96].

In general, we can define for a robot the subset $\mathcal{C}_{feas}$ of the configurations satisfying all the holonomic equality and inequality constraints. If the constraints are only of this type, an arbitrary path $\tau \subset \mathcal{C}_{feas}$ is feasible for the robot.

1.4.2 Motion Planning under Differential Constraints

Up to now, we have talked about constraints that reduce the subset of feasible configurations. Other motion constraints may affect the range of feasible paths (i.e. the possibility of connecting pairs of points in the same connected component of $\mathcal{C}_{feas}$). Mainly, constraints of this type involve velocities or accelerations. We call them *differential constraints*.

This thesis does not deal with differential constraints. The examples of applications that we will show stand in the most "popular" instance of the motion planning problem: with $\mathcal{C}_{feas} = \mathcal{C}_{free}$ and without differential constraints. Nevertheless, the formulation and the algorithms we present could be extended to handle some kinds of these constraints.

Kinematic *non-holonomic constraints* are the differential constraints that have mostly been treated within motion planning problems. Mathematically, non-holonomic constraints are non-integrable equations involving the first derivatives of the configuration parameters with respect to time (i.e. velocity parameters) [Laumond 86, Latombe 91, Laumond 98b]. Thus, some configuration parameters cannot variate independently. The most representative systems suffering non-holonomic constraints in robotic motion planning problems are car-like robots [Svestka 97b] and tractor-trailer robots [Lamiraux 97]. Let us imagine now than, in the example of the introductory chapter (see Figure 1), we replace one of the mobile robotic platforms, that are holonomic, by a car-like robot. Then, non-holonomic constraints should be treated in addition to closure constraints.

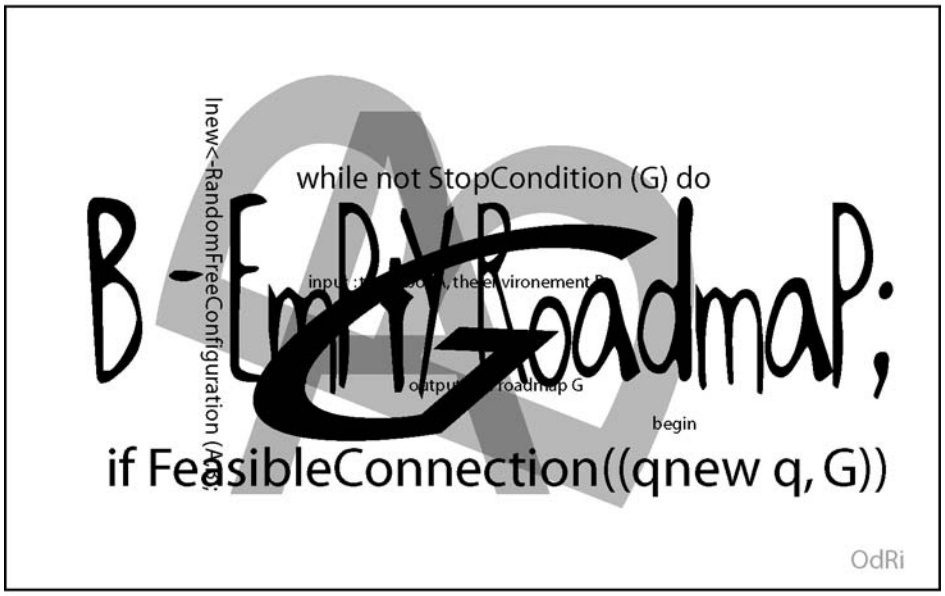
In general, the knowledge of the connectivity of $\mathcal{C}_{feas}$ is not sufficient to guarantee the existence of a feasible path between two given configurations if the mobile system is affected by non-holonomic constraints. However, last sentence is false under certain conditions. For *small-time controllable* systems [Laumond 98b, Laumond 01, Lamiraux 01b], any path in $\mathcal{C}_{feas}$ can be approximated by a finite sequence of feasible paths respecting non-holonomic constraints. For instance, a car moving forward and backward is small-time controllable while a car moving only forward is not.

Although kinematic non-holonomic constraints are much harder to treat than holonomic constraints, they can also be mapped in $\mathcal{C}$. Therefore, the motion planning problem can still be formulated and solved using the notion of configuration-space. In contrast, the representation of *dynamic constraints* in this space is less evident. Dynamic constraints are expressed on the second derivatives of the configuration parameters (i.e. they involve accelerations). Some dynamic constraints are integrable, and could be treated as kinematic constraints. However, generally they require to define a *state-space* including configuration parameters and velocity parameters. The motion planning problem considering dynamic constraints is referred to as the *kinodynamic planning problem* in literature [Donald 93, Fraichard 99, LaValle 01b], and it is not treated in this thesis.

There are many other constraints affecting motions in the real world that are more difficult to characterize by mathematic equations. For instance, when motion planning algorithms are applied to human-like characters, obtaining realistic animations require the incorporation of other techniques such as *motion capture* [Kuffner 99, Pettre 02]. With an insight into realistic motions, one could devise a piano mover's problem where the piano is transported by human-like robots.

Chapter 2

Available Techniques



This chapter aims to provide a state of the art on the available techniques for solving the problems treated in this thesis. We first present, in Section 2.1 and Section 2.2, an scope of techniques for solving motion planning problems and loop closure equations. Our approach to solve the motion planning problem for closed-chain mechanisms (described in Chapter 3) requires the combination of both kinds of techniques. In Section 2.3, we present the few existing approaches that tackle this problem.

2.1 Motion Planning Algorithms

The history of motion planning is quite recent. The first works appeared in the late 60's [Nilsson 69], and the active algorithmic development started in the 80's, with the notion of *configuration-space* [Lozano-Pérez 83]. During these two decades, a very large number of techniques have been proposed. Latombe's book [Latombe 91] provides an excellent overview of the progress on motion planning until the early 90's. A similar work collecting posterior theoretical advances and modern algorithms is not available yet. A couple of books that collect recent articles [Laumond 98a, Gupta 98] and the proceedings of the Workshop on the Algorithmic Foundations of Robotics [Goldberg 95, Laumond 97, Agarwal 98, Donald 01] provide a good overview and the essential references of the related works of the last decade.

The difficulty of motion planning depends on the complexity of the mobile system and on the intrinsic and extrinsic motion constraints. It has been proved that motion planning for an articulated mechanisms made of polyhedral links in a 3D environment is PSPACE-hard [Reif 79]. The applicability of exact algorithms (i.e. *complete* approaches that guarantee a solution if one exists and report failure when the problem is unsolvable) is limited by this computational complexity. The most efficient exact algorithm has time complexity exponential in the dimension of $\mathcal{C}$ [Canny 88].

Several approaches were subsequently proposed aiming to overcome complexity and implementation inconveniences of exact methods. For this, some continuous quantities in the problem definition, such as object dimensions or configuration parameters, are discretized. Algorithms based on an approximated cell decomposition of the free configuration-space (see Chapter 6 in [Latombe 91]) are *resolution complete*: they are complete for a given discretization size. Other approaches use a grid to quantize $\mathcal{C}$ and perform a search process over this grid. Efficient heuristic algorithms have been developed based on the *potential field approach* [Khatib 86] to carry out this search (see Chapter 7 in [Latombe 91]). Such algorithms are able to perform very fast, but get easily trapped at local minima. The design of the potential function is a critical point, and difficult for high-dimensional spaces. These approaches using cells or grids are applicable in practice to mobile systems involving only a few variables. The number of cells or grid points becomes enormous for attaining acceptable resolution in high dimension.

Practical planners, more general than the previous ones by satisfying a weaker form of completeness, appeared in the 90's [Barraquand 91, Kavraki 96, LaValle 98]. These approaches use randomization to treat the high dimensionality of $\mathcal{C}$ (see [Motwani 95] as one of the first textbooks on randomized algorithms). The term *probabilistically complete* was introduced to characterize these sampling-based algorithms, able to find a solution if sufficient running time is given. Up to date, the development of algorithms for motion planning has continued in this direction. Work has been done to analyze the behavior of randomized planners [Kavraki 98, Svestka 98, Laumond 01] and to define sampling strategies allowing to solve difficult problems [Hsu 98, Amato 98, Boor 99]. Currently, some scientists are taking a direction toward "derandomizing" some sampling-based approaches for motion planning [LaValle 03b], aiming to gain insight for the control of algorithms and to improve coverage properties. Section 2.1.1 deals about these (randomized and deterministic) sampling-based approaches. Then, two particular approaches, PRM and RRT, are detailed in Section 2.1.2. These are the approaches we have chosen to extend for the treatment of closed-chain mechanism (explained in Chapter3).

2.1.1 Sampling-Based Approaches

Sampling-based algorithms, developed in the last decade, have demonstrated their efficacy for solving motion planning problems in high-dimensional spaces. They capture the connectivity of the collision-free regions of the configuration-space $\mathcal{C}_{free}$ without requiring to explicitly compute this subset. The aim of this section is not to make an exhaustive survey of all the available techniques, but a simple overview which gives an idea on the last tendencies in motion planning. We present a classification (partially inspired by [LaValle 03a]) of some of the approaches which have been particularly well accepted in the Robotics community. The main ideas of each approach are briefly described and we point to papers which provide full explanations.

Sampling-based approaches can be grouped in two main families: those using sampling techniques for constructing a roadmap in $\mathcal{C}_{free}$ [Kavraki 96] and those using sampling within incremental search methods for exploring $\mathcal{C}_{free}$ looking for a particular path [Barraquand 91, LaValle 98]. The choice mainly depends on the application. Roadmap methods are more suitable when several motion planning queries involving the same system moving in a static environment must be solved. Computing time is spent in a preprocessing phase and then planning queries can be solved in real-time. They are called *multiple-query methods*, although some roadmap-based algorithms have been developed to efficiently solve particular queries [Bohlin 00]. Incremental search methods are used for solving single motion planning queries. *Single-query methods* are in general faster since they need not preprocessing. However, as they focus on solving a particular problem, the processed information is less appropriate for later use. A good example of application for single-query methods are assembly problems [Chang 95], where one must determine whether there exist a path to remove a part from an assembly for maintenance. Sometimes it is interesting to combine both kinds of methods. For instance, when only slight modifi-

cations are produced in the environment, a multiple-query method can be used at a global level and then single-query methods can rapidly solve local problems arisen from these slight changes. In the approach for manipulation planning, explained in Chapter 5, we use such a combined technique. As recently proposed in [Akinc 03], the use of single-query techniques within a multiple-query motion planning framework can have other advantages such as the efficient parallelization of the process.

Roadmap Methods

A roadmap method consists of two phases. First, in the *roadmap construction phase* a network of one-dimensional curves capturing the connectivity of $\mathcal{C}_{free}$ is computed. Once the roadmap has been constructed, a motion planning query can be answered by connecting the initial and goal configurations, q_{init} and q_{goal}, to points in the roadmap and searching for a path (a concatenation of curves) between them. This is called the *query phase*. The search in this second phase can be achieved by classical AI algorithms such as the A^*. Additionally, a third phase is usually carried out for smoothing the resulting path.

The ***Probabilistic RoadMap*** (**PRM**), developed simultaneously at Stanford and Utrecht [Overmars 95, Kavraki 95a, Kavraki 96, Svestka 97a], is the principal sampling-based approaches for motion planing. The simplest algorithm inspired in the PRM approach builds a graph (the roadmap) whose nodes are randomly sampled configurations lying in $\mathcal{C}_{free}$ and whose edges are short collision-free *local paths* (i.e. simple paths generated by a local method) linking "nearby" nodes. Samples and local paths are checked for collisions using effective collision detection algorithms [Jiménez 98, Lin 03]. Because of the random configuration sampling, the algorithms based on this approach are probabilistically complete under weak conditions for local paths [Svestka 97a].

The main difficulty with an uniform random sampling of $\mathcal{C}$ is find connections through some "critical" regions of $\mathcal{C}_{free}$. This difficulty is referred to as the *narrow passage problem*, and is common to randomized algorithms. Some algorithms try to increase the samples in such narrow passages that arise because of C-obstacles. One possibility is to admit some samples lying in $\mathcal{C}_{obst}$ and to "push" them toward $\mathcal{C}_{free}$ [Hsu 98]. The same idea has also been used in the other direction, pushing samples in $\mathcal{C}_{free}$ toward $\mathcal{C}_{obst}$, in the *Obstacle-Based PRM* approach [Amato 98]. The *Gaussian sampling strategy* proposed in [Boor 99] generates samples by close pairs and keep only those for which one of the configuration lies in $\mathcal{C}_{obst}$. Other approaches sample the generalized Voronoi diagram (also called the *medial-axis*) of $\mathcal{C}_{free}$ [Wilmarth 99, Lien 03] by retracting randomly sampled configurations using approximate values of clearance and penetration depth. In [Holleman 00], a PRM-based approach is proposed that samples at the medial-axis of the workspace. This approach may be interesting for free-flying robots; however, it is unclear how to generalize this sampling technique to more general articulated robots.

Another drawback of random sampling is that an analysis for devising control parameters of algorithms is very difficult. The ***Visibility-PRM*** approach described in [Nissoux 99, Siméon 00] allows a simple on-line estimation of the amount of $\mathcal{C}_{free}$ encoded in the roadmap. This information is useful for defining stop conditions in algorithms. The principle of this approach is to add samples as nodes of the roadmap only if they serve to link different connected components or if they are not "visible" by the other nodes. The number of consecutive failures when trying to add a new node is a parameter for estimating coverage [Siméon 00]. Another advantage of this technique is that the connectivity of possibly complex spaces is captured into a small data structure.

Being aware of the heavy cost of collision detection (about 90% of the computing time for basic PRM algorithms [van Geem 01b]), the ***Lazy-PRM*** approach [Bohlin 00] builds the roadmap initially ignoring the presence of obstacles and delays the collision checking to the query phase. Thus, the aim of this planner is to solve single queries efficiently while partially keeping the philosophy of roadmap approaches. For the solution of a particular query, paths in the roadmap are iteratively searched and checked for collisions until a valid sequence of local paths is obtained or all the possibilities have been tried. In this last case, more nodes may need to be added to the roadmap. The validity test on local paths is progressive, using a technique also proposed in [Nielsen 00]. This technique consists in incrementally increasing the resolution for collision detection in such a way that invalid segments can be rapidly detected and removed while valid ones are labeled with and index indicating the resolution at which they are already been tested.

The term ***Quasi-random RoadMap*** (**QRM**) is introduced in [Branicky 01] to designate a family of PRM-like algorithms using quasi-random sampling techniques. Quasi-random sets of numbers (e.g. *Hammersley* and *Halton* point sets [Neiderreiter 92]) present better properties for coverage than pseudo-random numbers normally used in PRM-based planners. Best performance of planners using such a configuration sampling has been experimentally proved in spaces of up to ten dimensions [Branicky 01, Geraerts 02, LaValle 03b]. Also in [Branicky 01], another approach is proposed that goes further in the "derandomization" of PRM-based algorithms. It is called ***Lattice RoadMap*** (**LRM**) in [LaValle 03b]. This approach uses lattice points to generate samples. Lattices can be defined with good coverage properties [Lindemann 04], but their main advantage is that they provide implicit neighborhood structure which can be exploited in planning algorithms. The interest of replacing the probabilistic roadmap constructed in the first phase of the Lazy-PRM approach by a family of quasi-random embedded lattices is discussed in [Branicky 01], and comparative results are promising. However, as pointed out in [LaValle 03b], such grid-based sampling techniques are interesting in practice to problems in spaces with a restricted number of dimensions (say ten). The reason is that the neighborhood of a grid point increases exponentially with the dimension of $\mathcal{C}$. Thus, non-lattice sampling techniques, using pseudo-random or quasi-random sequences, are today the only alternative for constructing roadmaps in very-high-dimensional spaces.

Incremental Search Methods

Incremental search methods applied to motion planning explore the collision-free regions of the configuration-space trying to find a feasible path between to given points, q_{init} and q_{goal}. Normally, the exploration is biased to solve this particular planning query and not to obtain information about the whole space. Most of the algorithms construct trees whose nodes are configurations computed during exploration. The search can be performed in only one direction or in the two directions. *Unidirectional methods* develop a single tree from one of the two given configurations until the other configuration is reached. *Bidirectional methods* construct one tree from q_{init} and another from q_{goal}. The solution is found when the two trees meet at a point. Choosing a unidirectional or a bidirectional search mainly depends on the characteristics of problem to be solved. For example, if the robot is highly constrained around q_{init} and quite free to move around q_{goal}, it will be more efficient to build only a tree rooted at q_{init} and to try connections of nodes to q_{goal}.

The ***Randomized Path Planner*** (**RPP**) [Barraquand 91] is recognized in most of the surveys on motion planning as the first randomized algorithm. This planner is based on a randomized potential field approach. It searches for a path from the initial to the goal configuration by following the negative gradient of an artificial potential field constructed over $\mathcal{C}$ and uses random walks to escape local minima of the potential function. The RPP is probabilistically complete and has provided very good results. However, it is now well known that this planner is hindered by narrow passage problems [Kavraki 96].

The ***Ariadne's Clew Algorithm*** (**ACA**) [Bessière 93] interleaves two optimization algorithms. A genetic algorithm called EXPLORE is used to generate landmarks (nodes of a search tree rooted at q_{init}) trying to optimize their distribution over $\mathcal{C}_{free}$ by maximizing the distance between them. After the generation of a new landmark, another algorithm, SEARCH, looks for its connection with the goal configuration. The drawback attributed to this approach is that the optimization process carried out by EXPLORE is costly and may require some parameter tuning.

The ***Expansive-Space Tree*** (**EST**) planner presented in [Hsu 97, Hsu 00] shares some ideas with PRM approaches, hoverer it tries to sample only the portion of $\mathcal{C}$ that is relevant for a particular planning query, avoiding the cost of precomputing a roadmap for the whole free-space. The algorithm iteratively executes two steps, *expansion* and *connection*, in a similar way than ACA. This is a bidirectional planner (i.e. two trees are built), although a unidirectional version is also implementable. The choice of the node q to be expanded at one iteration is biased in order to increase samples in few explored regions. Configurations are randomly sampled in a predetermined neighborhood of q. Samples with good properties for the coverage of $\mathcal{C}$ and with a valid link (i.e. collision-free local path) to q are kept as new nodes. In the connection step, valid links between nodes of both trees which are closer than a given distance in $\mathcal{C}$ are tested.

Similarly to the Lazy-PRM planner, the ***SBL-PRM*** [Sánchez 03] is another approach issued from the probabilistic roadmap framework for single planning queries. However, in this case, a roadmap is not built trying to cover the whole configuration-spaces. This approach aims to better exploit the delayed collision checking by combining it with an adaptive sampling technique similar to the one used by EST. The algorithm incrementally constructs two trees of collision-free configurations rooted at q_{init} and q_{goal}, but does not tests the validity of connections. When the trees meet, the sequence of local paths connecting the query configurations is checked for collisions. The validity test is iterated along the whole path, using a a similar technique than [Nielsen 00] and [Bohlin 00]. When a collision is detected, the corresponding segment is removed and the construction process goes on until the trees meet again. This algorithm has been applied to problems involving several manipulator arms operating in the same workspace [Sánchez 02].

The ***Rapidly-exploring Random Tree*** (**RRT**) approach, introduced in [LaValle 98], has become the most popular single-query motion planner in the last years. RRT-based algorithms where first developed for non-holonomic and kinodynamic planning problems [LaValle 01b] where the space to be explored is the state-space (i.e. a generalization of $\mathcal{C}$ involving time). However, tailored algorithms for problems without differential constraints (i.e. which can be formulated in $\mathcal{C}$) have also been developed based on the RRT approach [Kuffner 00, LaValle 01c]. As the above approaches, RRT-based algorithms combine a construction and a connection phase. For building a tree, a configuration q is randomly sampled and the nearest node in the tree (given a distance metric in $\mathcal{C}$) is expanded toward q. In the basic RRT algorithm (which we refer to as RRT-Extend), a single expansion step of fixed distance is performed. In a more greedy variant, RRT-Connect [Kuffner 00], the expansion step is iterated while keeping feasibility constraints (e.g. no collision exists). As explained in the referred articles, the probability that a node is selected for expansion is proportional to the area of its Voronoi region. This biases the exploration toward unexplored portions of the space. The approach can be used for unidirectional or bidirectional exploration. For the later case, several combinations of more or less greedy functions have been proposed in [Kuffner 00, LaValle 01c].

Like for the PRM approach, recent works on the RRT approach have a tendency toward "derandomized" algorithms [Lindemann 03]. However, the utility of randomization in the RRT planners is more subtle than in PRM planners. If random samples were simply replaced by deterministic ones, the RRT approach will loose the essence of its exploration strategy: the Voronoi biasing. Recently, features of RRT-based algorithms and *dynamic programming algorithms* [Bellman 57, Barraquand 93, LaValle 01a] have been compared in [LaValle 02]. The author identifies complementary advantages and drawbacks of these techniques and discerns new directions for future research. First works on the development of such improved RRT-based planners have been published [Cheng 01, Cheng 02], but several questions remain open.

2.1.2 PRM and RRT Algorithms

The most successful motion planning approaches in the last years are PRM for multiple-query problems and RRT for single-query problems. This success is mainly due to their great efficiency, reliable performance, conceptual simplicity and applicability to many different types of problems. Indeed, algorithms based on these approaches have been used to solve more complex instances of the motion planning problem than the classic mover's problem. In Robotics, they have been extended to: non-holonomic planning [Svestka 97b, Sekhavat 98, Lamiraux 01b], kinodynamic planning [LaValle 01b, Hsu 02], sensor-based motion planning [Yu 00], motion planning under closure constraints (see Section 2.3 and Chapters 3 and 4) and manipulation planning [Nielsen 00, Siméon 03] (see also Chapter 5). They have also been adopted to solve related problems in other areas, as for example: maintenance problems in industrial logistics [Siméon 01c] or transportation problems [Lamiraux 03], animation of characters in computer graphics [Kuffner 00, Pettre 02], or computational Biology [Finn 98, LaValle 00, Apaydin 02, Amato 02, Cortés 03b] (see also Chapter 6). In this section, we briefly explain the main features of these two approaches.

Ingredients

The design of PRM-based and RRT-based motion planning algorithms requires some ingredients we describe next. Several options are available for some of these basic elements. The behavior of the algorithm is associated with the selection of these ingredients. Some choices are more appropriate than others depending on particularities of the problem to be solved. This fact could be exploited by a meta-planner, as proposed in [Dale 01], which switches between algorithms with different settings for regions of the configuration-space with different characteristics.

A recent work published in [Geraerts 02], provides a comparative study of variants of the basic-PRM algorithm obtained by selecting different options for ingredients. This study yields very useful conclusions for the design of PRM planners. However, the determination of this analysis must be relativized since studied examples only involve free-flying robots (i.e. $\mathcal{C}$ is a 6-dimensional space). As far as we know, a similar study has not been made for RRT-based algorithms. However, as argued in [Kuffner 00, LaValle 01c], the choice between different expansion heuristics conditions their adequateness to solve different problems.

The software platform **Move3D** [Siméon 01b] in which we develop motion planning algorithms integrates several options for each one of next basic elements. Thus, experiments can be carried out in order to determine good settings of algorithms for each family of problems.

Configuration Sampling: The main item of both PRM-based and RRT-based motion planning algorithms is configuration sampling. The former sample the configuration-space in order to add nodes to the roadmap and the later use an incremental search approach biased by the sampled configurations.

The basic PRM approach applies a uniform random sampling. Some of the above commented variants apply other strategies, either devised to solve problems with narrow passages [Amato 98, Boor 99, Wilmarth 99, Holleman 00] or looking for better coverage properties [Branicky 01, LaValle 03b].

For RRT planners, the only effective choice that has been proposed for configuration sampling is a uniform random sampling, although deterministic sampling strategies are currently being investigated [LaValle 02, Lindemann 03].

Distance Metric: Distance metrics are used to determine the adjacency of configurations. The selection of a good distance metric is critical for the performance of PRM-based and RRT-based algorithms since this information is normally used to decide which connections or expansions must be tested (see next paragraph). The ideal metric should consider the existence of motion constraints (i.e. joint limits, obstacles, differential constraints, etc.). Obviously, designing such a metric remains a very difficult issue. In general, a simple scaled-Euclidean metric in $\mathcal{C}$ provides good results for holonomic planning [Amato 00]. However, under differential constraints, the design of an appropriate metric in $\mathcal{C}$ is quite more complex [Laumond 98b, LaValle 02].

Connection Strategy: The connection strategy consists in the selection of nodes for testing feasible links by local paths in the case of PRM-based algorithms, or the choice of the node to be expanded in the RRT approach.

The PRM approach tests the connectivity of a new sampled configuration to nodes in the roadmap. Since each test for a feasible connection is very expensive, it is important to reduce the number of tests as much as possible. Thus, only a small set of "best" nodes is selected. Normally, the nearest nodes of each connected component are chosen. Once a feasible connection is found, connections with other nodes in the same component need not be tried. In general, these extra tests increase the computational cost without improving the roadmap construction process [Overmars 95]. Note that the such a selection of nodes requires a distance metric to be defined. Other heuristics that do not use a metric can also be considered. A strategy choosing as best nodes those receiving the highest number of edges is also proposed in [Nissoux 99].

RRT-based algorithms select a node in the tree to be expanded toward a sampled configuration. The choice of the nearest neighbor seems the most reasonable strategy, for with efficient search algorithms have been proposed (e.g. [Atramentov 02]). Due to this strategy, the behavior of the planner strongly depends on the distance metric. Practical metrics do not consider the existence of obstacles (i.e. they measure a distance in $\mathcal{C}$ and

not in $\mathcal{C}_{free}$) or other feasibility constraints. Reducing metric sensibility of RRT-based planner has been investigated [Cheng 01, Cheng 02, LaValle 02], but effort must still be made in this direction.

Steering Method: The *steering method* is a procedure computing a feasible path in $\mathcal{C}$ between any two configurations considering some intrinsic motion constraints of the mechanical system (it considers constraints imposed by joints and non-holonomic constraints but ignores collision avoidance). In the PRM literature, the steering method is often called local method because it is the basic tool for generating the local paths. The choice of a steering method for a given system is in general non unique. This choice may affect the combinatorial complexity of the algorithm [Laumond 01]. For mobile systems that are not affected by differential constraints, devising steering methods is in general an easy task. In most cases, a simple straight-line segment in $\mathcal{C}$ is an admissible connection. However, under differential constraints the design of adequate steering methods is a difficult problem [Svestka 97b, Sekhavat 98, Laumond 98b]. Indeed, techniques are only available for some classes of systems depending on controllability issues.

Collision Detection: PRM and RRT approaches do not require an explicit representation of $\mathcal{C}_{free}$. They call a procedure able to determine if the sampled configurations and the connecting paths lie in $\mathcal{C}_{free}$ or not. Such a collision checking is made on three-dimensional models of the robot and the environment.

PRM-based algorithms need to test for collisions each new sampled configurations and each computed local paths. A variety of *static collision detection* algorithms are available for checking configurations [Jiménez 98, Lin 03]. For checking a local path, the simplest way is to discretize it with uniform resolution and to call the static checker iteratively. On the one hand, high resolution is necessary in order to guarantee the validity of the path; on the other, calls to the collision detection algorithms are very time consuming, so that the number of calls should be restricted for increasing efficiency. More efficient *dynamic collision detection* algorithms can be designed adapting the resolution depending on the distance to C-obstacles [Nissoux 99, van Geem 01b, Schwarzer 02]. These algorithms relate the distance between objects in the workspace to lengths of walks on local paths. Several works show that a test using a dichotomic division of the local path generally gives a better performance than an incremental test for both uniform and non-uniform resolution checking [van Geem 01b, Geraerts 02].

RRT-based algorithms perform collision checking while trying to expand a node of the tree. This expansion process can also benefit from efficient dynamic checkers with adaptive resolution, in particular when using the RRT-Connect variant, which generates longer local paths to be tested compared to the RRT-Extend variant.

Algorithm 2.1: Construct_PRM

```
input  : the robot A, the environment B
output : the roadmap G
begin
    G ← EmptyRoadmap;
    while not StopCondition(G) do
        q_new ← RandomFreeConfiguration(A, B);
        AddNewNode(q_new, G);
        L_best ← ListBestNodes(q_new, G);
        for all q ∈ L_best do
            if not InSameComponent(q_new, q, G) then
                if FeasibleConnection(q_new, q) then
                    AddNewEdge(q_new, q, G);
end
```

Algorithm 2.2: Construct_One_RRT

```
input  : the robot A, the environment B, q_root
output : the tree G
begin
    G ← InitTree(q_root);
    while not StopCondition(G) do
        q_rand ← RandomConfiguration(A, B);
        q_near ← NearestNeighbor(q_rand, G);
        q_feas ← q_near;
        state ← OK;
        while state = OK do
            q_step ← MakeStep(q_feas, q_rand);
            if FeasibleConfiguration(q_step) then q_feas ← q_step;
            else state ← FAIL;
        if not TooSimilarConfigurations(q_near, q_feas) then
            q_new ← IntermediateConfiguration(q_near, q_feas);
            AddNewNode(q_new, G);
            AddNewEdge(q_new, q_near, G);
end
```

Basic Algorithms

Algorithms 2.1 gives pseudocode for the roadmap construction phase of the basic-PRM approach. It combines the above explained ingredients so that different behaviors of the planner are obtained with the different settings. Note that this algorithm could be completed with an *expansion step* as proposed in [Kavraki 96].

Algorithms 2.2 builds a search tree in the configuration-space based on the unidirectional RRT approach, originally formulated in the state-space. In such a tailored version, the function MakeStep performs a walk on a local path connecting q_{rand} and q_{near}. This path is computed by a steering method, like in the PRM approach. The presented algorithm corresponds to the RRT-Connect variant.

Some Properties

The analysis of sampling-based planners is a difficult problem. Main subjects of analysis involve completeness (i.e. the capacity to find a solution whenever one exists), coverage and connectivity properties (i.e. the amount of the search-space encoded in the graph structure) and computational complexity.

Concerning the PRM approach, proofs for probabilistic completeness are given in [Kavraki 95b, Kavraki 98]. A probabilistically complete planner will find a solution path, if one exists, in bounded time with high probability. In [Svestka 97a], detailed demonstrations of probabilistic completeness are extended to more general kinds of mobile systems, affected by non-holonomic constraints.

The notion of *ϵ-goodness* is introduced in [Barraquand 97] for the analysis of coverage properties. $\mathcal{C}_{free}$ is ϵ-good if the volume of the visibility set (i.e. the set of configurations attainable by the steering method) of any configuration in $\mathcal{C}_{free}$ is greater than some fixed percentage $(1 - \epsilon)$ of the total volume of $\mathcal{C}_{free}$. The authors prove that if $\mathcal{C}_{free}$ is ϵ-good the probability that a roadmap computed by the basic-PRM algorithm does not cover this subset decreases exponentially with the number of nodes.

The notion of *expansiveness* introduced in [Hsu 97] deals with connectivity. The proposed model is rather technical. It deals with the notion of narrow passages in $\mathcal{C}_{free}$ and the difficulty to go through them. It is shown that for expansive $\mathcal{C}_{free}$ the probability for a (basic) roadmap not to capture the connectivity of $\mathcal{C}_{free}$ decreases exponentially with the number of nodes.

The *clearance* of a path is also a pertinent factor. In [Kavraki 98] a bound on the number of nodes required to capture the existence of a path of given clearance is provided. This bound depends also on the length of the path. In [Svestka 98] the dependence on the length is replaced by the dependence on the number of visibility sets required to cover the path. Again the probability to fail in capturing the existence of a path decreases exponentially with the number of nodes.

Finally, the dependence of the above notions (ϵ-goodness, expansiveness and clearance) on the dimension of $\mathcal{C}$ is discussed in [Hsu 99].

All these results are based on parameters characterizing the geometry of $\mathcal{C}_{free}$ which are a priori unknown. An approximated knowledge of such parameters could be very useful for the setting of algorithms, but it seems that a reliable estimation would take at least as much time as building the roadmap itself [Hsu 99]. An idea could be to try to estimate these parameters during the construction of the roadmaps. This is the underlying principle of the Visibility-PRM approach [Nissoux 99, Siméon 00], that allows an on-line estimation of the coverage.

A recent publication [Ladd 02] unifies some previous works and makes a reformulation looking for more intuitive parameters and opening a novel framework for the analysis of PRM-based algorithms applicable to more general problems such as kinodynamic planning.

In [Branicky 01], resolution completeness has been stated for PRM-based algorithms in which configuration sampling is made using quasi-random sequences. These are the so called QRM and LRM approaches. The argument for this assertion is that quasi-random samples are deterministic. The authors also suggest the existence of a relationship between dispersion and discrepancy of quasi-random samples and coverage and connectivity notions studied in the above cited works. Theoretical performance characteristics of these new sampling-based approaches have been recently published in [LaValle 03b].

The theoretical development and analysis of RRT algorithms has been, up to date, entirely fulfilled by LaValle's group. Main properties are presented in [LaValle 01c]. RRT planners are probabilistically complete, and the tree nodes converge to the sampling distribution (i.e. in theory, to a uniform coverage of $\mathcal{C}$ if a uniform random sampling is used). First algorithms developed by combining ideas of the RRT approach and dynamic programming algorithms have been proved to be resolution complete [Cheng 02].

2.2 Solution Methods for Loop Closure Equations

As explained in Section 1.2 loop closure constraints yield a system of non-linear polynomial equations (1.9) relating joint variables. Solving such a set of equations is a very hard problem with high computational complexity. The main difficulty with non-linearity is that, even when the system is not underdetermined, it has in general non-unique solutions. Moreover, the number of solutions may be infinite for mechanisms with special geometry (e.g. [Bennett 03, Crippen 92]).

The solution of loop closure equations is a usual requirement for the kinematic analysis and the synthesis of mechanisms. Linkage synthesis [Erdman 91], inverse kinematics for serial manipulators [Angeles 03] and forward kinematics for parallel mechanisms [Merlet 00], are the more investigated categories of problems leading to such sets of non-linear equations. Despite the interest and the intensive work in Robot Kinematics and Algebraic

Constraint Solving, no general solution, convenient for practical use in motion planning is available yet.

2.2.1 Solutions for Non-Redundant Mechanisms

Non-redundant closed-chain mechanisms have a finite number of solutions to their loop closure equations (excepting for mechanisms with special geometry). Several kinds of technique have been developed to predict bounds of the number of solutions and to find all these solutions for such systems of polynomial equations. Each method has strengths and weaknesses, and the applicability of most of them is highly dependent on the particular problem being addressed.

Two main approaches have been classically used: *continuation* and *elimination* (see [Nielsen 97] for a complete survey of these techniques). Polynomial continuation methods [Wampler 90] are purely numerical procedures able to find all possible solutions of the set of non-linear equations (in contrast to other numerical methods, such as Newton-Raphson, which converge to a single solution). These methods are based on homotopy techniques for gradually transforming a "start" system whose solutions are known to the system whose solutions are sought. These methods are robust and applicable to any set of equations. However, since they are iterative procedures, they are too slow in practice. Elimination approaches use one of the next algebraic methods: the Gröbner Basis method [Buchberger 82], which is an iterative variable elimination technique, or the resultant method [Gelfand 94], capable of eliminating all but one variable in a single step. In both cases, the elimination process normally leads to an univariate polynomial, relatively easy to solve [Pan 99]. The applicability of the Gröbner Basis method is mainly limited by its algorithmic complexity. Resultant methods can provide computationally fast techniques, but they require geometric intuition to find (if possible) the formula for the resultant.

Lately, *interval methods* for solving systems of non-linear equations have been proposed as an alternative to continuation and elimination methods. They are based on interval arithmetic [Moore 79, Hansen 92] and manipulate upper and lower bounds of variables. Two main classes of interval-based methods have been applied in Robotics: those based on the interval version of the Newton method [Rao 98, Castellet 98], and those based on subdivision [Sherbrooke 93, Merlet 01, Porta 02]. They are completely numerical and robust techniques. Although implemented techniques are still slow, recent improvements are significantly increasing their efficacy [Porta 03].

Note that all the above mentioned techniques address mechanisms with (more or less) general geometry. In most industrial applications, mechanisms are normally designed with particular geometries which allow a closed-form analytical solution to the loop closure equations. For instance, non-redundant serial manipulators often have the last three revolute axes intersecting in a same point, which greatly simplifies the solution of the inverse kinematics problem [Angeles 03].

2.2.2 Dealing with Redundancy

When the mechanism is redundant, the system of loop closure equations is underdetermined. Thus, there is an infinite number of solutions. Two problems have to be faced: the first is to obtain the set of all solutions, and the second is to represent such set.

Among the existing methods, only interval methods are able to provide a fully geometric representation of the solution. Interval methods return a set of boxes that discretize the continuum of solutions in the joint-space $\mathcal{Q}$ of the mechanism. Unfortunately, current implementations of interval methods are capable of handling only a few (two or three) degrees of redundancy [Porta 03].

Using elimination methods, the solution of a system with m variables could be in principle represented by combining a parameterization of ρ independent variables with equations giving the value of the $m-\rho$ dependent variables. Note that m is the dimension of $\mathcal{Q}$ and ρ is the degree of redundancy of the mechanism. However, such a representation is not a "proper" one since it is a priori unclear which combinations of parameter values will provide real solutions of the equations involving dependent variables. Each joint variable has a restricted set of values satisfying closure (this set is called the closure range in Section 1.3). Besides the closure range of a given parameter generally depends on the values already assigned to other parameters. Thus, there is a kind of "dependency" between the independent variables. As mentioned in [Celaya 93], only an "on-line" parameterization is possible: before assigning a value to a parameter, its closure range compatible with the already assigned ones has to be computed. An additional problem is that determining these sets is as difficult as solving the loop closure equations, and detailed analytical approaches have been proposed only for planar or spherical closed mechanisms without joint limits [Celaya 93].

In conclusion, for redundant closed-chain mechanism with ρ greater than two or three, the solution of the loop closure equations can be only geometrically represented by a set of points discretizing this solution. This could be made using an exhaustive search method for the independent variables, following a ρ-dimensional grid for example. However this solution is prohibitive when ρ is high. The other possibility is then to use randomized techniques to sample these parameters. The remaining difficulty is how to perform the sampling of the ρ independent variables in order to obtain real solutions for the dependent ones.

A Different Formulation

When the aim is only to obtain some of the possible solutions satisfying closure constraints, the problem can be formulated in a different way. For robotic manipulators, such a problem is often referred to as *redundancy resolution* [Siciliano 90]. The loop closure problem is usually formulated in the velocity domain and solved using the Jacobian pseudo-inverse [Klein 83], although other very different approaches have also been proposed [DeMers 97].

However, redundancy resolution involves, in addition to closure, other constraints such as moving obstacles or energy minimization. The solution techniques are more complex than necessary for the instance addressed here. Indeed, the redundancy resolution problem is closer to the motion planning problem than to the inverse kinematics problem, but the problem is treated locally.

Another formulation has been proposed in Robotics literature to solve the inverse kinematics problem. Given a metric in the Euclidean space, the problem consists in minimizing an error function which represents the distance between the current location and the sought location of the end-effector. Such a formulation is used for example in [Ahuactzin 99] to solve the point-to-point problem, in which not only a configuration of the robot satisfying the goal end-effector location must be found, but also a feasible (i.e. collision-free) path between an arbitrary initial configuration and this goal configuration.

Techniques to solve the point-to-point problem can be used to sample the variety, solution to the loop closure equations, by iteratively choosing different initial configurations. Nevertheless, since the feasibility of the path from the sampled configuration to the goal configuration is not important for this particular aim, simpler optimization approaches can be used. For instance, the use of a *randomized gradient descent* is proposed in [LaValle 99]. Other optimization-based approach that could be applied is the *cyclic coordinate descent* (CCD) [Welman 93]. This approach, with great success in the last years, is based on a minimization applied to each joint variable separately, which allows simple analytical procedures to accelerate convergence. The drawbacks of such optimization-based approaches are that they are exposed to the local minima problem and that the convergence can be slow. These drawbacks are more significant for highly-redundant mechanisms.

2.3 Motion Planning under Kinematic Closure Constraints

2.3.1 Complete Approaches

Exact algorithms for solving the general instance of the piano mover's problem (e.g. [Schwartz 83, Canny 88]), are, in theory, capable to handle closure constraints. These methods make an algebraic formulation of the motion planning problem in which polynomial equations defining closure constraints can be directly treated as other holonomic constraints such as those imposed by obstacles. However, as mentioned before, these approaches are impractical because of their computational complexity. Further developments have been carried out attempting to reduce the complexity of exact algorithms. In particular, algorithms presented in [Basu 00] are more interesting for motion planning problems under closure constraints since they are specifically designed to compute the connectivity of algebraic varieties, and their complexity mainly depends on the dimension of the variety rather than on the dimension of the ambient space (the joint-space in our case). But even for these improved techniques, problems involving complex mechanisms still remain intractable.

Lately, a complete path planning method has been proposed for closed kinematic chains with only spherical joints [Trinkle 02]. The approach is based on a complete understanding on the singularity sets in the configuration-space of the closed-chain mechanism. Unfortunately, techniques providing such a complete information are only available for the limited classes of mechanisms treated in the referred paper: spatial mechanisms with spherical joints and planar mechanisms with revolute joints. Besides, this method neglects collision avoidance. For computing collision-free motions, the best option proposed by the author is to use this technique as the local planner within a PRM-based algorithm. Such hybrid motion planner cannot be considered complete, but only probabilistically complete.

2.3.2 Sampling-Based Approaches

To the best of our knowledge, Koga and Latombe were the first to extend a randomized algorithm for planning collision-free motions of a closed-chain mechanism. This planner aimed to generate paths for several cooperating robot arms that manipulate a movable object among obstacles [Koga 94]. Basically, a feasible path is computed for the movable object using the RPP algorithm, which generates a concatenation of configurations. For each one of these configurations, the planner verifies if the the object can be grasped by the manipulators using a closed-form inverse kinematics solution. This method presents several drawbacks for more general applications. However, it provides some interesting ideas exploited in the approach we describe in Chapter 4.

Recently, two sampling-based approaches have been proposed which directly address the motion planning problem for closed kinematic chains [LaValle 99, Han 01]. Both approaches extended PRM-based (and one of them also RRT-based) algorithms to handle closure constraints, but they attack the problem in a quite different way. Optimization techniques are used in [LaValle 99] for computing nodes and local paths satisfying kinematic closure, while [Han 01] applies kinematic tool functions. As we will see below, some important details in these approaches make them inefficient for complex mechanisms. Nevertheless, they represent a significant advance of the state of the art in motion planning.

Optimization-Based Approach

The first PRM-based approach able to handle mechanisms with closed chains was presented in [LaValle 99]. More details of the method and the extension of RRT-based algorithms were subsequently published in [Yakey 00, Yakey 01]. The problem is formulated in the joint-space $\mathcal{Q}$. Closure constraints are expressed by error functions involving distances in the Euclidean space and numerical optimization techniques are used to sample and to connect configurations in the subset of $\mathcal{Q}$ satisfying these constraints within a given tolerance.

In the extended version of the PRM approach, the nodes of the roadmap are obtained by sampling random configurations that ignore constraints, and then performing a ran-

domized gradient descent to minimize closure error functions. The edges are computed by executing a randomized traversal of the constraint surface between two nodes, also using gradient descent. A technique to randomly sample the tangent space of the constraints is proposed in the referred papers which increases the efficiency of this process. This approach is completely general but suffers the drawbacks of optimization-based methods to solve inverse kinematic problems, mentioned in Section 2.2.

The application of the above method to RRT-based algorithms is discussed in [Yakey 00]. In this work, the random configurations q_{rand} used to bias the exploration are generated ignoring closure constraints. The argument is that computing feasible configurations is too expensive and does not provide appreciable benefit (we will show in Chapter 3 that this last assertion is not totally right). The tangent space sampling is applied to the expansion of the nearest neighbor q_{near} toward q_{rand}. The author shows via experimental results that this technique improves the exploration process in relation to a standard uniform random sampling.

Kinematics-Based Approach

The approach described in [Han 01] treats closed kinematic chains within a PRM-based planner. Each loop in the mechanism is broken into two subchains. For computing nodes, uniform random sampling is used to generate the configuration of one of the subchains (called *active subchain*) and then an inverse kinematics problem is solved to obtain the configuration of the remaining part of the loop (called *passive subchain*) in order to force closure. For computing edges, the local planner (the steering method) is limited to act on the active configuration parameters and the corresponding passive variables are computed for each intermediate configuration along the local path. Although this approach could be considered less general than the method in [LaValle 99], it performs more efficiently.

Another new technique proposed in this work is the application of a two-stage strategy for the construction of the roadmap when the closed-chain mechanism is "mobile" (e.g. two mobile manipulators grasping an object). In a first phase, a so called *kinematic roadmap* is constructed for a fixed location of a (virtual) base of the system without considering obstacles. Next, the environment is populated with copies of the kinematic roadmap using a random sampling strategy. The global roadmap is computed by connecting configurations in the different copies of the kinematic roadmap corresponding to the same closure type using a PRM planner for such a rigid body.

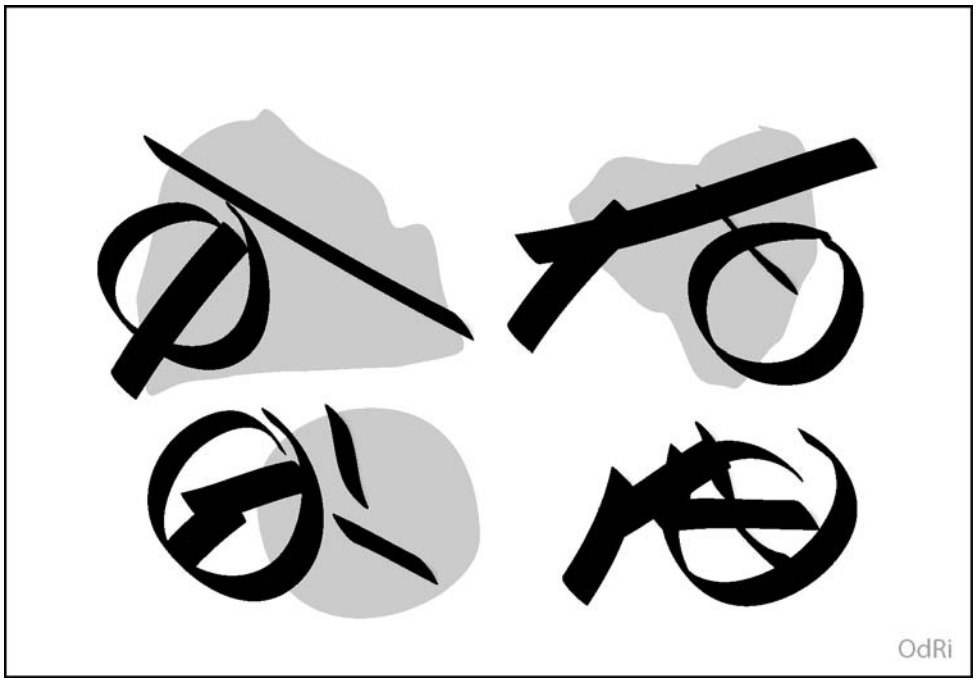
For the efficiency of the kinematic roadmap computation, the passive subchain of each loop must have closed-form inverse kinematics solution. Thus, passive subchains are in general articulated mechanisms with degree of mobility $M = 6$. As the authors admit, this means an important drawback when the approach is applied to a highly-redundant loop since the active part is a long kinematic chain. The probability of randomly generating configurations of a long active subchain for which a configuration of the passive chain satisfying closure constraints exists (i.e. the inverse kinematics problem can be solved) is

very low. This fact significantly drops off the performance of the algorithm.

One of the contributions in this thesis is to resolve the main drawback of the approach in [Han 01]. We have developed a general and simple geometric algorithm called Random Loop Generator (RLG) for sampling random configurations satisfying loop closure constraints [Cortés 02b, Cortés 03a]. RLG performs a "guided"-random sampling of active subchains that notably increases the probability to obtain solutions for the corresponding passive subchains. This algorithm is explained in the next chapter, within our approach for planning the motions of general closed-chain mechanisms.

Chapter 3

Sampling-Based Motion Planning for Closed-Chain Mechanisms



As explained in Chapter 1, solving motion planning problems for closed-chain mechanisms requires to explore the connectivity of subsets on an algebraic variety, and as we have seen through Chapter 2, only a few techniques allow to tackle complex instances of this problem. Our approach is placed within the frame of sampling-based motion planning algorithms. The aim is to improve these techniques with new tools that allow their application to more complex classes of problems involving kinematic closure constraints.

One of the main contribution of our work is to introduce sampling-based algorithms into a detailed formulation of the motion planning problem in presence of kinematic closure constraints, different from the formulation in [LaValle 99, Yakey 01], which had not been made in this context (see Section 3.1). Some of the ideas in our approach have been proposed in [Han 01] for the extension of PRM-based algorithms to treat closed kinematic chains. The most important technical advance concerns the configuration sampling algorithm explained in Section 3.2. This new technique resolves the main drawback of the method in [Han 01] and can be implemented within different kinds of motion planning algorithms. Section 3.3 deals with the extension of the PRM approach, and RRT-based planners are treated in Section 3.4. In these two sections, we present performance tests that show the interest of integrating our configuration sampling algorithm within these planners. Then, in Section 3.5, we comment experimental results obtained when solving motion planning problems. Note that the efficiency and generality of the extended planners will be further demonstrated in the second part of this thesis through the different examples of application.

3.1 Overview of the Approach

Our approach for planning the motions of closed-chain mechanisms combines existing sampling-based algorithms with simple geometric tools and efficient solution methods for loop closure equations. The algorithms explained in the next sections explore the subset of the configurations satisfying motion constraints $\mathcal{C}_{feas}$ and construct data structures (graphs or trees) that encode the connectivity of this subset.

Let us go back to the fictive motion planning problem introduced in Section 1.4. Figure 3.1 shows a probabilistic roadmap capturing the connectivity of the search-space. Black and white points (the nodes) are feasible configurations (we will explain later the meaning of the different colors). Straight-line segments (the edges) represent feasible connections between pairs of nodes. Explicit information about how to connect the extreme points while maintaining motion constraints is associated with each edge. Then, a planning query will be immediately answered by connecting the initial and goal configurations to nearby nodes and searching for a path in the roadmap.

The use of sampling-based motion planning algorithms is strongly justified since, for the kind of problems we address (see the second part of this thesis), there is no available technique providing a complete, exact representation of $\mathcal{C}_{feas}$. However, there is an

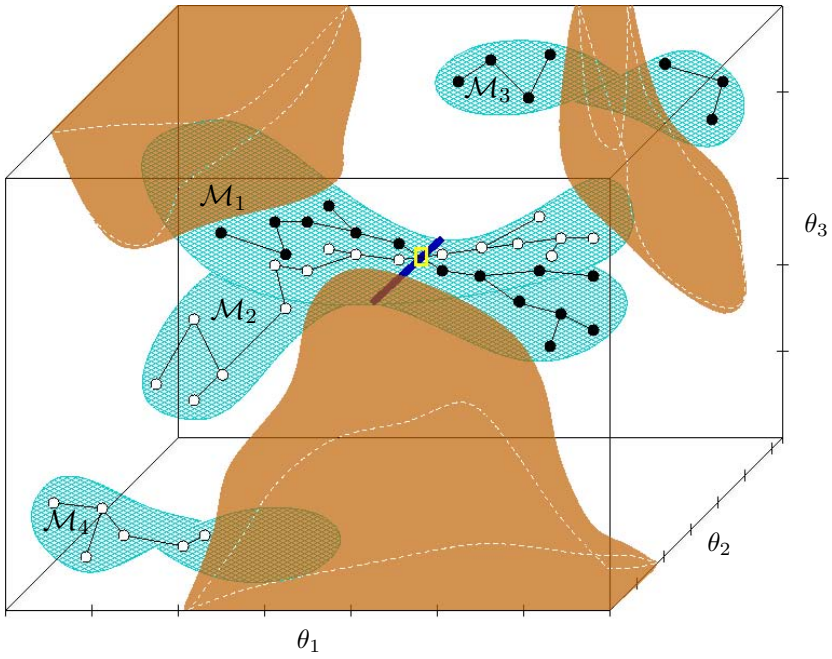


Figure 3.1: Probabilistic roadmap capturing the connectivity of $\mathcal{C}_{feas}$ for the fictive motion planning problem involving kinematic closure constraints of Section 1.4.

important difficulty involving the sampling of the configuration-space $\mathcal{C}$ of closed-chain mechanisms. Configurations satisfying closure constraints form an algebraic variety embedded in the joint-space $\mathcal{Q}$ (i.e. the space of the variables in our problem). Thus, the probability of obtaining a feasible configuration $q \in \mathcal{C}_{feas}$ by sampling the joint variables is null. Consequently, this family of algorithms cannot be directly applied in theory. However, the topological structure of $\mathcal{C}$ allows a parameterization via local charts that enables the use of sampling-based planners by combining them with other numerical and/or algebraic techniques.

On the Parameterization of $\mathcal{C}$

Configurations of a closed-chain mechanism are grouped into a finite number of M-dimensional smooth manifolds $\mathcal{M}_i$, with M the global mobility of the mechanism. These manifolds can be parameterized, at least locally, by a set of M (independent) parameters. Points in the different manifolds can be generated by sweeping the M parameters through their range and evaluating expressions that provide the value of the other (dependent) variables. Such expressions are the loop closure equations (1.9). An *atlas* of each one of these manifolds $\mathcal{M}_i$ can be constituted by a finite number (in general not exceeding the

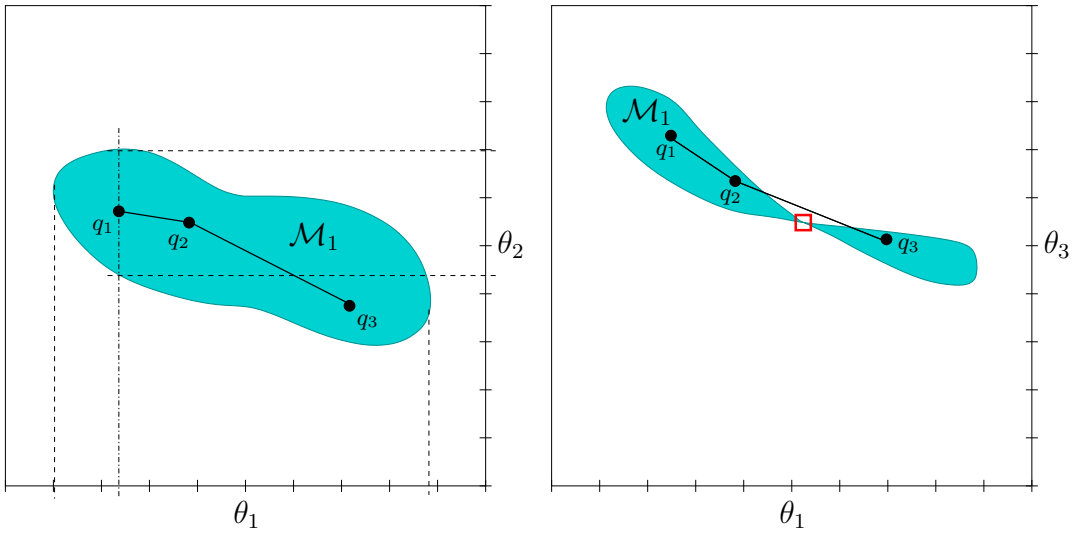


Figure 3.2: Projection of $\mathcal{M}_1$ on the planes $\theta_1\theta_2$ and $\theta_1\theta_3$.

dimension m of $\mathcal{Q}$) of *local charts* considering different combinations of M parameters [1]. Without loss of generality, sets of M consecutive joint variables can be chosen as local coordinates [Thomas 93]. However, any combination of variables is not permitted, specially in the case of multi-loop mechanisms.

Following the terminology in [Han 01], we call *active variables* q^a the set of the M joint variables chosen as parameters of a local chart and *passive variables* q^p the remaining set of the $m - M$ dependent joint variables, so that $\{q^a, q^p\} = q \in \mathcal{Q}$.

Main Principle

The core of our approach is to explore the connectivity of $\mathcal{C}_{feas}$ by sampling configurations and by testing feasible connections through local parameterizations of $\mathcal{C}$. Motion planning algorithms are applied on the local parameters q^a. Using a roadmap method such as PRM, the nodes are generated by sampling q^a and local paths are obtained by applying steering methods to these parameters. In a similar way, q^a are the configuration parameters handled by incremental search methods as RRT. Obviously, for each computed value of the parameters, loop closure equations must be solved for obtaining the whole configuration of the mechanism $q \in \mathcal{C}$. Therefore, the efficiency of the planner partially relies on the efficient solution of these equations. In most cases, an adequate selection of parameters permits the application of efficient methods for solving inverse kinematics problems.

Without a topological characterization of $\mathcal{C}$, the number of different sets of parameters q^a required to solve a particular motion planning is a priori unknown. This fact can

[1] Charts and atlas are usual notions in Differential Geometry which can be found in any textbook (e.g. [Talpaert 93, Henderson 97]).

easily be understood by the illustrative example in Figure 3.1. The supposed mechanism with three joint variables $\{\theta_1, \theta_2, \theta_3\}$ has a global mobility $M = 2$. Thus, three different sets of parameters q^a can be chosen: $\{\theta_1, \theta_2\}$, $\{\theta_1, \theta_3\}$ and $\{\theta_2, \theta_3\}$. Figure 3.2 shows the projection of the manifold $\mathcal{M}_1$ on the planes $\theta_1\theta_2$ and $\theta_1\theta_3$. For each point of a parameterization with $q^a = \{\theta_1, \theta_2\}$ there is an only configuration q lying on $\mathcal{M}_1$ (i.e. this mapping is injective). One can easily understand that such a parameterization is sufficient for capturing the connectivity of $\mathcal{M}_1$ using sampling-based approaches. A PRM-based algorithm would construct a roadmap with only one connected component covering the whole manifold. Any two configurations could also be connected using the RRT approach. On the contrary, choosing θ_1 and θ_3 as only parameters could lead to a non-complete solution. Let us consider that the steering method makes a linear interpolation of the parameters q^a. A solution path between configurations q_1 and q_2 could be immediately obtained since they can be directly connected by a local path. However, a path between q_1 and q_3 could not be found using sampling-based techniques. The point indicated by the small square is a singularity of this parameterization. The probability of generating this point by sampling values of $q^a = \{\theta_1, \theta_3\}$ is null, as well as the probability of sampling two points on a line (local path) passing through this singularity. Finding a feasible path between q_1 and q_3 requires to use another set of local parameters.

In practice, the last assertion is not totally true when the test for validating local paths is discrete. Closure constraints can be violated as other constraints (e.g. collision avoidance) depending on the resolution at which local paths are discretized. Thus, one only set of local parameters is sufficient for solving motion planning problems under a certain tolerance [2] associated with this resolution. However, some parameterizations are more appropriate than others, and the planner could be more efficient by combining different sets of parameters q^a. We will come back to this discussion later, when talking about kinematic singularities.

How to Sample q^a

Figure 3.2 also illustrates another difficulty related to the definition of the parameters q^a. A parameterization of $\mathcal{M}_1$ using θ_1 and θ_2 as coordinates allows to solve motion planning problems on this manifold. However, it is not a proper parameterization. Loop closure equations have real solutions only for a range of values in the interval of each joint variable, defined as closure range in Chapter 1. Besides, the closure range of a parameter depends on the value of the other parameters. If we sample first θ_1 and then θ_2 for generating a configuration $q_1 \in \mathcal{M}_1$, then θ_1 can be sampled in its whole closure range (i.e. the feasible range for any value of the other joint variables). However, θ_2 is valid only in a subset of its whole closure range determined by the value of θ_1.

There is no general and efficient method to define closure ranges of joint variables. Thus, in practice, the only possibility for sampling configurations is to use a trial method:

[2] Note that the planners in [LaValle 99, Yakey 00, Yakey 01] also work within a given tolerance.

sampling parameters q^a in the intervals defined for joint variables and solving the loop closure equations. Nevertheless, when a closure range is very restricted with respect to the interval of a joint variable, too many samples maybe tested before finding a feasible configuration. Hence, too much computing time is spent in solving closure equations leading to imaginary values. This is an important drawback for the efficiency of motion planners, and mainly for those using a roadmap approach, such as the planner in [Han 01]. We have developed an algorithm called *Random Loop Generator* (RLG) that resolves this problem using simple geometrical operations. The RLG algorithm performs a particular random sampling for q^a that notably increases the probability of obtaining real solutions for q^p. The only limitation for this algorithm is to require that the joint variables in q^a and q^p correspond to consecutive joints in the mechanism; thus it can not be applied for any choice of parameters. The selection of active and passive variables for the application of RLG will be further discussed in Section 3.2, which also describes the RLG algorithm.

Dealing with Kinematic Singularities

Up to now we have limited our discussion to the case of a single manifold. However, $\mathcal{C}$ may be composed of several manifolds. These manifolds are either disjoint, or they intersect at lower-dimensional subsets corresponding to kinematically singular configurations. Therefore, exploring the connectivity of $\mathcal{C}_{feas}$ requires to deal with these singularities.

In our illustrative example (see Figure 3.1), configurations are grouped into four manifolds. Figure 3.3 shows their projections on the planes $\theta_1\theta_2$ and $\theta_1\theta_3$. For each point of a parameterization with $q^a = \{\theta_1, \theta_2\}$ we can have zero, one or two real values of $q^p = \theta_3$ satisfying loop closure equations within the interval defined by the joint limits. In this case,

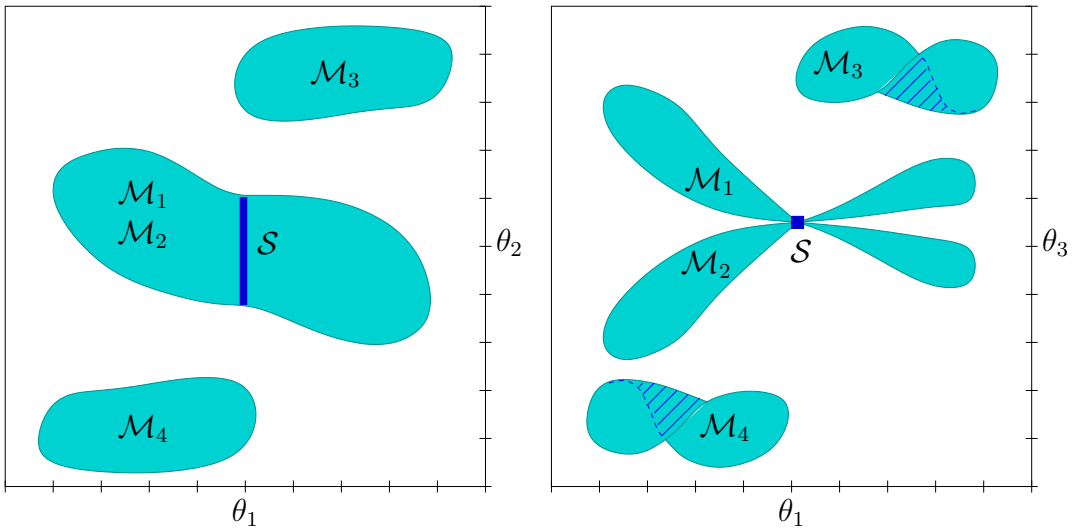


Figure 3.3: Projection of $\mathcal{C}$ on the planes $\theta_1\theta_2$ and $\theta_1\theta_3$.

different solutions of q^p for a given value of the parameters q^a yield configurations lying on different manifolds. However, this is not a general rule, see for example the projection of $\mathcal{M}_3$ and $\mathcal{M}_4$ on the plane $\theta_1\theta_3$. Into the striped regions, two points of these manifolds have the same projection. But the most important detail illustrated in this figure is that $\mathcal{M}_1$ and $\mathcal{M}_2$ have the same projection on the plane $\theta_1\theta_2$. Each point $\{\theta_1, \theta_2\}$ maps to two configuration on different manifolds for all the domain under $\mathcal{M}_1$ and $\mathcal{M}_2$ except for the singular set $\mathcal{S}$ where they intersect, so that configurations in these manifolds are connected by this singular set. Note that this kinematic singularity corresponds to the singularity of the parameterization $q^a = \{\theta_1, \theta_3\}$ that we have commented above. As we have mentioned, the probability of sampling a configuration $q \in \mathcal{S}$ is null, but in this case, this singularity can be traversed by a local path on a parameterization with $q^a = \{\theta_1, \theta_2\}$ or $q^a = \{\theta_2, \theta_3\}$.

The roadmap in Figure 3.1 has been built using $q^a = \{\theta_1, \theta_2\}$ as only set of parameters. White nodes correspond to one of the solutions of θ_3 and black nodes to the other. When a local path is computed between two configurations at different sides of $\mathcal{S}$, the bifurcation of the solution is detected, and the singular configuration (marked by the small rectangle in Figure 3.1) connecting $\mathcal{M}_1$ and $\mathcal{M}_2$ is then identified.

Let us consider now a worst case. Imagine that, in our three-dimensional example (see Figure 3.1), $\mathcal{M}_1$ and $\mathcal{M}_2$ do not intersect along a line but they meet at a point. None of the three possible parameterizations would allow to identify such a singular point exactly. We could only aim to find a local path passing through this point within a given tolerance. But even that is very improbable if paths are discretized with high resolution, aiming to provide good-quality solutions.

The difference between the two above mentioned cases of connection between manifolds is that the singular set has dimension $M - 1$ for the former and $M - 2$ for the latter. In theory, sets of kinematically singular configurations can have dimension from $M - 1$ to zero. Using sampling techniques for generating configurations on $\mathcal{C}$ and steering methods on subsets of configuration parameters q^a, our approach has only the guarantee (forgetting tolerance) to find connections through singular sets of dimension $M-1$. The other singular sets, from dimension $M - 2$ to isolated singularities, must be identified by other methods. Unfortunately, identifying singular sets on an algebraic variety is in general extremely difficult. However, for most of practical applications, techniques developed in the field of Robot Kinematics could be applied (e.g. [Gosselin 90b, Merlet 92]). Many mechanisms that can be treated by our approach are too complex for a direct application of these techniques onto the whole system. But the global analysis is not strictly necessary, the goal is to identify singularities associated with the different local parameterizations. Indeed, for a single loop and for a given choice of parameters, singularities can be found by analyzing the passive variables q^p. Singularities appear at points q^a where there is a change in the number of solutions of q^p. An appropriate selection of q^p should permit to know these points. Identified singular configurations could then be injected as special nodes in a roadmap or in a search tree used to connect components on different manifolds

Nevertheless, as we mentioned at the beginning of this document, the algorithms we present in this thesis do not deal with singular configurations. This could be an interesting extension for future work. However, the interest of a treatment of kinematic singularities, at least for robotic applications, should be to remove singular configurations from paths rather than to search connections between manifolds.

Planning under Holonomic Inequality Constraints

Because of our definition of configuration-space, every configuration $q \in \mathcal{C}$ satisfies holonomic equality constraints. It remains now to discern the feasible subset $\mathcal{C}_{feas}$ determined by holonomic inequality constraints. The satisfaction of these constraints is checked for nodes and local paths computed to construct a roadmap or during the expansion of a search tree. Since feasibility constraints affect the whole configuration q and not only the parameters q^a, loop closure equations must be solved before the checking. Thus, it seems logical to check closure constraints and other holonomic constraints with the same resolution along paths. Although, when it is possible, different discretization steps for different constraints can yield more efficient algorithms.

In standard motion planning problems, holonomic inequality constraints are limited to collision avoidance and joint limits (i.e. $\mathcal{C}_{feas} = \mathcal{C}_{free}$). The examples of application we show in the second part of this thesis only consider this instance. In Section 2.1, we mentioned the importance of collision detection in sampling-based planners, and we explained that efficient algorithms test collisions along path using an adaptive resolution instead of a constant discretization step. However, it is not clear how to apply these dynamic collision detection techniques to problems involving closure constraints. The difficulty is that the steering method is applied to the subset q^a of the joint variables and we have not a priori information about the variation of the passive variables q^p along a local path. In Section 3.3, we propose a method for collision detection along local paths.

Planning under Non-holonomic Constraints

Even if non-holonomic constraints go beyond the scope of this thesis, we want to give some guidelines on how they could be considered within our approach.

For a given parameterization, the steering method is applied to the active variables q^a. In principle, steering methods producing any variation of these parameters can be used. Passive variables q^p are limited to follow the variation of q^a through the solution of loop closure equations. Since the design of steering methods for the q^a producing a variation of q^p that satisfies a certain type of differential constraint seems highly complex, it is preferable to choose joint variables affected by non-holonomic constraints within the parameters q^a. For some types of non-holonomic constraints, available methods (e.g. [Sekhavat 98, Laumond 98b, Lamiraux 01b]) can be applied to q^a for computing kinematically feasible local paths.

3.2 Configuration Sampling Under Closure Constraints

We describe a technique, that we call Random Loop Generator (RLG), for sampling configurations of general closed-chain mechanisms. It is based on a decomposition of the mechanism into open kinematic chains. The configuration of the open chains involving active joint variables q^a is sampled using a simple geometric algorithm that notably increases the probability of obtaining real solutions for the passive variables q^p when solving the loop closure equations. Later, in Chapter 4, we will explain an adapted version of RLG for parallel mechanisms, that are a particular class of multi-loops.

3.2.1 Mechanical System Decomposition

RLG handles the single loops in the articulated mechanism separately. For each single loop, sets of active and passive joint variables are defined consecutively such that they correspond to segments of the kinematic chain. We call *passive subchain* the segment involving the passive variables and *active subchains* to the other segments. There can be one or two active subchains depending on the placement of the passive subchain. Look at Figure 3.4 for a simple illustration. The $6R$ planar linkage has mobility $M = 3$. Thus, q^a and q^p contain three joint variables each. In this illustration we have chosen θ_3, θ_4 and θ_5 (the joint variables associated with joints J_3, J_4 and J_5) to be the passive variables. Then, active variables can be seen as configuration parameters of two open chains rooted at a (fictive) link $\mathcal{A}_{0,6}$.

The passive subchain is a non-redundant mechanism whose end-frame can span full-rank subsets of the workspace. In general, this requires three joint variables for a planar mechanism and six for a spatial mechanism. Efficient methods to solve inverse kinematics problems for such mechanisms are available [Angeles 03]. Furthermore, in many practical cases, analytical solutions can be applied if the passive subchain is appropriately chosen. Imagine for instance an example where an object is handled by two non-redundant robotic arms with decoupled position/orientation for the end-effector (see examples in Section 3.5). A sensible choice for sampling $\mathcal{C}$ is to make q^p alternately correspond to the joint variables

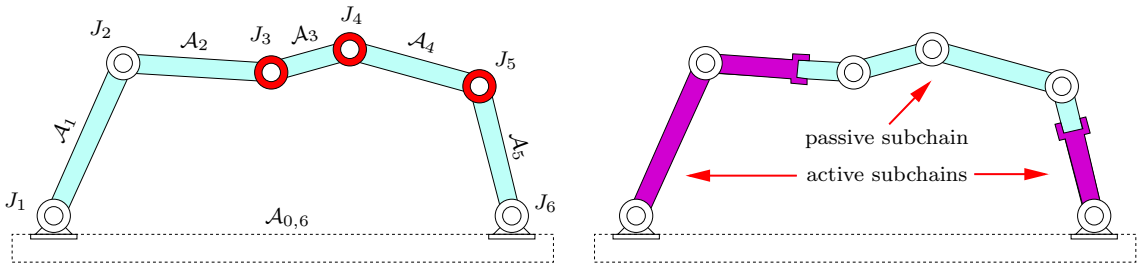


Figure 3.4: Planar $6R$ linkage: decomposition into active and passive subchains.

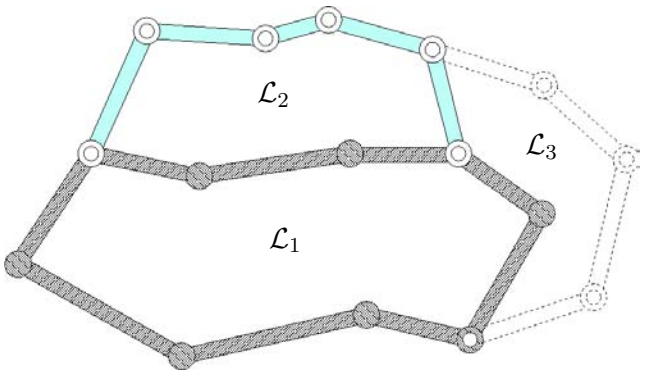


Figure 3.5: Decomposition of a multi-loop. The indexes of the individual loops indicate the order for generating the configuration.

of one of the arms. Such a choice allows analytical solution for these passive variables and permits to identify its class (e.g. up elbow or down elbow). Besides, singular configurations of such parameterizations are know.

When several loops compose the mechanism, an order has to be defined for generating the configuration of each one of them. In multi-loops, some joint variables are involved in the configuration of several individual loops. Their value is computed for the single loop treated first, and then, these common portions of the mechanism become rigid bodies when treating the other loops. Figure 3.5 shows an example of a planar multi-loop mechanism. The individual loops are designated by $\mathcal{L}_i$. If $\mathcal{L}_1$ is treated first, then $\mathcal{L}_2$ corresponds to the loop in Figure 3.4. Once generated the configuration of these two loops, RLG considers them rigid for the treatment of $\mathcal{L}_3$.

Defining a good order for treating portions of a multi-loop is not an easy task. A good heuristic is to treat first the loop sharing the highest number of joint variables with other loops. However, there are many other restrictions to consider, concerning the mobility of the individual loops or the existence of differential constraints for example. The decomposition of a complex multi-loop requires a careful analysis of the kinematic diagram of the mechanism considering differential motion constraints.

3.2.2 RLG Algorithm

The algorithm generating the configuration of a single-loop closed chain $\mathcal{L}$ for a given selection of parameters is synthesized in Algorithm 3.1. First, the configuration parameters of the active subchains, q^a, are computed by the function SAMPLE_q^a detailed in Algorithm 3.2. These joint variables are computed sequentially. The idea of the algorithm is to progressively decrease the complexity of the closed chain treated at each iteration until only the configuration of the passive subchain, q^p, remains to be solved. The two active subchains are treated alternately. The ideal solution should be to sample each joint

Algorithm 3.1: RLG_SingleLoop

input : the loop $\mathcal{L}$
output : the configurations $q[n_{sol}]$
begin
  $q^a \leftarrow$ Sample_$q^a(\mathcal{L})$;
  $q^p[n_{sol}] \leftarrow$ Compute_$q^p(\mathcal{L}, q^a)$;
  if $n_{sol} = 0$ **then return** Failure;
  else $q[n_{sol}] \leftarrow$ CompoundConf$(\mathcal{L}, q^a, q^p[n_{sol}])$;
end

Algorithm 3.2: Sample_q^a

input : the loop $\mathcal{L}$
output : the parameters q^a
begin
1 $(J_b, J_e) \leftarrow$ InitSampler$(\mathcal{L})$;
  while not EndActiveChain$(\mathcal{L}, J_b)$ **do**
    $I_c \leftarrow$ ComputeClosureRange$(\mathcal{L}, J_b, J_e)$;
    if $I_c = \emptyset$ **then goto** line 1;
    SetJointValue$(J_b,$ Random$(I_c))$;
    $J_b \leftarrow$ NextJoint$(\mathcal{L}, J_b)$;
    if not EndActiveChain$(\mathcal{L}, J_e)$ **then** Switch(J_b, J_e);
end

variable from the subset of values satisfying the loop closure equations (i.e. the closure range). However, computing this subset is as difficult as solving the general inverse kinematics problem. Thus, an approximation is used. The approximation must be conservative in the sense that no region of $\mathcal{C}$ is excluded for the sampling. This is required in order to guarantee any form of sampling-based completeness of motion planning algorithms. The function ComputeClosureRange returns a set of intervals I_c which approximate the exact closure intervals. We explain how to obtain them in Section 3.2.3. The closure range of the joint variable treated at one iteration depends on the configuration of the previously treated joints. Hence, I_c must be recomputed for all the joints (except the first treated one) in the generation of each new configuration. Because of the conservative nature of the approach, it is possible to obtain an empty set in an iteration. In this case, the process is restarted.

For a sampled value of parameters q^a, the value of q^p is computed by solving an inverse kinematics problem for the passive subchain. In general, there are several, n_{sol}, possible solutions. Thus, n_{sol} configurations are obtained by combining q^a with the different solutions for q^p.

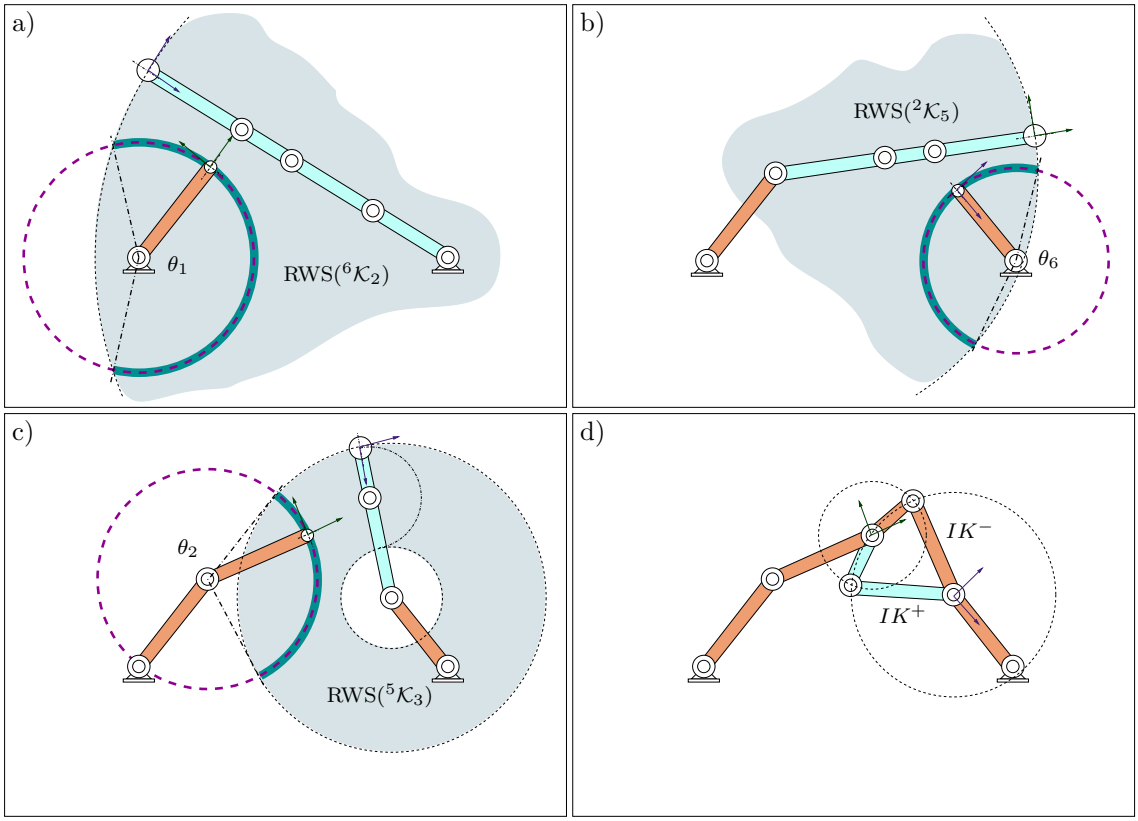


Figure 3.6: Steps of the RLG algorithm performing on a $6R$ planar linkage.

Figures 3.6.a,b,c illustrate how the values of θ_1, θ_6 and θ_2 (the parameters q^a) are sequentially generated for the example of the planar $6R$ linkage. For each joint variable, the estimation of the closure intervals (there is only one interval in this example) is computed and a random value is sampled inside this set. Figure 3.6.d shows the two inverse kinematics solutions for the $3R$ planar mechanism corresponding to the passive subchain. In this case, these solutions are obtained by simple trigonometric operations.

3.2.3 Approximated Closure Range

The problem of computing the closure range of a joint variable can be formulated as follows. Given a closed kinematic chain ${}^b\mathcal{K}_e$ involving joints from J_b to J_e (we consider $b < e$ in this explanation), two open kinematic chains are obtained by breaking the link $\mathcal{A}_b$ between J_b and J_{b+1}. A suitable break-point is the physical placement of J_{b+1}, but any other point can be chosen. A frame F_c associated with this break-point can be seen as the end-frame of both open chains. The closure range of the joint variable (or variables, for joints allowing several d.o.f.) corresponding to J_b is the subset of values making F_c reachable

by the chain ${}^{e}\mathcal{K}_{b+1}$. Solving such a problem requires to represent the workspace [3] of this chain, which is in general very complicated. Indeed, works in Robot Kinematics literature that determine the workspace of manipulators are limited to particular instances (e.g. [Ricard 98, Merlet 95]). Thus, we must use an approximated approach. For our purpose, a simple and fast method is preferred versus a more accurate but slower one.

We solve the problem considering only positional reachability. The *reachable workspace* [4] of the chain ${}^{e}\mathcal{K}_{b+1}$ is approximated by a simple bounding volume, that we denote by RWS(${}^{e}\mathcal{K}_{b+1}$). For chains containing any kind of lower-pair joint except planar joints (see Section 1.1.2), *spherical shells* are a reasonable option for the RWS. A spherical shell is defined by the intersection of the volume between two concentric spheres and a cone whose apex coincides with their common center. Parameters characterizing the spherical shell are derived from the geometry of the chain. The center is the origin of the base-frame, which normally corresponds to the origin of $F_{\mathcal{A}_e}$, O_e. The external and internal radii, r_{ext} and r_{int}, correspond respectively to the maximum and minimum extensions of the chain. This extension is measured from O_e to the origin O_c of frame F_c. The axis of the cone cutting the full shell is a vector $\hat{U}$ associated with the base-frame. The half-opening angle γ is the maximum angle between $\hat{U}$ and the vector $\overrightarrow{O_eO_c}$. We will further discuss how to compute the spherical shell dimensions later in this section.

[3] In the current context, we call workspace to the subset of $SE(3)$ that can be mapped by the end-frame of a kinematic chain.

[4] Subset of $\mathbb{R}^3$ that can be mapped by the origin of the end-frame.

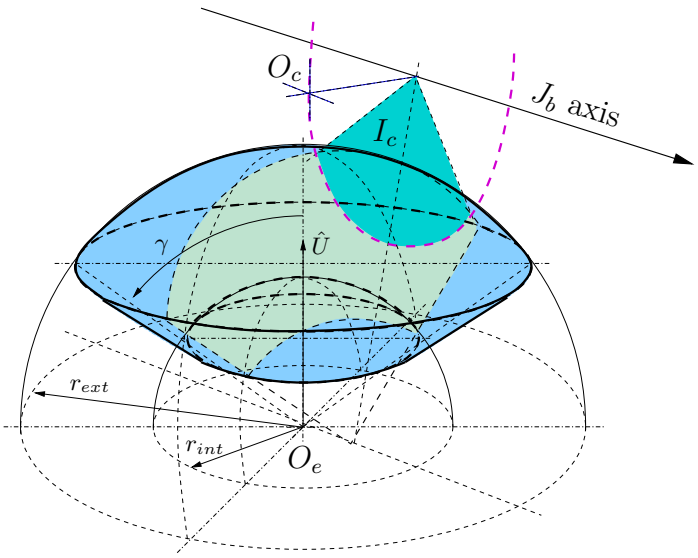


Figure 3.7: The intersection of a spherical shell and a circle determines the (approximated) closure range of a rotational d.o.f..

When the chain ${}^{e}\mathcal{K}_{b+1}$ involves a planar joint, usually, the positional variation produced by this joint is very important with relation to the length of the rest of the mechanism. Imagine for instance a robotic arm placed on a mobile platform (i.e. a mobile manipulator). If J_e is the planar joint, then $\mathrm{RWS}({}^{e}\mathcal{K}_{b+1})$ is the volume covered by moving $\mathrm{RWS}({}^{e-1}\mathcal{K}_{b+1})$ on the plane xy defined by the joint limits of J_e. A box containing such a volume is a simple and sufficiently good approximation.

Once defined $\mathrm{RWS}({}^{e}\mathcal{K}_{b+1})$, computing the approximation of the closure intervals for J_b is very simple. If J_b is a revolute joint, then O_c describes a circle (or a circular arc, considering joint limits) around its axis. If J_b is a prismatic joint, O_c moves on a straight-line segment. Then, I_c is obtained from the intersection of a circle or a line with a simple volume RWS. Figure 3.7 illustrates the case of a revolute joint and a spherical shell. When J_b is a joint allowing several d.o.f., then each one of the elementary joints $J_{b.j}$ is treated sequentially.

Computing the Spherical Shell Dimensions

Particular methods can be adopted for computing the dimensions of the spherical shell for particular classes of mechanisms. The solution is straightforward for a planar linkage made up with revolute joints without mechanical stops (like the $6R$ in the previous illustrations). The external and internal radii of the annulus are given by:

$$r_{ext} = \sum_{i=1}^{n_{link}} a_i \quad ; \quad r_{int} = \begin{cases} 2a_{max} - r_{ext} & \text{if } 2a_{max} > r_{ext} \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

where the a_i are the link lengths and a_{max} is the length of the longest one. Since the joints allow full 2π rotation, the half-opening angle γ equals to π.

For general kinematic chains, obtaining the spherical shell dimensions requires to solve complex optimization problems. Unfortunately, most of the global optimization algorithms (see [Bliek 01] for an extensive survey) return suboptimal solutions (i.e. lower bounds when maximizing and upper bounds when minimizing). Remember that we need a conservative approximation to guarantee sampling-based (e.g. probabilistic) completeness. Besides, these algorithms are limited in the number of variables they can handle. Thus, giving a general recipe for arbitrary mechanisms is not easy. We next propose a simple method to obtain r_{ext} and r_{int} that provides in general good results.

The distance between the origin of consecutive frames $F_{\mathcal{A}_{i-1}}$ and $F_{\mathcal{A}_i}$ is obtained from the mDH parameters as: $l_i = \sqrt{a_{i-1}^2 + d_i^2}$. If J_i is a prismatic joint, then d_i is variable. We denote by l_i^+ and l_i^- the distances obtained for the maximum and minimum values of translational d.o.f.. The next expressions are a simple method for obtaining an upper bound for the maximum extension, $\hat{r}_{ext}$, and a lower bound for the minimum extension,

$\hat{r}_{int}$, of any open kinematic chain:

$$\hat{r}_{ext} = \sum_{i=1}^{n_{link}} l_i^+ \quad ; \quad r_{int} = \begin{cases} 2l_{max} - \sum_{i=1}^{n_{link}} l_i^- & \text{if } 2l_{max} > \sum_{i=1}^{n_{link}} l_i^- \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

Note that these equations provide extensions of an equivalent mechanism where all the joints producing rotation are replaced by spherical joints without joint limits. The accuracy of these bounds could be improved using simple algorithms used in Distance Geometry (e.g. [Dress 88, Rajan 99]). However, in many cases, these values are bad approximations, mainly because of the presence of joint limits. We use an artifice that increases the efficiency of RLG while keeping completeness. More accurate non-conservative values $\tilde{r}_{ext}$ and $\tilde{r}_{int}$ can be computed by a simple optimization method such a randomized gradient descent. Then, instead of using a constant value for the dimensions of the spherical shell, we sample them from a distribution between the approximation $\tilde{r}_{int/ext}$ and the conservative bound $\hat{r}_{int/ext}$ each time they are required within the function ComputeClosureRange in Sample_q^a (see Algorithm 3.2). The type of distribution to be used depends on each particular chain. We have not already made an analysis for definig criteria for this choice. Nevertheless, we noted that a Gaussian distribution with $\mu = \tilde{r}_{int/ext}$ and $\sigma^2 = 1$ provides in general good results.

In practice, considering the full shell (i.e. $\gamma = \pi$) is a reasonable approximation for the subchains handled by RLG. Note that we need to compute the volume RWS for chains involving at least 6 d.o.f. for general spatial mechanisms and 3 d.o.f. for planar mechanisms. Therefore, only when the joint limits are very restrictive, γ should be computed for getting a better performance of RLG. However, we do not know a general strategy for defining $\hat{U}$ and computing an upper bound of γ. We only introduce the cone cutting the full shell in some cases when the kinematic features of the chain are well known, such as in examples of Chapter 4.

3.3 Extension of PRM-based Algorithms

The main ideas on the application of our approach to sampling-based motion planning algorithms have been outlined in Section 3.1. In this section we detail some points concerning more particularly PRM-based planners.

The extension of the basic algorithm for the roadmap construction explained in Chapter 2 does not require to modify lines of the pseudocode (Algorithm 2.1). The difference is that the functions generating nodes (RandomFreeConfiguration) and computing feasible local paths (FeasibleConnection) must consider the presence of kinematic closure constraints. These functions operate separately with the active and the passive variables. Indeed, these functions directly handle only the active variables q^a. The rest of the variables, q^p, are implicitly determined via the loop closure equations.

3.3.1 Generating Nodes

Closed-chain mechanisms can be in general decomposed as explained in Section 3.2, and then RLG can be applied for generating random configurations within the function RANDOMFREECONFIGURATION. Normally, several decompositions of this type can be achieved for the same mechanism so that RLG can work with different parameterizations. The use of several parameterizations for sampling $\mathcal{C}$ could provide a more uniform coverage of the variety. However, it is not important from the point of view of singularities, since lower-dimensional subsets of $\mathcal{C}$ will (in theory) be never sampled.

Note that, using RLG, we obtain in general n_{sol} configurations sharing parameters q^a and with different configurations for the passive subchains. Then we have two options, either generating only one new node or trying all these configurations. The random samples are checked for collisions and, eventually, other holonomic constraints. If we choose the former option, the first valid sample is the new node q_{new}. Otherwise, all the valid configurations are inserted in the roadmap as nodes q_{new_i}, with $0 \leq i \leq n_{sol}$.

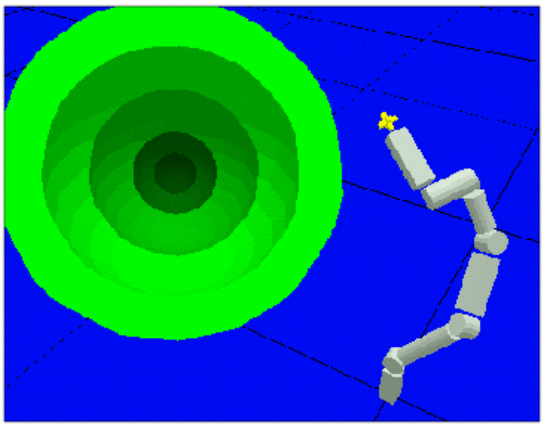
The benefit of using RLG for the global performance of the planner will be demonstrated through comparative results when solving motion planning problems that we describe in Section 3.5. Here, we present results of simple tests that allow to analyze the performance of RLG as a separated module.

Figure 3.8 contains illustrations and results of two tests. They consist in generating random configurations of spatial kinematic chains satisfying positional constraints. The mechanisms represent the active chain of a loop. The positional constraint consists in placing the origin of the end-frame into a sphere. This sphere is interpreted as the workspace of a imaginary passive chain. In the first test (at left), the goal is to generate 1000 valid configurations for a $7R$ spatial mechanism (the 7^{th} joint produces a rotation of the end-frame). The difficulty of the problem is increased by decreasing the radius of the sphere. In the second test (at right), the mechanism is made up with revolute joints around arbitrary (non-orthogonal) axes. In this case 10 configurations are wanted. The volume of the sphere is very restricted and the complexity is increased by increasing the length (in number of joints) of the chain.

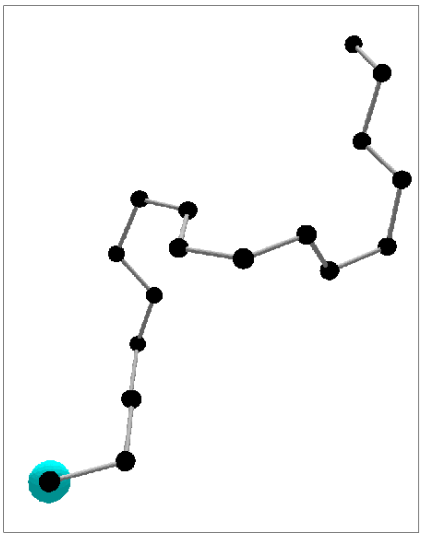
The tables compare the performance of the RLG algorithm versus a uniform random sampling (as used in [Han 01]). The uniform sampling has been implemented using the `rand()` function of the GNU C Library [5]. **N** is the number of sampled configurations and **T** is the computing time in seconds [6]. It can first be noted that, for these examples, RLG always provides configurations satisfying the constraints. The reason is that spheres correspond to the kind of volumes handled by the algorithm for approximating the workspace of subchains. Anyway, as we will show in next chapters, the function SAMPLE_q^a provides a high percentage of feasible configurations for all the examples where we have applied our algorithms, also involving constraints for the orientation of the end-frame. Results

[5] GNU C Library web site: `http://www.gnu.org/software/libc/libc.html`

[6] Tests were performed using a Sun Blade 100 Workstation with a 500-MHz UltraSPARC-IIe processor.



Radius	Uniform		With RLG	
	N	**T**	**N**	**T**
200	29995	0.01	1000	<0.01
150	94959	0.02	1000	<0.01
100	441157	0.25	1000	<0.01
50	3549709	4.31	1000	<0.01
25	26080062	38.72	1000	<0.01



d.o.f.	Uniform		With RLG	
	N	**T**	**N**	**T**
4	155942	0.28	10	<0.01
15	1957901	24.25	10	0.01
30	4063067	363.34	10	0.03
42	$\rightarrow \infty$	$\rightarrow \infty$	10	0.08

Figure 3.8: Two tests for analyzing the performance of RLG.

in tables show that RLG is not much affected by the difficulty of problems. The second example shows its capacity to efficiently handled long chains. The increment of computing time is mainly due to the higher number of operations with matrices required to build the configuration of the chain when the number of joints increases.

These two tests show that higher is the complexity of the problem, better is the relative performance of RLG. Moreover, in these two cases, the computations associated with closure constraints consist in a simple test: is this point into this sphere ? When sampling the configuration of a closed-chain mechanism, loop closure equations must be solved for obtaining q^p form q^a. Then, RLG avoids an enormous number of futile operations which drop off the performance of the planner.

3.3.2 Computing Local Paths

Each sample $q_{new} \in \mathcal{C}_{feas}$ is tested for feasible connections with a set of nodes L_{best} in the roadmap. When the information on the topological structure of $\mathcal{C}$ is available (it is possible for particular classes of mechanisms), then only nodes in the same self-motion set than q_{new} should be candidates for selection into L_{best}. In general, we do not have such information. Thus, the most reasonable criterion is to choose the nearest nodes given a

distance metric in $\mathcal{C}$, and then to check later, during the validity test of the local path, if two nodes lie on the same manifold or not.

In principle, connections between pairs of nodes should be tried using every local parameterization, aiming to undergo singularities (at least those of dimension $M-1$). However, this is not efficient since a lot of computing time will be spent checking the validity of very similar paths on $\mathcal{C}$. Hence, the connection between q_{new} and nodes in L_{best} is tried only using one combination of M joint variables as parameters q^a. When looking for a fast planner, it is important to make the selection of q^a such that q^p can be obtained by an efficient method. In most cases, it is convenient to make the same choice of parameters than for the configuration sampling with RLG.

The function FeasibleConnection applies a steering method onto the parameters q^a. The passive variables q^p follow the variation imposed by loop closure equations along the local path. Figure 3.9 illustrates the different situations that possibly occur when testing the connection between two configurations in the example of Figure 3.1. The figure shows the projection of $\mathcal{M}_1$ and $\mathcal{M}_3$ on the plane $\theta_1\theta_2$, which are the variables chosen as the parameters q^a. In this case, the steering method produces a linear interpolation of these parameters. Configurations q_1 and q_2 are directly connected by a feasible local path, as well as q_1 and q_3. The difference is that the path between q_1 and q_3 goes across a singularity. When testing this local path, a bifurcation for the solution of θ_3 is detected. This point of bifurcation, the singular configuration, allows to connect components of the roadmap on $\mathcal{M}_1$ and $\mathcal{M}_2$. The other local paths illustrated in the figure are not feasible.

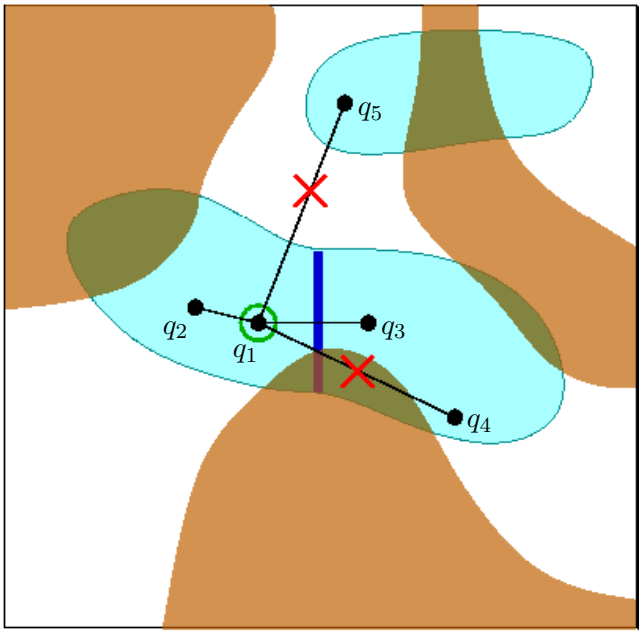


Figure 3.9: Different situations can rise when testing the validity of local paths.

q_1 and q_4 are separated by a C-obstacle that forbids their direct connection. Nevertheless, they can be connected through q_3. On the contrary, q_1 and q_5 will never be connected since they belong to disjoint manifolds.

Dynamic Collision Detection

Verifying that a local path produced by a variation of q^a remains in $\mathcal{C}$ requires to solve the loop closure equations for each intermediate point in this path. For ensuring the validity of a solution, this test must be carried out with a sufficiently high resolution ϵ_{clos}. Making collision detection along the local path with this same constant resolution is very expensive in terms of computing time. Since we do not know a priori the variation of q along a local path, but only the variation of q^a, we can not apply dynamic collision detection algorithms used for open-chain mechanisms. However, we can adopt some of the ideas in these procedures for making a more efficient collision checking along local paths than using constant resolution.

For an intermediate configuration q in a local path τ between configurations q_{new} and $q_{end} \in L_{best}$, we measure de minimum distance d_{coll} between the links of the articulated mechanism and the obstacles. This distance can be obtained using complicated methods that return an exact or very accurate value [Guibas 99, Lin 03], or by approximated methods, less accurate but much more efficient. Using oriented bounding boxes (OBB) [Gottschalk 96] is a good trade-off. When moving on τ from q making n steps with discretization ϵ_{clos}, we obtain consecutive configurations $q_{\epsilon \cdot n}$. We measure the maximum variation of the position of links, d_{move}, between configurations q and $q_{\epsilon \cdot n}$. Collision checking with static obstacles can be delayed until $d_{move} \geq d_{coll}$. For self-collision avoidance, the distance between OBBs of links corresponding to the collision pairs defined for the mechanism can be used in a similar way.

When PRM-based algorithms are applied to open-chain mechanisms, a possible choice for testing the validity of local paths concerns the manner of walking on them. The test can be done incrementally from q_{new} toward q_{end} or using a dichotomic division. In general problems involving closed kinematic chains, when the class of solution of the passive variables cannot be identified, τ has to be tested incrementally. The reason is that the class of solution for a configuration $q_{\epsilon \cdot n}$ corresponding to the current local path can only be identified in relation with the previous configuration $q_{\epsilon \cdot (n-1)}$. For particular cases, if we are able to recognize the different solution classes for q^p, a dichotomic division can be used to test local paths. In general, this kind of test allows to more rapidly identify invalid local paths.

When all the valid configurations q_{new_i} sampled by RLG are kept, several local paths can be tested simultaneously. For the same choice of parameters q^a than q_{new_i}, we have in general n'_{sol} configurations q_{end_j}, $j = 1 \ldots n'_{sol}$, with identical value of these parameters and different value of passive variables q^p. Since loop closure equations have to be solved for the same values of q^a when walking on these "parallel" local paths, it is more efficient to check them simultaneously.

3.4 Extension of RRT-based Algorithms

Most of the explanations on our extension of PRM-based algorithms to handle closed-chain mechanisms can be extrapolated to the RRT approach. In a similar manner, the planner is applied on local parameterizations of $\mathcal{C}$. The basic algorithm of Chapter 2 (Algorithm 2.2) can be used making some considerations that we comment next.

3.4.1 Tree Expansion

The first step in an iteration of the algorithm constructing a search tree is to sample a configuration q_{rand}. In contrast to the PRM approach, this configuration is not aimed to be a new node of the tree. It is only used as a local goal for the exploration. Thus q_{rand} needs not satisfy either closure or other constraints (e.g. collision avoidance). Then, the nearest node q_{near} is tried to be expanded toward q_{rand} on a "local path" linking both configurations. The same considerations as for local paths in the PRM approach can be made here: the appropriate selection of local parameters q^a is important for the performance of the planner. Also the same procedure for collision checking along local paths can be applied within this extended RRT approach.

An uniform sampling of the joint variables is generally used in the RRT-based planners developed for open-chain mechanisms. However, under closure constraints, results of our

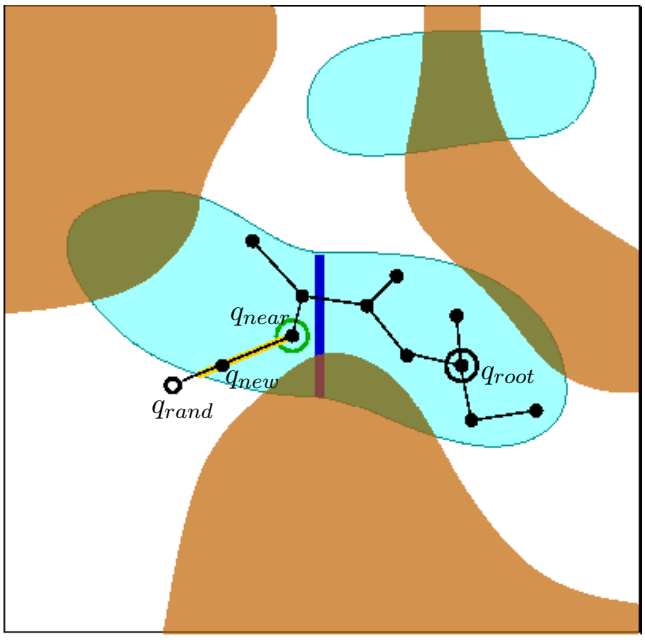
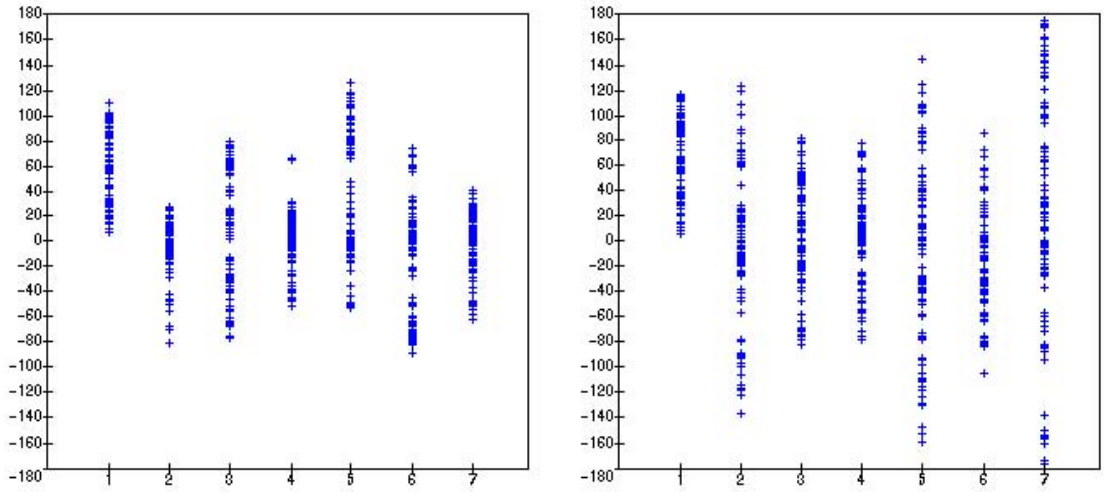


Figure 3.10: Expansion of a node in search tree using an RRT-based algorithm.

tests show that sampling (at least) close to the variety $\mathcal{C}$ improves the exploration. As argued in [Yakey 00, Yakey 01], sampling configurations $q_{rand} \in \mathcal{C}$ is expensive. Nonetheless, we do not need to sample the whole configuration q_{rand}. While we expand the search tree on a local chart, only parameters q^a really require to be sampled. Thus, we should replace q_{rand} by q^a_{rand} in Algorithm 2.2. Note that this implies that the nearest node must be selected using a distance metric only applied to this set of joint variables instead of the whole configuration. If we choose parameterizations as explained in Section 3.2.1, such that the function SAMPLE_q^a of the RLG algorithm can be applied, then we are sure of generating samples into the projection of $\mathcal{C}$ on the hyperplane of the variables q^a, or at least in a neighborhood. Figure 3.11 illustrates the expansion process of a search tree computed for the same example that we have used in explanations of the PRM approach (Figure 3.9). The tree is expanded on a local chart whose parameters are $q^a = \{\theta_1, \theta_2\}$. These parameters are sampled using RLG for generating q_{rand}. Then, the nearest neighbor q_{near} is selected. The new node q_{new} is an intermediate configuration in the feasible portion of the local path between q_{near} and q_{rand}.

Figure 3.11 shows the results of a test proving the better performance of the planner when RLG is applied to generate q_{rand}. The test was made with the $7R$ mechanism of Figure 3.8. It consists in constructing a tree containing 100 nodes exploring the subset of configurations for which the end-effector remains inside the sphere of radius = 50. The figure shows plots of the configurations corresponding to the 100 nodes. Using RLG, the portion of $\mathcal{C}$ covered by the tree is wider than using uniform sampling. Furthermore, sampling with RLG makes the construction faster: 0.68 seconds versus 1.10 seconds with



q with uniform sampling q with RLG

Figure 3.11: Plots of 100 configurations that are the nodes of search trees constructed by an RRT-based algorithm exploring an space under positional constraints.

uniform sampling. The reason of this difference is that the use of RLG reduces the number of iterations of RRT-based algorithms. New potential nodes q_{feas} are rejected when they are too similar to q_{near}, in order to prevent the generation of an excessive number of nodes in some regions. The construction using RLG required to sample 188 configurations while 470 were necessary with the uniform sampling.

3.4.2 Other Issues Under Analysis

In this section, we outline other issues concerning the design of improved RRT-based algorithms. More work is required for extracting determinant conclusions. However, the preliminary results show that we work in a good direction.

Pruning Branches

An important problem of RRT-based algorithms is their difficulty for expanding the search tree beyond "tight corners" of $\mathcal{C}_{feas}$[Yakey 00]. This undesired phenomenon arises because of the use of metrics that do not consider motion constraints for the selection of q_{near}. The problem is accentuated in presence of kinematic closure constraints, since the search-space is more restricted that for open-chain mechanisms. In [Yakey 00], the author tries to solve the problem by adding new nodes to the tree, but it is not clear how to do this. We try to resolve this drawback doing the contrary: removing nodes. Our interpretation of the problem is that, is such difficult situations, the same nodes are repeatedly selected for futile expansion. Defining a heuristic to identify these "saturation" nodes is not easy. However, we can be confident with randomization to erase them. Figure 3.12 shows a simple example [7] that clearly illustrates this unsuitable behavior of the RRT approach. The robot is a box moving on a plane. The problem consists in getting the box out of the saw-like obstacle. The shape of this obstacle hinders the expansion of the search tree. The detail of the top image shows the high density of nodes in the corners. The nodes selected for expansion a highest number of times are marked by a frame. The fact that these nodes correspond to the closest configurations to the obstacle is not hazardous.

We have developed a technique that yields in general good results. The idea is to interleave expansion and pruning phases for constructing the search trees. After a given number of iterations, the expansion of the tree is stopped and we apply what we call a *visibility-pruning*. This process, inspired by the Visibility-PRM [Nissoux 99, Siméon 00], consists in testing connections between the leaves (i.e. extreme nodes) of the tree. Only connections of branches that have been expanded before the last pruning (between them and with the old branches) have to be tested. If two leaves are "visible", one of the corresponding branches is pruned. The choice of the branch to be pruned is made at random. The number of iterations for expansion before pruning can be automatically adapted depending on the evolution of the tree size. The bottom image in Figure 3.12

[7] This example has been made by Leonard Jaillet.

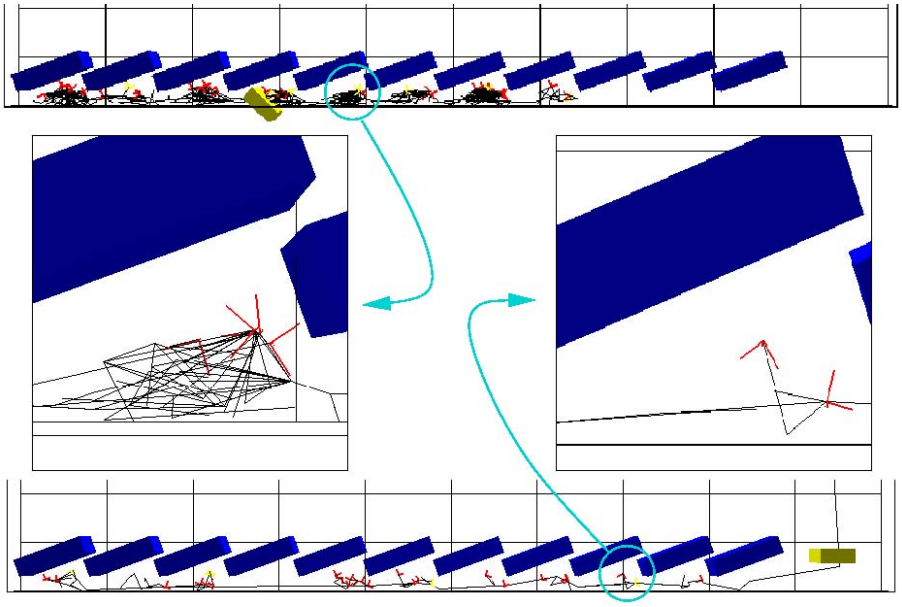


Figure 3.12: The top image shows a dense search tree required to solve an a priori simple problem. The bottom image shows the tree computed with the integration of the visibility-pruning.

shows the search tree computed for this example after integrating the visibility-pruning in the basic RRT algorithm. The number of times that different nodes are selected for expansion is much more homogeneous than without pruning the tree. We have applied this new technique to very different motion planning problems (involving closure constraints or not). The average reduction of computing time is about 30%. We believe the success of this technique is that, by pruning the tree, some problematic nodes are eliminated and we give other ones a chance for expansion. Another advantage of this technique is that the size of the search trees is notably reduced.

Estimating Coverage - Stop Condition

In some applications (see for example Chapter 6), the RRT approach can be used to capture the connectivity of a constrained space, rather that to solve a particular motion planning problem. For this kind of applications, it would be interesting to estimate the amount of space encoded in the search tree. As for the PRM approach, making such a estimation remains a difficult issue. Nevertheless, we can define a simple condition for stopping the algorithm. A counter n_{fail} is introduced for the number of consecutive times the algorithm fails when trying to expand the tree. Then, the iterative process can be interrupted when n_{fail} exceeds a given threshold. Combining this stop condition with the visibility-pruning seems to be a reasonable method to determine the end of the exploration. An analytical or experimental study in order to find a relationship between n_{fail} and coverage, as made in [Nissoux 99, Siméon 00], remains for future work.

3.5 Motion Planning Examples

Here we present motion planning problems involving closed kinematic chains which are solved by the algorithms explained in preceding sections. The PRM approach is applied to one of them and the RRT approach to the others. In both cases, results demonstrate the good performance of these extended planners and the interest of using our configuration sampling algorithm, RLG.

The next examples are solved using one only set of parameters q^a. The configuration sampling, with or without RLG, is applied to this same set of variables. Note that RLG does not require any particular setting. For these examples, RLG has been applied using only the conservative bounds $\hat{r}_{int/ext}$ of the extensions of subchains for obtaining the volumes RWS. The tests have been made with the software Move3D [Siméon 01b], in which our algorithms have been implemented. We have used the same setting of the PRM and RRT planners for all the tests. We do not discuss about this setting here, the goal of the experiments is to compare the performance of the planners with and without incorporating RLG.

In the example illustrated by Figure 3.13, two holonomic mobile manipulators cooperate to transport an object. The system composed by the two manipulators grasping the object is modeled as a single closed kinematic chain. A suitable decomposition of this virtual loop for configuration sampling is to choose one the arms as the passive subchain.

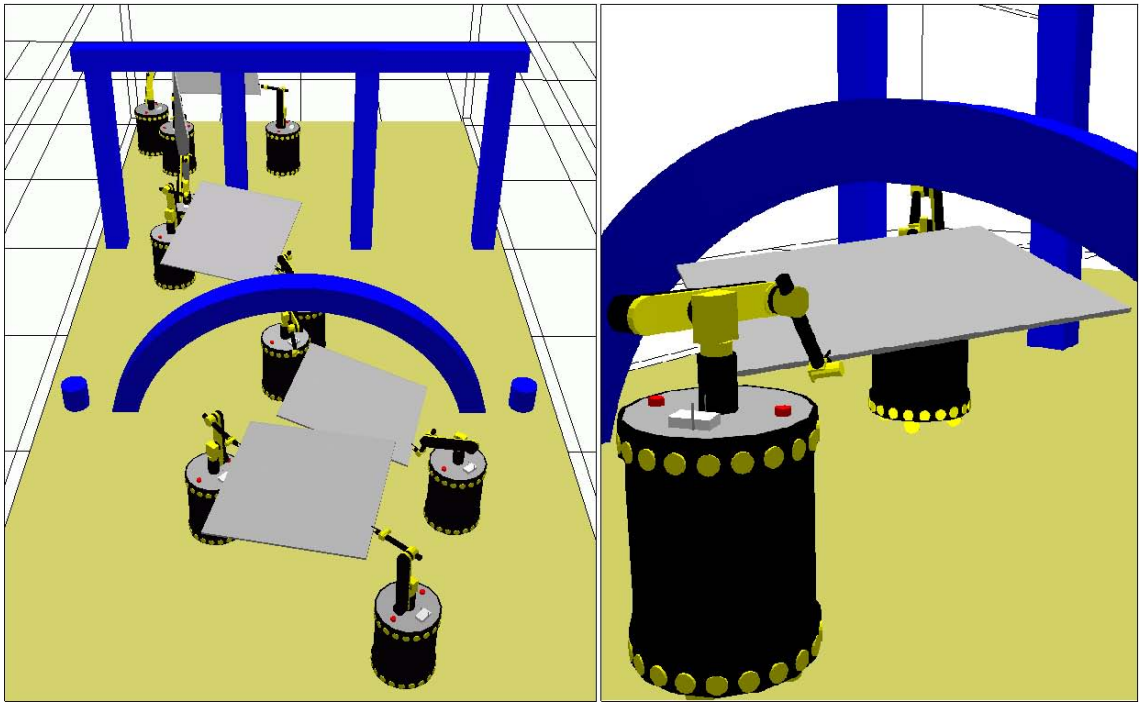


Figure 3.13: Two mobile manipulators coordinate through narrow passages.

Then, the configuration of the mobile robotic platforms, one arm and the rigidly grasped object is generated by the function SAMPLE_q^a. The configuration of the passive arm q^p is obtained by a closed-form inverse kinematics solution. About 50% of the active subchain configurations q^a are reachable by the passive subchain (i.e. the inverse kinematics problem has real solutions). We wanted to known the probability of success using an uniform random sampling for the parameters q^a. If the frames associated with the mobile platforms are independently defined in this workspace, this probability is infinitesimal. However, we can consider that one of the mobile robotic platforms moves relatively to the other. Since these platforms are holonomic, their relative motion can simply be modeled by a prismatic joint. Note that this simplification is not necessary using RLG. Even with an appropriate setting of the limits of the fictive prismatic joint, less than 0.5% of the samples yield a configuration satisfying loop closure constraints. This fact has an important repercussion when solving motion planning problems using the PRM approach.

In this example, avoiding the arc-like obstacle requires precise configurations, with some joint values of the manipulator arms close to their limits (see the right image in Figure 3.13). Consequently, a narrow passage in $\mathcal{C}_{free}$ must be traversed. To find narrow passages using a probabilistic roadmap approach generally needs to try a very high number of random configurations. Using the extended Visibility-PRM algorithm, a roadmap containing 80 nodes was computed that allows to rapidly solve motion planning queries between any two points in the workspace. For this, more than 1000 random configurations in $\mathcal{C}$ were tried. Using RLG the roadmap was computed in 25 seconds, the same process took 3 minutes with the uniform sampling.

Another example involving a virtual closed kinematic chain is illustrated in Figure 3.14. Two robotic arms coordinate for manipulating a twisted bar among two vertical bars that restrict its motion. The goal is to solve a motion planning query between configurations in Figure 3.14.a and Figure 3.14.f. The figure shows a sequence of intermediate configurations of the solution of this puzzle-like problem. The difficulty of this problem depends on the distance between the vertical bars. We have made tests with three settings: 150mm, 175mm and 200mm (the model scale is 1:1). The next table shows results of tests with the extended version of the RRT algorithm using bidirectional search. We have compared the performance of a uniform random sampling versus RLG for generating q^a_{rand}. In this example, q^a_{rand} corresponds to the configuration of one of the arms grasping the bar. **N** is the number of iterations for expanding the search trees. **T** is the computing time. The numerical results have been averaged over a serial of tests. These results show the importance of an appropriate sampling considering the presence of kinematic loops. In addition, let us note that the planner using uniform random sampling returned *Failure* in several tests with $d_{bars} = 150$ because the maximum size allowed for the trees (determining the stop of the algorithm) was reached.

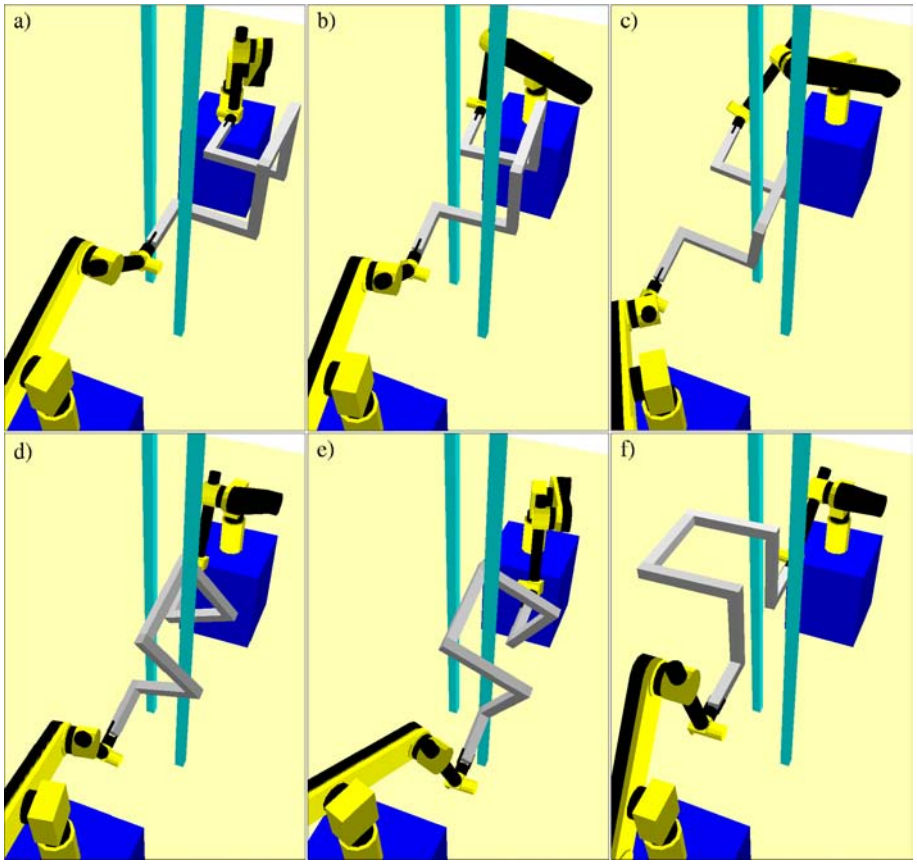


Figure 3.14: Sequence of the solution to a difficult motion planning problem for two robotic arms manipulating an object among obstacles.

d_{bars}	Uniform		With RLG		
	N	**T**	**N**	**T**	**gain T**
200	2719	42.26	118	3.23	× 13
175	4995	102.92	424	6.30	× 16
150	9761	312.76	615	9.14	× 34

Figure 3.15 clearly shows the difference between the two types of sampling. This figure shows the search trees computed for solving a very similar but even more difficult problem than the one above. For representing these trees (computed in a 12-dimensional space), the location of the end-frame of one of the arms corresponding to each node and the connections are displayed. The manipulators have to pass the bar from the one side of the columns to the other side. The bottom-left image shows the two search trees, one rooted at q_{init} and another rooted to q_{goal}, computed with the extended bidirectional-RRT algorithm using uniform random sampling. The planners was unable to find the solution in this case (i.e. the trees are not connected). The bottom-middle image shows the trees computed using RLG for sampling. The algorithm also stopped before finding the solution. However,

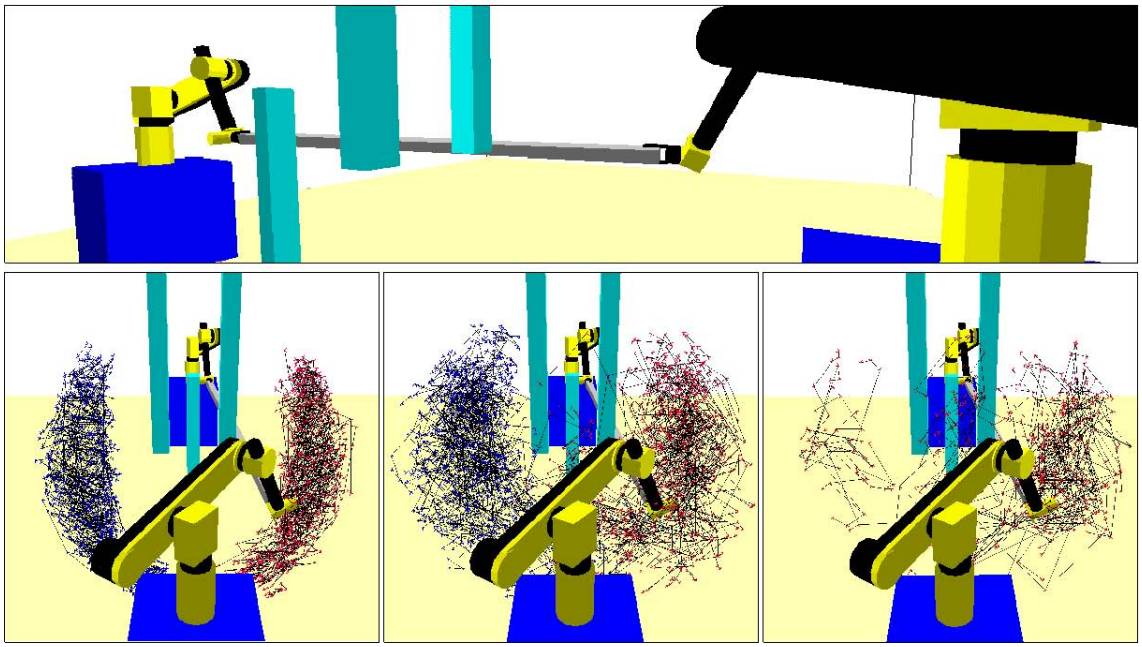


Figure 3.15: Trees computed by the bidirectional version of the RRT algorithm extended to closed kinematic chains. From left to right: using uniform random sampling; sampling with RLG; sampling with RLG and integrating the visibility-pruning.

the image shows that the coverage is much better. A wider region was explored with the same number of nodes (2000) and taking less computing time (180 versus 210 seconds). The bottom-right image shows the trees computed by the RRT algorithm using RLG and the visibility-pruning. In this last case, the solution was found (i.e the two trees met). The reduction of the size of the trees is remarkable: less than 600 nodes are sufficient to solve the problem. The solution was found in less than 2 minutes.

The presented examples correspond to coordinated manipulation problems. We will further discuss the interest of applying our motion planning algorithms within manipulation planning techniques in Chapters 4 and 5. The examples in this section only involve single-loop mechanisms. Nevertheless, our approach can be applied to multi-loops. The problem with the three mobile manipulators and the piano illustrating the introduction of this thesis (Figure 1) could be solved using the techniques explained in this chapter. However, the virtual multi-loops modeling the composite mechanisms has particular characteristics: it can be seen as a parallel mechanism. We have developed a variant of the RLG algorithm for this kind of structures. This extension is explained in Chapter 4.

Finally, it should be noted that the closed-chain mechanisms in the above examples are simple in comparison with other examples in the second part of this thesis. In particular, the portions of molecular chain handled in Chapter 6 can be seen as highly-redundant closed kinematic chains. A single loop can involve several dozens of joint variables. RLG is able to treat efficiently also such complex mechanisms.

3.6 Discussion

All along this chapter, we have explained an approach to extend sampling-based motion planning algorithms to handle closed-chain mechanisms. We have centered particular attention in PRM-based and RRT-based algorithms, but main ideas are applicable to other planners. A considerable effort has been made to give general directives, without focusing on a particular implementation.

The main technical contribution of our work concerns RLG. The "guided"-random sampling performed by this algorithm notably increases the efficiency of motion planners for closed kinematic chains. RLG is general and easy to implement, and its potential applications go beyond motion planning. For instance, it could be used within Monte Carlo simulations applied to molecular chains.

The performance of RLG mainly depends on how well the volumes RWS approximate the workspace of the different subchains. We have given a general procedure to compute these volumes that provides good results in all the examples of mechanisms treated in this thesis. However, more exhaustive tests on different classes of mechanisms are required to extract more precise conclusions about RLG. This work remains for the future. Note that the only tuning parameter in RLG, if necessary, concerns the choice of the distribution between the accurate value and the conservative bound to sample each dimension of the RWS.

A particular attention has been made for keeping completeness properties of the extended planners. If the motion planning problem is formulated within only one self-motion manifold, the extended versions of the PRM and the RRT algorithms remain probabilistically complete. This is possible because RLG is guaranteed to generate samples on the whole variety $\mathcal{C}$ [8]. When a motion planning problem involves several self-motion manifolds, possible connections have to be found through lower-dimensional subsets of singular configurations. In this case, probabilistic completeness can be guaranteed only for a given tolerance related to the resolution at which local paths are discretized. The difficulty represented by the existence of singularities could be overcome if a topological characterization of $\mathcal{C}$ were available. More elaborated algorithms, working separately on the subspaces of different dimension, should be devised exploiting such information. This is unfortunately possible in practice only for particular cases.

We bear in mind an improvement of RLG for a near-future work. It concerns the quality of samples. Our current implementation of RLG applies a "standard" pseudo-random sampling (using the `rand()` function of the GNU C Library). We aim to make RLG properly handle quasi-random sequences in order to get a better coverage of the variety $\mathcal{C}$. This improvement goes in the direction of most recent sampling-based motion planners [LaValle 03b].

Another improvement involves the selection of active and passive variables. In different

[8] We consider the methods for solving inverse kinematics do not miss solutions.

sections of this chapter, we have given guidelines to make this selection for sampling and connecting configurations. However, at this stage, we are not able to propose a general methodology suitable for direct implementation. The choice of parameters has to be made for each particular problem. The study of a general automatic method, based on an analysis of kinematic diagrams of mechanisms, remains for future work.

Part II

Applications

Chapter 4

Motion Planning for Parallel Mechanisms

A parallel mechanism is an articulated structure in which a solid, the end-effector, is connected to the base by at least two independent kinematic chains (see Section 4.2). The most representative parallel mechanism is the 6-d.o.f. parallel manipulator known as the *Gough-Stewart platform* [Gough 56, Stewart 65, Dasgupta 00]. However, the definition of parallel mechanism can also be applied to more complex multi-loop mechanisms. For instance, the virtual structure formed by several manipulators while handling a same object. Our approach for planning the motions of closed-chain mechanisms, described in Chapter 3, can be applied to the most general instances of parallel mechanisms. For increasing efficiency, we have developed a variant of the RLG algorithm for this type of multi-loop structures. It is presented in Section 4.3.

In Section 4.1, we discuss the interest of applying sampling-based motion planning algorithms to parallel mechanisms. These novel techniques are general tools, able to treat problems that currently require ad hoc solutions. Different examples involving complex systems are commented in Section 4.4. Data of the performance of the algorithms solving these problems demonstrate the efficacy of the approach.

4.1 Interest of the Application

The motions of a parallel robot or of several serial manipulators handling an object are affected by the same kind of constraints. In both cases the (real or virtual) structure is a multi-loop with the characteristics that we explain in Section 4.2. Our approach can be applied to solve interesting open problems concerning these robotic systems.

4.1.1 Encoding the Workspace and Planning the Motions of Parallel Manipulators

Parallel manipulators [Merlet 00] have some advantageous characteristics with relation to serial-structure robots, mostly used in industrial applications, such as: high stiffness, high motion accuracy and high load/structure ratio. Thus, they are particularly interesting for handling heavy objects [Bostelman 01], or for operations requiring very high precision, in surgery for example [Lazarevic 97]. The drawback of parallel mechanisms is that their workspace is quite reduced. Besides, the workspace analysis of a given architecture is a difficult problem. The set of reachable positions of the end-effector (also called the platform) is highly dependent on its orientation. Normally, subsets of the workspace are studied separately. Some works address the reachable workspace for a given fixed orientation of the end-effector (e.g. [Gosselin 90a, Masory 95]). Other, are limited to the possible orientations for a given fixed position (e.g. [Merlet 95]). Indeed, only planar mechanisms have been analyzed in fairly general manner [Merlet 98], while spatial structures remain largely ignored [Merlet 99].

Our extended motion planning algorithms could be a useful tool in computer-aided de-

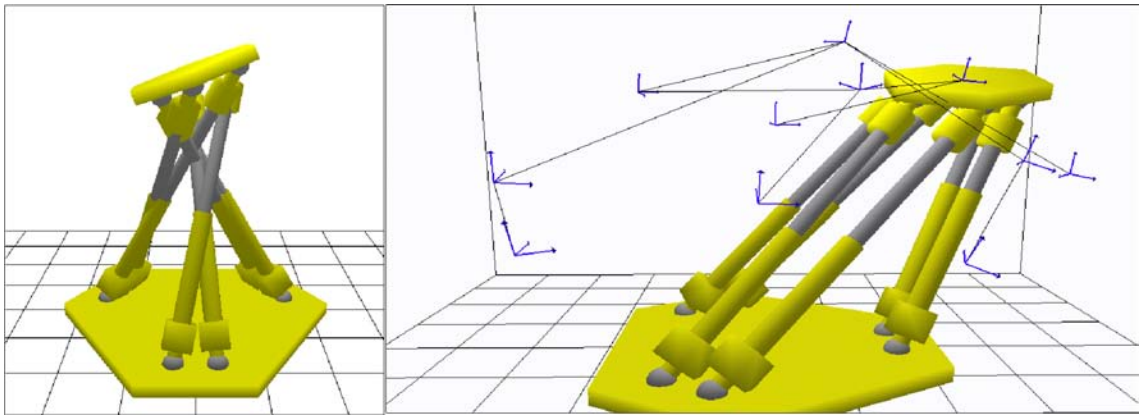


Figure 4.1: Illustration of the visibility-roadmap computed for a Gough-Stewart platform encoding the subset of the self-collision-free configurations.

sign of parallel manipulators. Since there is a direct relationship between the configuration-space of the mechanism and the workspace of the platform (see Section 4.2), the PRM approach can be applied to construct a roadmap that encodes this subset. Indeed, the workspace is implicitly represented by the roadmap. Moreover, using such a technique, not only constraints of the kinematic structure (i.e. loop closure) can be considered, but also other constraints such as self-collision avoidance, which are important for the design. We call the *self-collision-free workspace* $\mathcal{W}_{\mathcal{P}_{sc-free}}$ the subset of possible locations of the platform (often called *poses*) for which there are not collisions between elements of the articulated structure, also called *interferences* by some authors in this field.

Figure 4.1 illustrates the above application. The mechanism is a model of a Gough-Stewart platform. Nodes of the roadmap (i.e. feasible configurations) shown in the right image correspond to possible locations of the platform. The left image represents an invalid configuration, because of collisions between the legs, that leads to an impossible platform location. Such a location would be contained in the workspace when only considering loop closure constraints, but it is not a valid point in $\mathcal{W}_{\mathcal{P}_{sc-free}}$. The roadmap in the figure has been computed by a Visibility-PRM algorithm extended to handle closed-chain mechanisms. It contains only 11 nodes in one connected component. This roadmap covers more than the 99.99% of $\mathcal{W}_{\mathcal{P}_{sc-free}}$ [1].

Despite the increasing interest on parallel mechanisms, as far as we know, no effective general technique has already been proposed to solve motion planning problems [Merlet 02]. Indeed, most of the effort has been focused in the development of techniques for *trajectory verification* (e.g. [Merlet 94, Merlet 01]): verify if a given trajectory of the platform lies completely within the workspace of the robot. Motion planning is however a more difficult problems. It consists in determining if a feasible trajectory of the

[1] According to our tests.

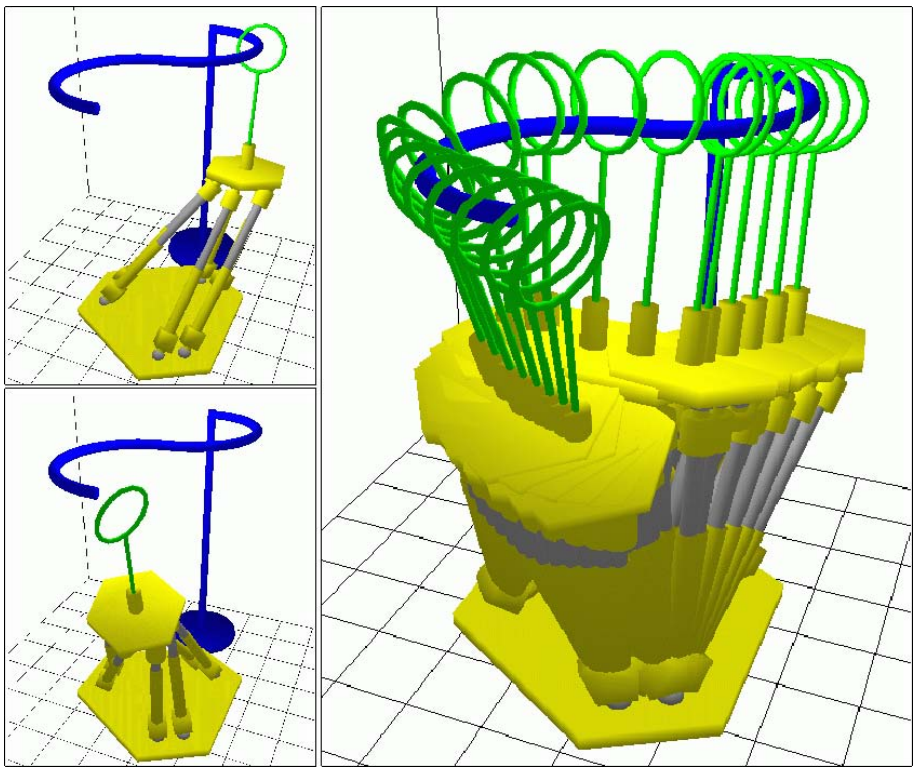


Figure 4.2: Initial an goal configurations, and trace of the solution path of a motion planning problem for a Gough-Stewart platform.

platform exists between two given locations. Global trajectory planning approaches have been proposed for particular classes of mechanisms (e.g. [Lee 96, Dasgupta 98]). Such techniques normally check the presence of singular configurations along the trajectory but most of them neglect collision avoidance. Only a few works consider such motion constraints [Wenger 98, Chablat 98]. However, since they rely on an explicit representation of the workspace of the mechanism, their applicability is limited to simple problems.

Figure 4.2 illustrates a motion planning problem for a Gough-Stewart platform. Finding a solution to this problem with our algorithms only requires the 3D model of the robot and the environment, and the inverse geometric model (i.e. a function providing solutions to the inverse kinematics -existence- problem) for the legs of the platform, which is straightforward in this case. The path to extract the ring mounted on a Gough-Stewart platform from the s-shaped obstacle is computed in only a few seconds. In Section 4.4 we give numerical data of the performance of PRM-based and RRT-based algorithms when solving this problem and other more difficult ones. As we will show, the generality of these techniques allow to handle complex structures made up with serial or parallel associations of Gough-Stewart platforms.

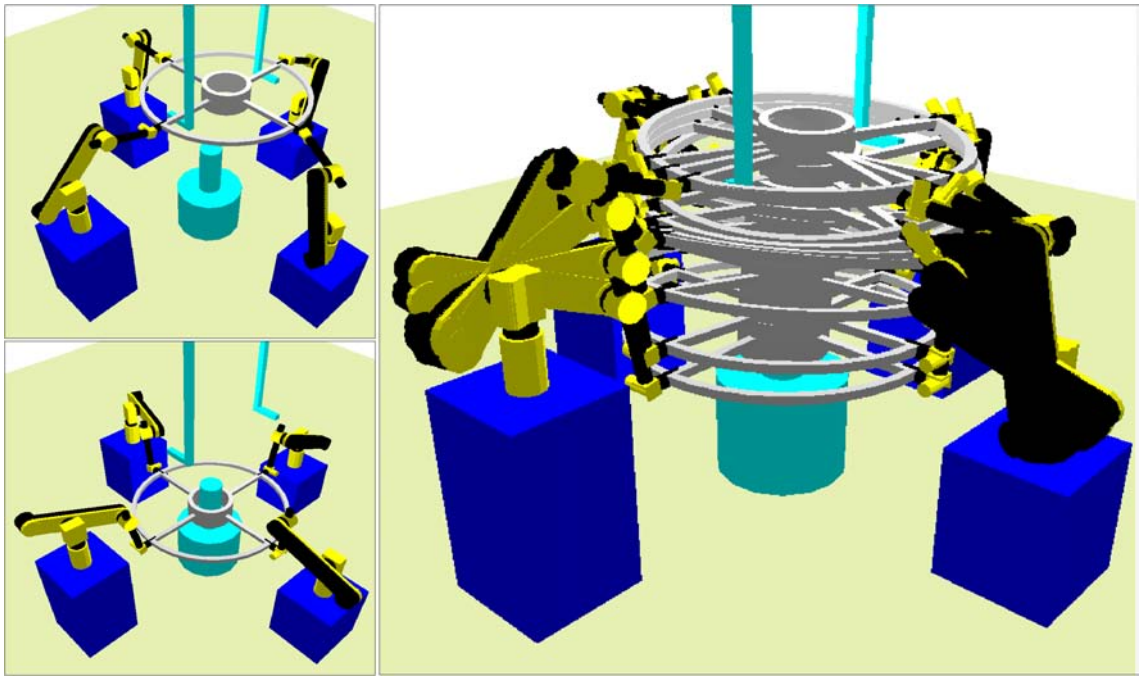


Figure 4.3: Four robotic arms manipulating an object.

4.1.2 Solving Coordinated Manipulation Planning Problems

Coordinated manipulation requires to compute the motion of several robots simultaneously handling an object. In such a situation, a virtual closed-chain mechanism is formed. The grasps of the movable object can be modeled either as a fixed attachment (i.e. the object and the gripper form an only rigid body) or by a joint, if some degrees of freedom are allowed between the gripper and the object. Look for example at Figure 4.3. The four robotic arms and the object compound an articulated structure that could be considered a parallel mechanism. Indeed, if we consider firm grasps, the composite mechanism has mobility $M = 6$, and according to equation (1.5) the degree of parallelism is $P = \frac{3}{5}$, thus, it is a partially parallel mechanism. The analogy with a parallel robot is obvious: the movable object correspond to the platform and the robotic arms to the legs of the mechanism.

Problems for coordinated multi-arm manipulation have mostly been formulated from the point of view of control [Khatib 88, Schneider 92, Desai 99, Chang 00]. The motion planning issue has been rarely tackled. The approach closely related to ours was presented in [Koga 94]. A motion planning algorithms is proposed that computes paths for a movable object grasped by several robotic arms. It is a worthy pioneer work in the application of randomized motion planning algorithms to solve manipulation planning problems. However, the method for computing these coordinated constrained motions is rather naive.

A collision free path is first computed for the movable object (using the RPP algorithm) regardless loop closure constraints inferred by the manipulators. Then, the possibility of maintaining grasps along this path is checked. Obviously, even unconstrained motions of a free-flying object can difficultly be followed by the end-effectors of several arms simultaneously. And, normally, the higher the number of manipulators, the lower the chance of generating a feasible paths for the object while grasped by all of them.

Using our approach, motions of the multi-loop mechanism are also commanded (at least partially) by the movable object. The degrees of freedom of the (virtual) platform are chosen as configuration parameters q^a within our method. However, these motions are generated bearing in mind the architecture of the manipulators (i.e. closure constraints). Besides, the algorithms we have presented in Chapter 3 are more general than the technique in [Koga 94], since no restriction is imposed for the manipulators, while they have to be non-redundant in this referred work.

In this chapter, we only deal with the motions of the composite closed-chain mechanism: robots & movable object. However, the main interest of our technique is its integration into more complicated algorithms for planning manipulation tasks (as made in [Koga 94]). Chapter 5 shows a successful application of our algorithms for planning the motions of a single-loop mechanisms within a manipulation planner for one robot and one movable object. The present extension for multiple robots and one object could be integrated into more sophisticated manipulation planning approaches, such as the combined geometric/symbolic planner explained in [Gravot 02, Gravot 03].

4.2 Description of Parallel Mechanisms

Elements: A parallel mechanism is composed of a base $\mathcal{A}_0$, a platform $\mathcal{P}$ and n_k kinematic chains $\mathcal{K}_i$ [2] linking them. We call F_{b_i} and F_{e_i} the frames corresponding to the connections of each chain $\mathcal{K}_i$ to $\mathcal{A}_0$ and $\mathcal{P}$ respectively. $F_{\mathcal{A}_0}$ and $F_{\mathcal{P}}$ are the frames associated with $\mathcal{A}_0$ and $\mathcal{P}$ (see Figure 4.4).

Pose: The spatial location (or pose) of $\mathcal{P}$ is defined by a vector:

$$q_{\mathcal{P}} = \{x_{\mathcal{P}}, y_{\mathcal{P}}, z_{\mathcal{P}}, \gamma_{\mathcal{P}}, \beta_{\mathcal{P}}, \alpha_{\mathcal{P}}\}.$$

The three first elements represent the position of $F_{\mathcal{P}}$ relative to $F_{\mathcal{A}_0}$. The orientation is given by three consecutive rotations around the coordinate axes of $F_{\mathcal{P}}$ [3].

Workspace: The platform $\mathcal{P}$ is considered to be the end-effector of the parallel mechanism. Hence, the workspace $\mathcal{W}_{\mathcal{P}}$ is defined as the subset of $SE(3)$ mapped by $F_{\mathcal{P}}$. In other

[2] We simplify notation with relation to preceding chapters, the chains should be denoted by ${}^{b_i}\mathcal{K}_{e_i}$.

[3] The approach is valid for other parameterizations of the orientation (e.g. Euler angles).

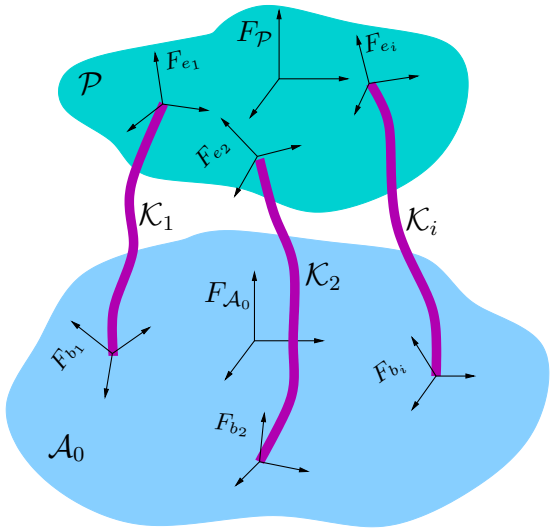


Figure 4.4: General description of a parallel mechanism.

words, $\mathcal{W}_{\mathcal{P}}$ corresponds to the set of all the possible poses $q_{\mathcal{P}}$ of the platform. The bounds of this region depend on the workspaces $\mathcal{W}_{\mathcal{K}_i}$ of the chains $\mathcal{K}_i$, and on the dimensions of $\mathcal{P}$.

Configuration: We consider the configuration q of a parallel mechanism is defined by the platform pose and the configuration of the chains $\mathcal{K}_i$:

$$q = \{q_{\mathcal{P}}, q_{\mathcal{K}_1}, \ldots, q_{\mathcal{K}_{n_k}}\}$$

Therefore, the workspace $\mathcal{W}_{\mathcal{P}}$ can be directly extracted from the configuration-space $\mathcal{C}$.

4.3 RLG Variant for Parallel Mechanisms

We make a particular treatment of parallel mechanisms with relation to what has been explained in Section 3.2 for general multi-loop systems. Next we describe a proper methodology for determining active and passive variables of the configuration of these structures. Such a selection of parameters is used for the sampling of configurations by a variant of the RLG algorithm. Normally, the same parameterizations are suitable for computing connections with local paths when computing a roadmap with a PRM-based algorithm, or for expanding a search tree with a RRT-based one. In all the examples shown in Section 4.4, the motion planning problems are solved using the same set of parameters for both sampling and connecting configurations.

4.3.1 Mechanism Decomposition - Choice of Parameters

The parameters defining the platform pose $q_{\mathcal{P}}$ are selected as active variables. Once the platform pose is defined, the chains $\mathcal{K}_i$ have to be treated as single-loop closed kinematic chains between frames F_{b_i} and F_{e_i}. When the chains $\mathcal{K}_i$ are non-redundant kinematic chains (e.g. the legs of a Gough-Stewart platform or the robotic arms of the example in Figure 4.3), then there are no other active variables than $q_{\mathcal{P}}$. In general, the chains $\mathcal{K}_i$ can be redundant. Each one of them can be decomposed as described in Section 3.2 for a single loop. Thus, we can separate joint variables as follows: $q_{\mathcal{K}_i} = \{q^a_{\mathcal{K}_i}, q^p_{\mathcal{K}_i}\}$. Hence, variables defining the configuration q of a parallel mechanism are divided into active and passive such that:

$$q^a = \{q_{\mathcal{P}}, q^a_{\mathcal{K}_1}, \ldots, q^a_{\mathcal{K}_{n_k}}\}$$
$$q^p = \{q^p_{\mathcal{K}_1}, \ldots, q^p_{\mathcal{K}_{n_k}}\}$$

4.3.2 RLG_Parallel Algorithm

We give the pseudocode of the algorithm generating random configurations of a parallel mechanism in Algorithm 4.1. First, $q_{\mathcal{P}}$ is computed by the function Sample_$q_{\mathcal{P}}$ that we detail later. This function performs a guided-random sampling, similarly to function Sample_q^a (Algorithm 3.2). The goal is to make the platform reachable by all the chains $\mathcal{K}_i$ simultaneously. The configuration of these chains, that must satisfy loop closure constraits, is computed by the algorithm SingleLoop_RLG (Algorithm 3.1). In the case of non-redundant chains, the inverse kinematics solutions are directly returned. For a Gough-Stewart platform, a simple analytical solution provides normally a single configuration for each leg. An analytical solution is also possible for determining the configuration of the robotic arms in the example of coordinated manipulation. However, in this case there are up to eight solutions. The analytical method allows to discern between the different

Algorithm 4.1: RLG_Parallel

input : the robot $\mathcal{A}$
output : the configurations $q[n_{sol}]$
begin
  $q_{\mathcal{P}} \leftarrow$ Sample_$q_{\mathcal{P}}(\mathcal{A})$;
  for $i = 1$ ***to*** n_k **do**
    $q_{\mathcal{K}_i}[n_{sol_i}] \leftarrow$ SingleLoop_RLG$(\mathcal{K}_i)$;
    if $n_{sol_i} = 0$ **then return** Failure;
  $q[n_{sol}] \leftarrow$ CompoundConf$(\mathcal{A}, q_{\mathcal{P}}, q_{\mathcal{K}_1}[n_{sol_1}], \ldots, q_{\mathcal{K}_{n_k}}[n_{sol_{n_k}}])$;
end

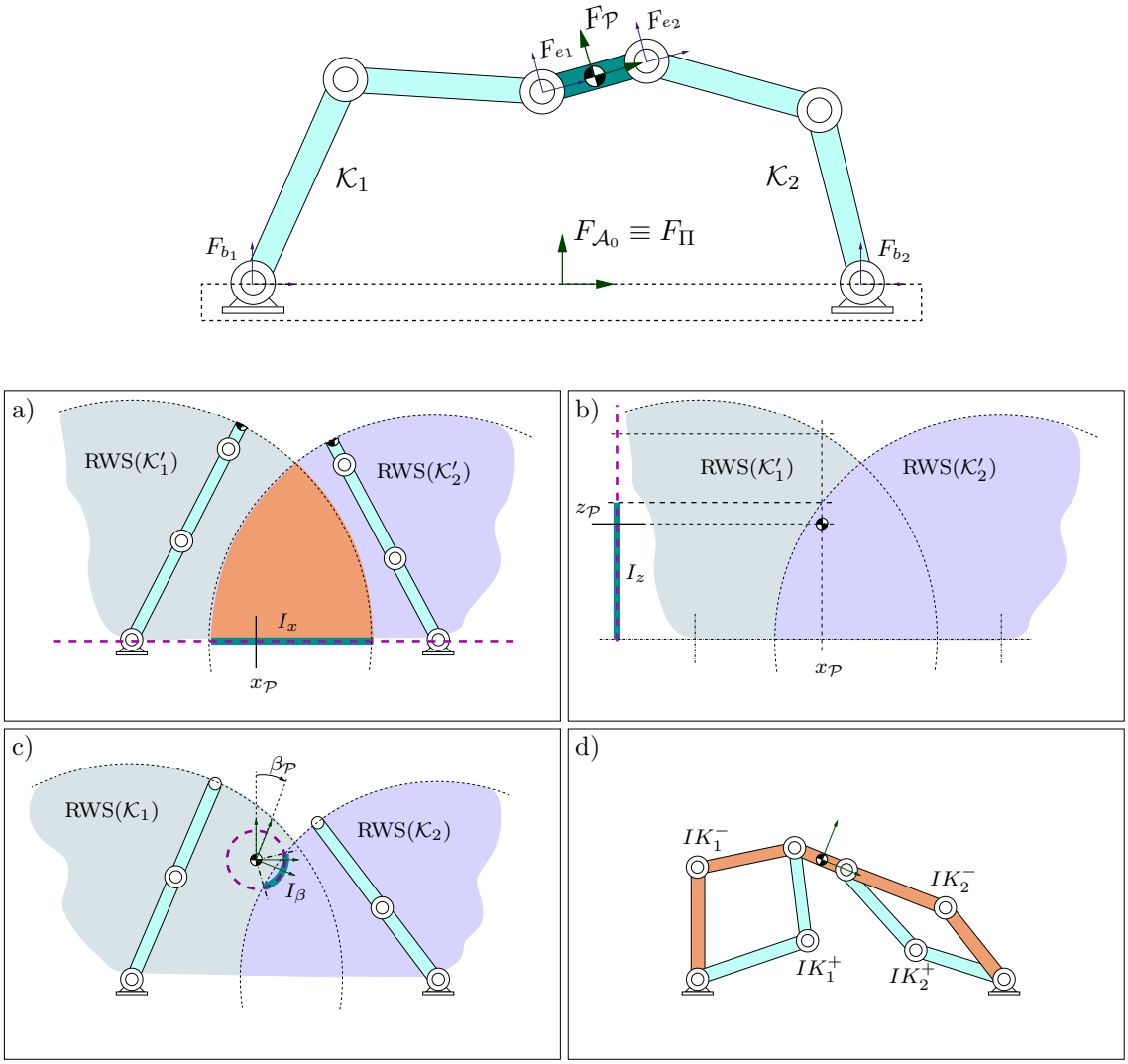


Figure 4.5: Decomposition of a $6R$ planar linkage seen as a parallel mechanism, and steps of the RLG_Parallel algorithm performing on it.

classes of solution. When $\mathcal{K}_i$ is redundant, SingleLoop_RLG returns a finite number upon the infinite feasible configurations. These configurations correspond to a sampled value for $q^a_{\mathcal{K}_i}$ and the n_{sol_i} inverse kinematic solutions for $q^p_{\mathcal{K}_i}$.

Sampling the Platform Pose

The function Sample_$q_{\mathcal{P}}$ acts in a very similar way than Sample_q^a. The parameters are sampled progressively from the computed closure range approximations. The main difference is that the closure range depends now on the satisfaction of closure constraints

that simultaneously involve several individual loops. Also in this case, simple volumes, that we called RWS, bounding the reachable workspace of chains $\mathcal{K}_i$ are used in the process. The algorithm first generates the position parameters of $q_{\mathcal{P}}$ and then it computes the orientation parameters. We describe below these two steps. The explanations are illustrated on Figure 4.5, which shows the different steps of the algorithm RLG_Parallel applied to a $6R$ planar linkage. This single-loop mechanism can also be seen as a simple parallel mechanism: the middle link is the platform $\mathcal{P}$, which is connected to the base $\mathcal{A}_0$ by two chains, $\mathcal{K}_1$ and $\mathcal{K}_2$. These chains are non-redundant $3R$ planar mechanisms.

Platform Position: All the frames F_{b_i} have fixed location with respect to $F_{\mathcal{A}_0}$. Then, a plane Π can be computed by interpolating the position of the frame origins O_{b_i} (when $n_k > 3$). The platform position relative to Π is generated in two steps: first coordinates x_Π and y_Π, and then coordinate z_Π. Since the coordinates transformation from a frame F_Π associated with this plane to $F_{\mathcal{P}}$ is constant, $x_{\mathcal{P}}$, $y_{\mathcal{P}}$ and $z_{\mathcal{P}}$ are directly obtained from x_Π, y_Π and z_Π.

Coordinates x_Π and y_Π are sampled in a rectangular region I_{xy} which is computed as follows. Let us call $\text{RWS}(\mathcal{K}'_i)$ the bounding volumes of the individual chains $\mathcal{K}_i$ considering $F_{\mathcal{P}}$ as the end-frame, instead of F_{e_i}. I_{xy} is a rectangle bounding the intersection of the projections of the $\text{RWS}(\mathcal{K}'_i)$ on Π. Thus, I_{xy} is a conservative approximation of the orthogonal projection of $\mathcal{W}_{\mathcal{P}}$ on Π. Figure 4.6 illustrates the computation of I_{xy} when the volumes RWS are spherical shells. For clarity purpose, we have only represented the external surface of one of these volumes.

The coordinate z_Π is then computed by considering the intersections of the line perpen-

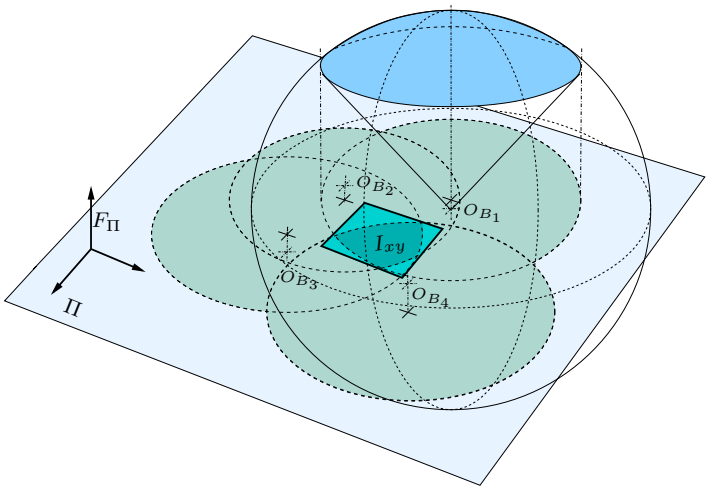


Figure 4.6: Illustration of the operations to obtain I_{xy} for a general spatial parallel mechanism with four legs.

dicular to Π passing through the sampled a point $\{x_\Pi, y_\Pi\}$ with each volume RWS($\mathcal{K}'_i$). When one or several volumes are not intersected, a new point in I_{xy} can be resampled a given number of times. The common intersection between this line and all the volumes defines one or several intervals I_z in which we sample z_Π.

Platform Orientation: For a given position of $\mathcal{P}$, its orientation is generated by progressively computing the three elementary rotations around the coordinate axes of $F_\mathcal{P}$: $\gamma_\mathcal{P}$, $\beta_\mathcal{P}$ and $\alpha_\mathcal{P}$.

The rotation of $\mathcal{P}$ around the x-axis of $F_\mathcal{P}$ produces a circular motion the origin O_{e_i} of each frame F_{e_i}. The intersection of this (segment of) circle with each volume RWS($\mathcal{K}_i$) is computed as described in Section 3.2.3 for the single-loop version of RLG (see Figure 3.7 where the origins O_e and O_c are respectively replaced by O_{b_i} and O_{e_i}). We call I_{γ_i} the result of the individual intersections. When none of the I_{γ_i} is null, $\gamma_\mathcal{P}$ is sampled in the common intersection of these sets, noted I_γ. The same process is repeated for $\beta_\mathcal{P}$ and $\alpha_\mathcal{P}$, bearing in mind that $F_\mathcal{P}$ must be reoriented after each step.

4.3.3 Associations of Parallel Mechanism

The presented technique can also be extended to handle more complex systems obtained by the associations of parallel mechanisms.

When n parallel mechanisms are connected in ***series*** (see Figure 4.7), each platform $\mathcal{P}_i$, $i = 1 \ldots n-1$, becomes the base for the next platform. The process for generating the configuration is progressively carried out for each platform, from the base ($\mathcal{P}_1$) to the top ($\mathcal{P}_n$). When the sampled pose of a given $\mathcal{P}_i$ is not valid, the process does not restart completely; it is only iterated from $\mathcal{P}_{i-1}$.

When several parallel mechanisms are disposed in ***parallel*** (see in Figure 4.8) they form a "multi-parallel" system. Each mechanism becomes a chain $\mathcal{K}_i$ of the main system. Therefore, their platforms are passive elements. RWS($\mathcal{K}_i$) is obtained by intersecting the RWS of the legs of the individual platforms. In general, this volume can also be approximated by a spherical shell.

4.4 Results

Our approach has been implemented in the generic motion planning software **Move3D** [Siméon 01b]. In this section we comment some of the results obtained for very different parallel mechanisms. Computing times correspond to tests performed using a Sun Blade 100 Workstation with a 500-MHz UltraSPARC-IIe processor. Numerical results of each example have been averaged over 10 runs.

4.4.1 Parallel Manipulators

The first experiment aims to demonstrate the performance of the approach to compute self-collision-free motions of the Gough-Stewart platform. The roadmap of the example in Figure 4.1, practically covering the whole subset $\mathcal{W}_{\mathcal{P}sc-free}$, was computed in 22 seconds by the closed-chain extension of the Visibility-PRM algorithm. The construction required the generation of 17442 configurations of the mechanism, of which 2328 were found to be collision-free. Using our sampling strategy, RLG, the function SAMPLE_$q_{\mathcal{P}}$ was called 30840 times. That means more than 50% of success for sampling random configurations of the Gough-Stewart platform satisfying closure constraints. We made similar tests using uniform random sampling to generate the platform pose [4]. In this case, less than a 2% of the samples produced valid configurations of the mechanism. This fact yields an important detriment to the performance of the planner. Constructing a similar roadmap is about 25 times slower !

Once the roadmap is computed, this data structure can be used to generate almost in real-time feasible motions avoiding collisions between the parts of the mechanism. In presence of obstacles, such a roadmap approach also allows us to solve motion planning problems like the one illustrated in Figure 4.2. This problem is particularly hard for a sampling-based approach because the motion getting the ring out of the s-shaped bar requires extreme deformations of the Gough-Stewart platform. A roadmap containing the solution path was computed in 60 seconds.

The two next examples exploit the capacity of sampling-based motion planning algorithms to solve problems in very-high-dimensional spaces. The manipulator in Figure 4.7 is a model of the Logabex-LX4 [Charentus 90]. The structure is composed of four Gough-Stewart platforms connected in series and an end-effector provided of a single wrist that allows one rotation. The model of each individual Gough-Stewart platform involves 24 joint variables: 3 for each leg [5] and 6 for the platform. Thus, the dimension of the joint-space $\mathcal{Q}$ of this robot is $m = 97$. The mobility (i.e. the dimension of the manifolds embedded in $\mathcal{Q}$ that compose $\mathcal{C}$) is $M = 25$. Planning queries for moving the manipulator with the grasped bar from one to another opening of the bridge were solved by the extended RRT-based algorithm in less than one minute (about 50 seconds in average). In the example illustrated by Figure 4.8, two teams of three Gough-Stewart platforms cooperate in an assembly task. This type of association has been proposed in [Chai 01] for the assembly of large and heavy components in industry. The motion to assemble the two puzzle-like pieces was computed in only 15 seconds by the RRT-based algorithm. Remark in the figure that the clearance between the pieces is very small.

[4] Samples are taken in a 6-dimensional box bounding the subset of feasible poses of $\mathcal{P}$.

[5] The joint variables corresponding to the sperical joint between each leg and the platform need not be considered in the model.

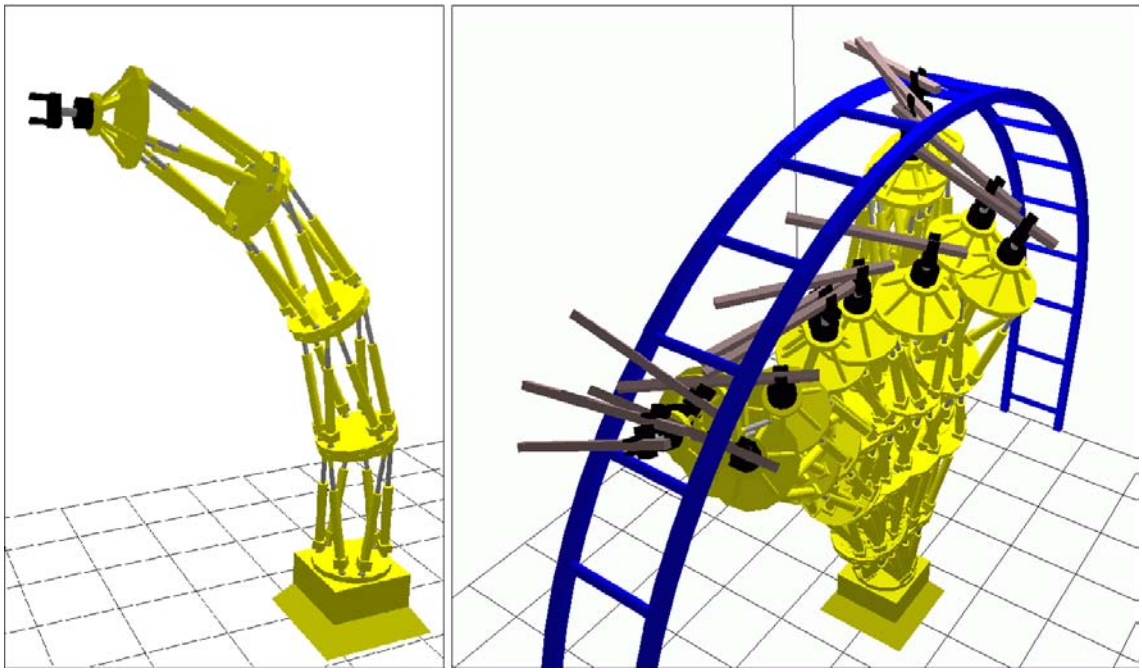


Figure 4.7: Model of the robot Logabex-LX4 and trace of a collision-free path.

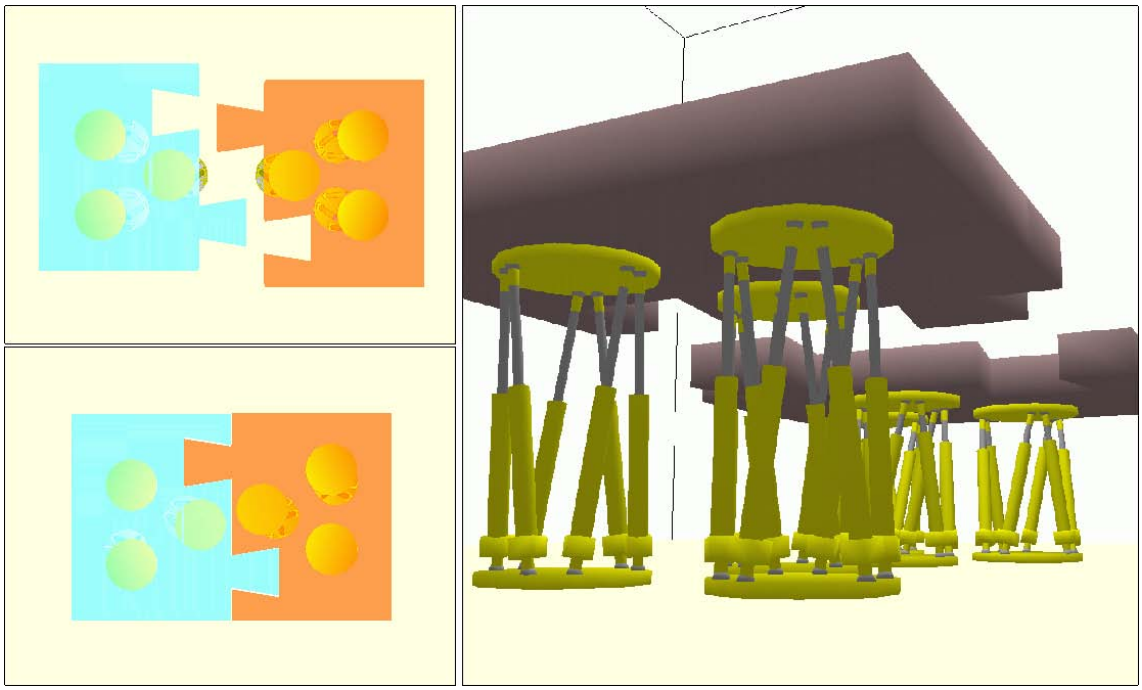


Figure 4.8: Two teams of three Gough-Stewart platforms have to assemble large puzzle-like pieces.

4.4.2 Coordinated Manipulation

Tests made with the model in Figure 4.3 continue demosntrating the good performance of RLG. Note that the number of joint variables in this model is $m = 30$ (6 for each manipulator and 6 for the movable object), and the mobility is $M = 6$. For generating 100 random configurations of the virtual parallel mechanism composed by the four $6R$ manipulators and the movable object, 326843 locations of the platform generated by uniform sampling were necessary. Using RLG, only 107 random poses were computed. In terms of computing time, the former test took 75.12 seconds, while the latter only took 0.19 seconds. The problem illustrated in the figure, where the manipulators have to unhook an object and to insert it into the cylindrical axis, was solved using both approaches, PRM and RRT. For computing a roadmap containing the solution path to this problem, more than 4000000 poses generated by uniform random sampling were necessary and the process took more than 20 minutes. Using RLG for sampling configurations, less than 500 random platform poses were generated and the roadmap was built in less than 20 seconds. The difference is less important when applying the RRT approach on this example. We used a bidirectional search to solve the problem. The process for expanding the trees was iterated 95 times using a uniform random sampling for q^a_{rand} versus 43 when calling the function SAMPLE_$q_{\mathcal{P}}$. The computing times for obtaining the solution path were 1.78 seconds and 0.99 seconds respectively.

Figure 4.9 illustrates a similar problem in which one of the robots has been replaced by a human-like character. The human arm is modeled as a $7R$ articulated mechanism. Thus, it is kinematically redundant. However, the inverse geometric model in our current implementation (inspired by IKAN [Tolani 00]), handles this redundancy. For a desired location of the hand, only one inverse kinematics solution is returned corresponding to a realistic posture. Therefore, the human arm is considered non-redundant for the planner,

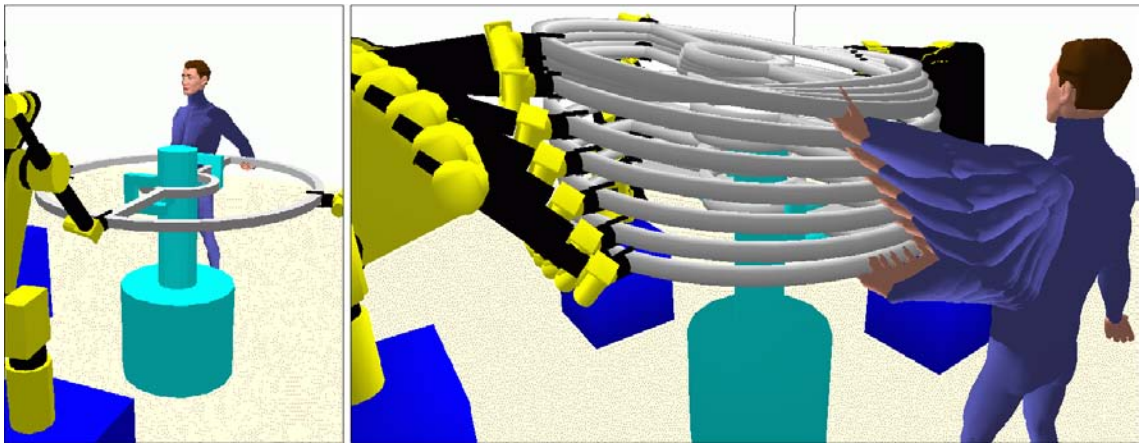


Figure 4.9: Human-like character interacts with robots in a manipulation task.

so that the only active variables are the parameters $q_{\mathcal{P}}$, as in the preceding example. The motion shown in the figure was computed in a few seconds by RRT.

The last results we comment in this chapter correspond to the example opening this thesis: the version of the piano mover's problem with three mobile manipulators (see Figure 1). This problem combines two types of difficulty with relation to the previous examples. First, the "legs" of the virtual parallel structure are redundant, since they correspond to the mobile manipulators. And second, the complexity of the scene makes collision checking of local paths very hard. Besides, obstacles are strategically placed in order to hinder the motion of robots for changing the orientation of the piano (the piano is rotated of 180° between the initial and goal locations). Note that the piano pedals restrain the classes of feasible trajectories. The redundancy is efficiently handled by RLG. Parameters $q^a_{\mathcal{K}_i}$ correspond to the mobile bases and passive variables $q^p_{\mathcal{K}_i}$ to the $6R$ arms. A roadmap that permits to solve this motion planning problem and most other possible queries in this scene was computed using the extended Visibility-PRM algorithm in 5 minutes.

4.5 Discussion and Prospects

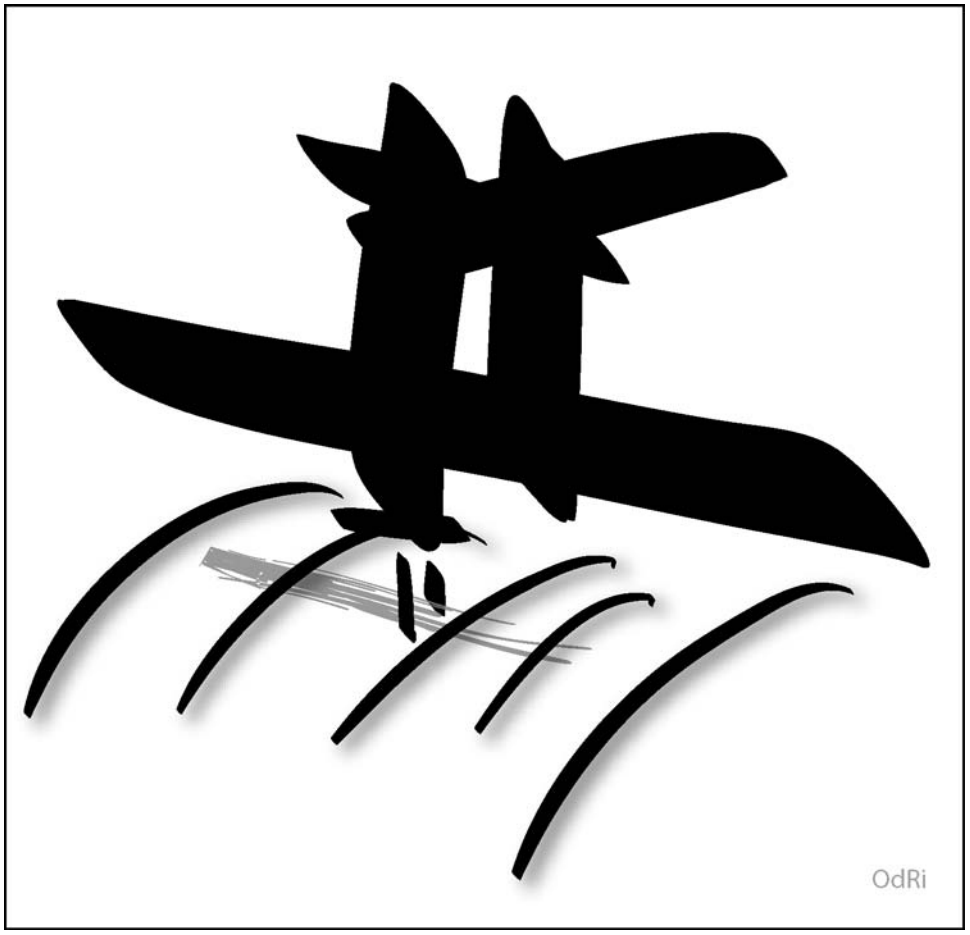
We have seen along this chapter how the approach described in Chapter 3 allows to extend PRM-based and RRT-based algorithms to solve efficiently motion planning problems involving complex mechanisms with multiple loops. Although it was suggested in previous related works [Han 01, LaValle 99, Yakey 01], to the best of our knowledge, this is the first time sampling-based motion planning algorithms are effectively applied to parallel mechanisms.

The major improvement to do concerning the applications to parallel manipulators is to include a more efficient method to validate local paths. This is not a hard task, since trajectory verification algorithms (e.g. [Merlet 01]) can be directly applied on these paths. The presence of singular configurations can be simultaneously checked in the verification process.

The general definition of parallel mechanisms we have adopted makes our techniques applicable to systems more complex than parallel (Gough-Stewart-like) robots. Sophisticated manipulation planning methods (e.g. [Gravot 03]) could integrate our geometric algorithms for solving parts of complicated problems involving several robots and several movable objects in the same workspace.

Chapter 5

Manipulation Planning



Manipulation planning concerns the automatic generation of the sequence of robot motions allowing to manipulate movable objects among obstacles. The presence of movable objects leads to a more general and computationally complex version of the motion planning problem than the instances formulated in Chapter 1. Indeed, the robot has the ability to modify the structure of its configuration-space depending on how the movable object is grasped and where it is released in the environment. Motion planning in this context appears as a constrained instance of the coordinated motion planning problem. The solution of a manipulation planning problem (see for example [Alami 89, Latombe 91]) consists of a sequence of sub-paths satisfying such restrictions: movable objects cannot move by themselves; either they are transported by robots or they must rest at some stable placement. Motions of the robot holding the object at a fixed grasp are called *transfer paths*, and motions of the robot while the object stays at a stable placement are called *transit paths*.

Consider the manipulation planning example illustrated by Figure 5.1. The manipu-

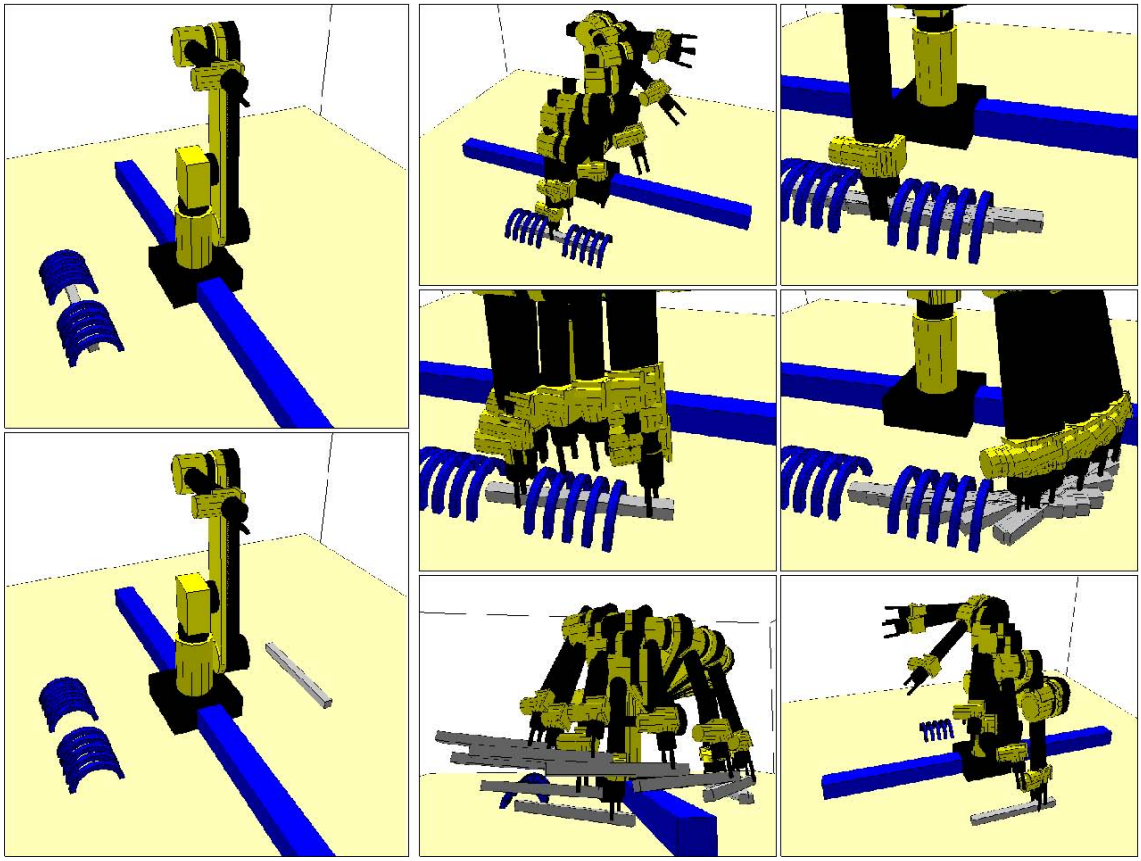


Figure 5.1: How to manipulate the bar from its initial position (top, left) to the goal (bottom, left)? The solution (right) requires several *pick and place* operations.

lator arm has to get a movable object (the bar) out of the cage, and to place it on the other side of the environment. Solving this problem requires to automatically produce the sequence of transit/transfer paths separated by grasp/ungrasp operations, allowing to get one extremity of the bar out of the cage; the manipulator can then re-grasp the object by the extremity that was made accessible by the previous motions, perform a transfer path to extract the bar from the cage, and finally reach the specified goal position. In particular, the motion shown in the second image illustrating the solution requires itself four re-grasping operations to obtain a sufficient sliding motion of the bar. This example shows that a manipulation task possibly leads to a complex sequence of motions including several re-grasping operations. A challenging aspect of manipulation planning is to consider the automatic task decomposition into such elementary collisions-free motions.

Most of existing algorithms (see Section 5.1) assume that finite sets of stable placements and possible grasps of the movable object are given in the definition of the problem. Consequently, a part of the task decomposition is thus resolved by the user since the initial knowledge provided with these finite sets has to contain the grasps and the intermediate placements required to solve the problem. Referring back to the example, getting the bar out of the cage would require a large number of grasps and placements to be given as input data.

The approach described in this chapter deals with a continuous setting of the manipulation problem while covering the scope of the previous proposed approaches. It allows us to devise a manipulation planner that automatically generates among continuous sets the grasps and the intermediate placements required to solve complicated manipulation problems like the one in Figure 5.1. The approach relies on a topological property first established in [Alami 89] and recalled in Section 5.2. This property allows us to reduce the problem by characterizing the existence of a solution in the lower-dimensional subset of configurations where the robot *grasps* the movable object *placed* at a stable position. There exists a bijective map between this subset and the configuration-space of a virtual closed-chain mechanism composed by the robot grasping the movable object affected by the placement (stability) constraints. Thus, we can apply the algorithms explained in Chapter 3 to compute a roadmap encoding the connectivity of this subset. Section 5.3 describes the proposed approach and Section 5.4 details the planning techniques developed to implement the approach. Finally, Section 5.5 presents experimental results and comments on the performance of the planner. Complementary information on this approach and its implementation can be found in [Siméon 01a, Sahbani 02, Siméon 03, Sahbani 03, Siméon 03].

5.1 Historical Development

One of the challenging issues of manipulation planning is to integrate the additional difficulty of planning the grasping and re-grasping operations to the path planning problem. This interdependency between path planning and grasp planning was first touched upon by work done in the 80's for the development of automatic robot programming systems.

In particular, the Handey system [Lozano-Perez 92] integrated both planning levels and was capable to plan simple *pick and place* operations including some re-grasping capabilities. The geometric formulation of manipulation planning [Alami 89, Latombe 91], seen as an instance of motion planning problem extended by the presence of movable objects, provided a unified framework allowing to better tackle the interdependency issues between both planning levels.

Motion planning in presence of movable objects is first addressed as such in [Wilfong 88]. In this work, an exact cell decomposition algorithm is proposed for the particular case of a polygonal robot and of one movable object translating in a polygonal workspace, assuming a finite grasp set of the movable object.

The manipulation graph concept is introduced in [Alami 89] for the case of one robot and several movable objects manipulated with discrete grasps and placements. In this case, the nodes of the manipulation graph correspond to discrete configurations and the edges are constructed by searching for transfer (or transit) paths between nodes sharing the same grasp (or placement) of the movable object(s). Following this general framework, the approach was implemented for a translating polygon [Alami 89] and a 3-d.o.f. planar manipulator [Alami 95]. An exact cell decomposition algorithm is also proposed in [Alami 95] for the specific case of a translating polygonal robot capable to manipulate one movable polygon with an infinite set of grasps.

The manipulation planning framework is extended in [Koga 92, Koga 94] to multi-arm manipulation where several robots cooperate to carry a single movable object amidst obstacles. In this work, the number of legal grasps of the objects is finite and the movable object has to be held at least by one robot at any time during a re-grasp operation. The planner proposed in [Koga 94] (that we also referred in Chapter 4) first plans the motions of the movable object using an adapted version of a randomized potential field planner (RPP), and then finds the sequence of re-grasp operations of the arms to move the object along the computed path. This planner relies on several simplifications, but it can deal with complex and realistic problems. Another heuristic planning approach proposed in [Barraquand 94] is to iteratively deform a coordinated path first generated in the unconstrained composite configuration-space of robots and objects using a variational dynamic programming technique that progressively enforces the manipulation constraints.

Variants of the manipulation planning problem have been investigated. In [Lynch 95], grasping is replaced by pushing and the space of stable pushing directions imposes a set of non-holonomic constraints that introduce some controllability issues to the problem. The heuristic algorithm described in [Chen 91] considers a problem where all the obstacles can be moved by a circular robot in order to find its way to the goal.

Two other contributions extend recent planning techniques to manipulation planning. In [Ahuactzin 98], the Ariadne's Clew algorithm is applied to a redundant robot manipulating a single object in a 3D workspace. The method assumes discrete grasps of the movable object; it is however capable to deal in realistic situations with redundant

manipulators for which each grasp possibly corresponds to an infinite number of robot configurations. Finally, [Nielsen 00] proposes a practical manipulation planner based on the extension of the PRM approach. The planner constructs a manipulation graph between discrete configurations; connections are computed using a *Fuzzy-PRM* planner that builds a roadmap with edges annotated by a probability of collision-freeness. Computing such roadmaps improves the efficiency of the planner for solving the possibly high number of path planning queries (in changing environments) required to compute the connections.

Contribution

The manipulation planning techniques above mostly address the discrete instance of the problem. Only the algorithms in [Alami 95, Ahuactzin 98] consider more difficult instances for which the nodes of the manipulation graph (i.e. the places where the connections between the feasible transit and transfer paths have to be searched) correspond to a collection of submanifolds of the composite configuration-space, as opposed to discrete configurations. Such manifolds arise when considering infinite grasps and continuous placements of the object. This continuous setting is only addressed in [Alami 95] for the specific case of a translating robot in a polygonal world. Manifolds also arise in [Ahuactzin 98] because of the redundancy of the robot although the planner assumes a set of pre-defined discrete grasps.

It this chapter, we describe a general approach for dealing with such continuous settings of the manipulation planning problem. This planning approach considers continuous placements and grasps, and it is also able to handle redundant robots. It relies on a structuring of the search-space allowing us to capture efficiently the connectivity of the submanifolds in a probabilistic roadmap computed for virtual closed-chain mechanisms. The resulting planner is general and practical for solving complicated manipulation planning problems in restrained environments. For example, one can describe the set of stable placements by constraining the movable object to be placed on top of some horizontal faces of the static obstacles. Such placement constraints define a 3-dimensional submanifold of the object's configuration-space (two translations in the horizontal plane and one rotation around the vertical axis). Also, one can consider sets of continuous grasping domains such that the jaws of a parallel gripper have a contact with two given faces of the object. Such grasp constraints also define a 3-dimensional domain (two translations parallel to the grasped faces and one rotation around the axis perpendicular to the faces).

5.2 Problem Statement

Notations: Let us consider a 3D workspace with a robot $\mathcal{A}$ and a movable object $\mathcal{O}$ moving among static obstacles. The robot (supposed an open-chain mechanisms) has degree of mobility M and its configuration is defined by the array of the joint variables q_{rob}. $\mathcal{O}$ is a free-flying object that can only move when it is grasped by the robot. Its configuration

q_{obj} is defined by the six location parameters. Let $\mathcal{C}_{rob}$ and $\mathcal{C}_{obj}$ be the configuration-spaces of the robot and the object, respectively. The composite configuration-space of the system is $\mathcal{C} = \mathcal{C}_{rob} \times \mathcal{C}_{obj}$ and we call $\mathcal{C}_{free}$ the subset in $\mathcal{C}$ of all the collision-free configurations. The domain in $\mathcal{C}$ corresponding to valid placements of $\mathcal{O}$ (i.e. stable placements where the object can rest when ungrasped by the robot) is denoted by $\mathcal{C}_P$. The domain in $\mathcal{C}$ corresponding to valid configurations of $\mathcal{O}$ grasped by the robot $\mathcal{A}$ is denoted by $\mathcal{C}_G$. Both $\mathcal{C}_P$ and $\mathcal{C}_G$ are lower-dimensional manifolds in $\mathcal{C}$.

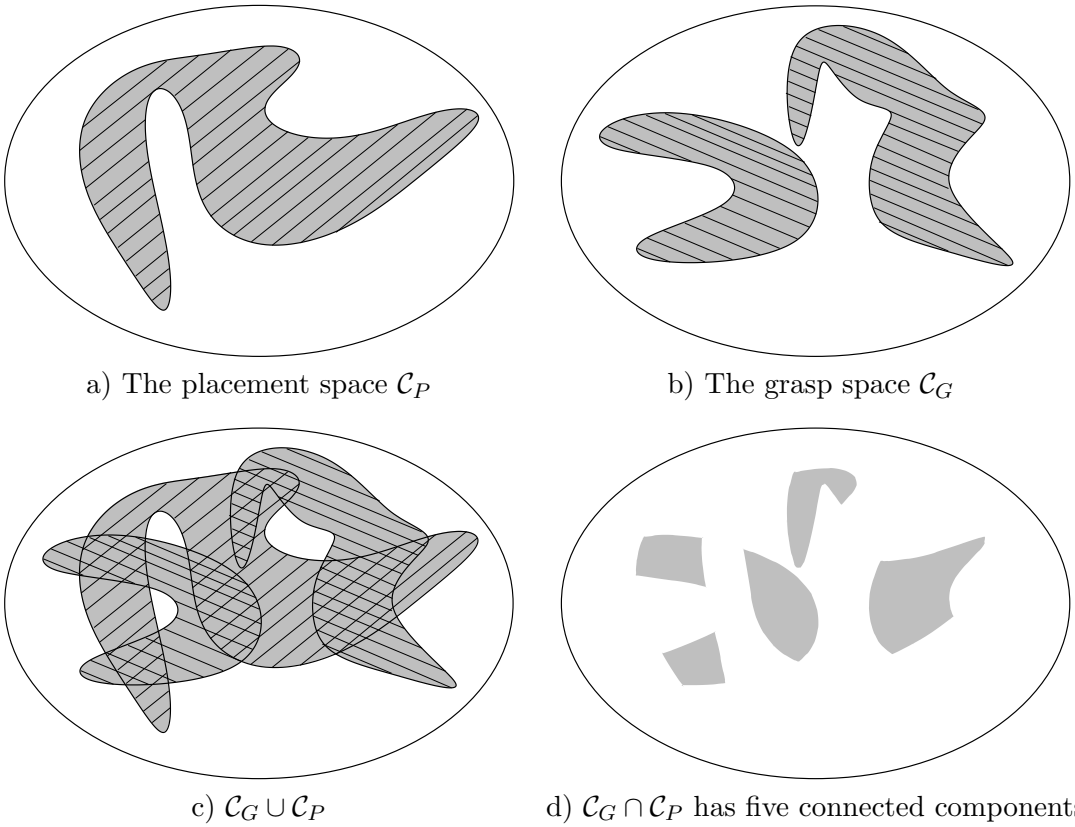
Manipulation Constraints: A solution to a manipulation planning problem corresponds to a constrained path in $\mathcal{C}_{free}$. Such a solution path is an alternate sequence of two types of sub-paths verifying the specific constraints of the manipulation problem, and separated by grasp/ungrasp operations:

- *Transit Paths* where the robot moves alone while the object $\mathcal{O}$ stays stationary at a stable position. The configuration parameters of $\mathcal{O}$ remain constant along a transit path. Such motions allow to place the robot at a configuration where it can grasp the object. They are also involved when changing the grasp of the object. Transit paths lie in $\mathcal{C}_P$. However, a path in $\mathcal{C}_P$ is not generally a transit path since such path has to belong to the submanifold corresponding to a fixed placement of $\mathcal{O}$. Transit paths induce a foliation [1] of $\mathcal{C}_P$ (see Figure 5.2.a).
- *Transfer Paths* where the robot moves while holding $\mathcal{O}$ with the same fixed grasp. Along a transfer path, the configuration of $\mathcal{O}$ changes according to the grasp mapping induced by the forward kinematics of the robot [2]. Thus, there is a relationship between q_{obj} and q_{rob}. Transfer paths lie in $\mathcal{C}_G$and they induce a foliation of this submanifold (see Figure 5.2.b).

Problem: Consider the two sets of constraints defining the stable placements and feasible grasps. A manipulation planning problem is to find a manipulation path (i.e. an alternate sequence of transit and transfer paths) connecting two given configurations q_{init} and q_{goal} in $\mathcal{C}_G \cup \mathcal{C}_P$ (see Figure 5.2.c). Manipulation planning then consists in searching for transit and transfer paths in a collection of submanifolds corresponding to particular grasps or stable placements of the movable object. Note that the intersection $\mathcal{C}_G \cap \mathcal{C}_P$ between the submanifolds defines the places where transit paths and transfer paths should be connected. The manipulation planning problem appears as a constrained path planning problem inside and between the various connected components of $\mathcal{C}_G \cap \mathcal{C}_P$ (see Figure 5.2.d).

[1] A *foliation* of a n-dimensional manifold is an indexed family of disjoint pathwise-connected submanifolds, called *leaves* (see [Candel 00] for details).

[2] Coordinates transformation between the base-frame and the end-frame of the robot, expressed by equation (1.3).



a) The placement space $\mathcal{C}_P$ b) The grasp space $\mathcal{C}_G$

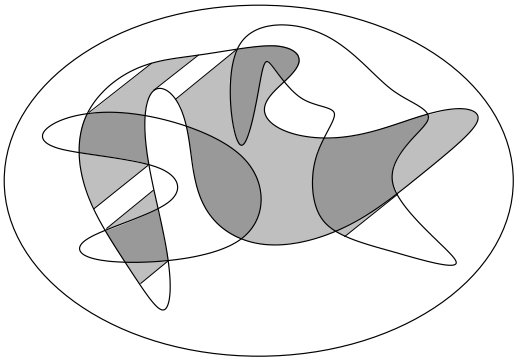
c) $\mathcal{C}_G \cup \mathcal{C}_P$ d) $\mathcal{C}_G \cap \mathcal{C}_P$ has five connected components

Figure 5.2: Moving along transit (resp. transit) paths induces a foliation of the placement (resp. grasp) space. Both foliations intersect themselves in $\mathcal{C}_G \cap \mathcal{C}_P$.

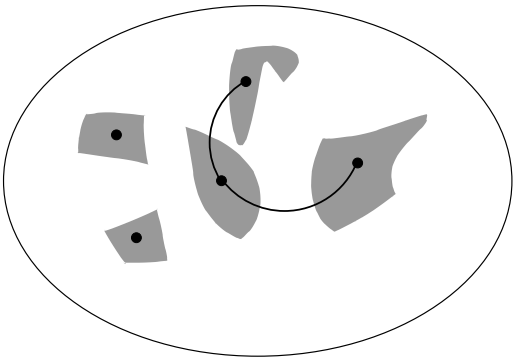
Reduction Property: Two foliation structures are defined in $\mathcal{C}_G \cap \mathcal{C}_P$: the first one is induced by the transit paths; the second one is induced by the transfer paths. As a consequence, any path lying in a connected component of $\mathcal{C}_G \cap \mathcal{C}_P$ can be transformed into a finite sequence of transit and transfer paths (the proof of this property [3] appears in [Alami 95]). Therefore two configurations which are in a same connected component of $\mathcal{C}_G \cap \mathcal{C}_P$ can be connected by a manipulation path.

It is then sufficient to study the connectivity of the various components of $\mathcal{C}_G \cap \mathcal{C}_P$ by transit and transfer paths. Let us consider a transit (or transfer) path whose endpoints belong to two distinct connected components $(\mathcal{C}_G \cap \mathcal{C}_P)_i$ and $(\mathcal{C}_G \cap \mathcal{C}_P)_j$ of $\mathcal{C}_G \cap \mathcal{C}_P$. From the reduction property above one may deduce that any configuration in $(\mathcal{C}_G \cap \mathcal{C}_P)_i$ can be connected to any configuration in $(\mathcal{C}_G \cap \mathcal{C}_P)_j$ by a manipulation path.

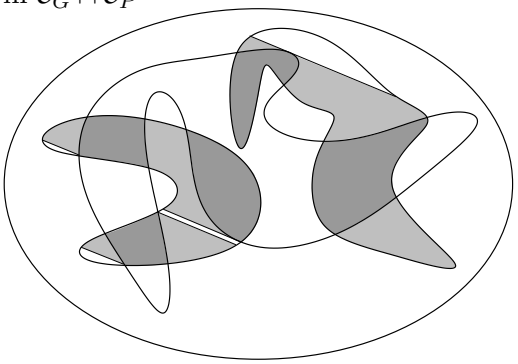
[3] Note that this property holds for a single movable object under the hypothesis that the robot does not touch the static obstacles.



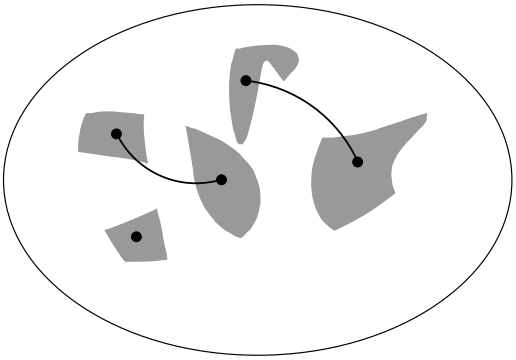
a) Set of configurations reachable by a transit path starting at a configuration in $\mathcal{C}_G \cap \mathcal{C}_P$



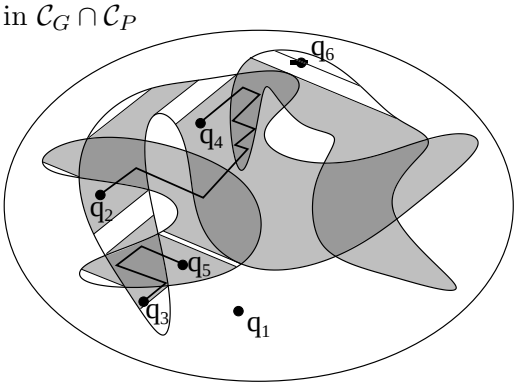
b) Adjacency of $\mathcal{C}_G \cap \mathcal{C}_P$ components via transit paths



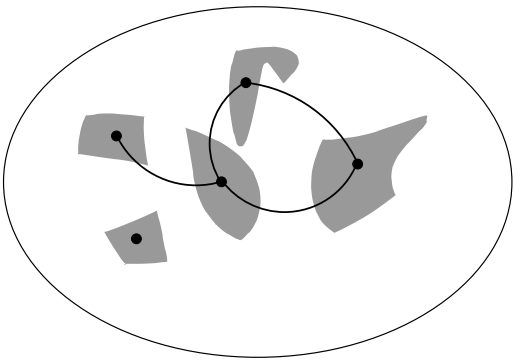
c) Set of configurations reachable by a transfer path starting at a configuration in $\mathcal{C}_G \cap \mathcal{C}_P$



d) Adjacency of $\mathcal{C}_G \cap \mathcal{C}_P$ components via transfer paths



e) Examples of manipulation paths



f) The Manipulation Graph

Figure 5.3: The topology of $\mathcal{C}$ induced by the manipulation problem constraints can be captured by a so-called Manipulation Graph.

Manipulation Graph: It is then possible to build a graph MG whose nodes (to which we will refer later as *mega-nodes*) are the various connected components of $\mathcal{C}_G \cap \mathcal{C}_P$, and whose edges indicate the existence of a transit (or transfer) path whose endpoints belong to different components $(\mathcal{C}_G \cap \mathcal{C}_P)_i$ and $(\mathcal{C}_G \cap \mathcal{C}_P)_j$. Figure 5.3 illustrates the graph structure for the example introduced in Figure 5.2. Examples of manipulation paths are shown in the bottom-left picture: q_1 is not a valid configuration for the manipulation problem (it does not belong to $\mathcal{C}_G \cup \mathcal{C}_P$). Configuration q_6 is in $\mathcal{C}_G$; nevertheless it cannot escape from its leaf in $\mathcal{C}_P$. A manipulation path exists between q_3 and q_5 and between q_2 and q_4. No manipulation path exists between q_5 and q_4.

Let q_{init} and q_{goal} be two configurations in $\mathcal{C}_G \cup \mathcal{C}_P$. There exists a manipulation path between q_{init} and q_{goal} *iff* there exist two mega-nodes $(\mathcal{C}_G \cap \mathcal{C}_P)_{init}$ and $(\mathcal{C}_G \cap \mathcal{C}_P)_{goal}$ in MG, called the *manipulation graph*, such as:

- there exists a transit (or transfer) path from q_{init} to some point in $(\mathcal{C}_G \cap \mathcal{C}_P)_{init}$,
- there exists a transit (or transfer) path from some point in $(\mathcal{C}_G \cap \mathcal{C}_P)_{goal}$ to q_{goal},
- $(\mathcal{C}_G \cap \mathcal{C}_P)_{init}$ and $(\mathcal{C}_G \cap \mathcal{C}_P)_{goal}$ belong to a same connected component of MG.

Combinatorial Issues: How to capture the various connected components of $\mathcal{C}_G \cap \mathcal{C}_P$? How to capture their adjacency by transit and transfer paths? These are the two key issues in manipulation task planning. All the techniques overviewed above fall in this general framework.

5.3 A General Approach to Manipulation Planning

We now describe our approach for solving manipulation problems in the general setting of continuous grasp and placement constraints. The proposed approach relies on the structure of $\mathcal{C}_G \cap \mathcal{C}_P$ discussed in the previous section. The main idea is to exploit the reduction property of Section 5.2 to decompose the construction of the manipulation graph at two levels:

- compute the connected components of $\mathcal{C}_G \cap \mathcal{C}_P$.
- determine the connectivity of $\mathcal{C}_G \cap \mathcal{C}_P$ components using transit and transfer paths.

A Two-Level Sampling-Based Manipulation Roadmap: The manipulation graph MG is computed, as in [Nielsen 00], using a sampling-based motion planning algorithms. But our construction of this roadmap integrates a specific step allowing us to directly capture the connectivity of the subset $\mathcal{C}_G \cap \mathcal{C}_P$ inside the graph. The structure of a manipulation graph computed using this approach is illustrated by Figure 5.3.

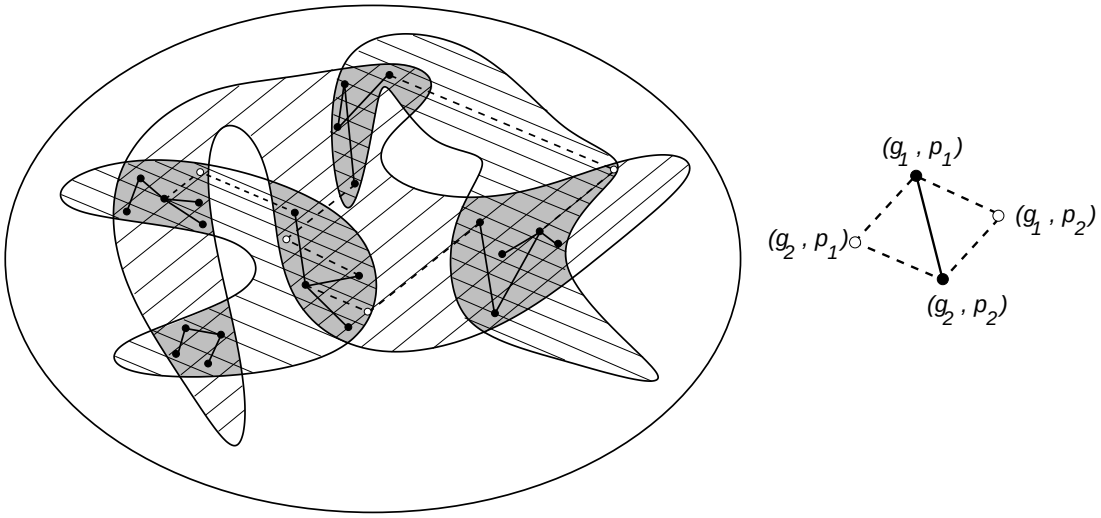


Figure 5.4: A probabilistic roadmap as a manipulation graph: mega-nodes belong to $\mathcal{C}_G \cap \mathcal{C}_P$ while edges model paths belonging to either $\mathcal{C}_G \cap \mathcal{C}_P$, $\mathcal{C}_P$ or $\mathcal{C}_G$. Two types of adjacency are considered: direct $\mathcal{C}_G \cap \mathcal{C}_P$ paths (plain segments) or elementary sequences of *Transit-Transfer* (or *Transfer-Transit*) paths (dashed segments).

The roadmap MG is composed by a small number of mega-nodes (the connected components of $\mathcal{C}_G \cap \mathcal{C}_P$) connected together with transit or transfer paths. Each component $(\mathcal{C}_G \cap \mathcal{C}_P)_i$ is captured into a sub-roadmap computed using a "local" planner that generates feasible $\mathcal{C}_G \cap \mathcal{C}_P$ motions (the black edges in Figure 5.3) between randomly sampled configurations in $\mathcal{C}_G \cap \mathcal{C}_P$ (the black points). These sub-roadmaps are connected via transit and transfer paths (the dotted edges) using some intermediate nodes (in white). The intermediate nodes are defined as follows. Consider two configurations in different connected component of $\mathcal{C}_G \cap \mathcal{C}_P$. These configurations correspond to fixed grasps and placements of the movable object, noted $(g_i, p_i)_{i=1,2}$. Using motions outside $\mathcal{C}_G \cap \mathcal{C}_P$, they can only be connected by following the particular leaves of $\mathcal{C}_P$ and $\mathcal{C}_G$ issued from both configurations. We then define the intermediate nodes as (g_1, p_2) and (g_2, p_1). An edge between (g_1, p_1) and (g_2, p_2) is added if at least one of the intermediate nodes (g_1, p_2) or (g_2, p_1) belongs to $\mathcal{C}_G \cap \mathcal{C}_P$ and is reachable from (g_1, p_1) and (g_2, p_2) by a collision-free transit/transfer path. The connection between two randomly sampled configurations of $\mathcal{C}_G \cap \mathcal{C}_P$ is then possible if one of the three types of adjacency (Figure 5.3) exists:

- `Type1`: a direct path from (g_1, p_1) to (g_2, p_2) lying on $\mathcal{C}_G \cap \mathcal{C}_P$ is collision-free.
- `Type2a`: a transfer path from (g_1, p_1) to (g_1, p_2) followed by a transit path from (g_1, p_2) to (g_2, p_2) are both collision-free.
- `Type2b`: a transit path from (g_1, p_1) to (g_2, p_1) followed by a transfer path from (g_2, p_1) to (g_2, p_2) are both collision-free.

Once the manipulation roadmap is computed, queries are solved by searching for a path in MG. The obtained solution alternates elementary manipulation paths (i.e. transit/transfer paths computed when traversing edges of MG using `Type2` adjacencies) with $\mathcal{C}_G \cap \mathcal{C}_P$ paths (i.e. paths computed inside the nodes of MG using `Type1` adjacencies). Note that the direct $\mathcal{C}_G \cap \mathcal{C}_P$ paths correspond to simultaneous changes of grasp and placement; they are therefore not feasible from the manipulation point of view. However, thanks to the reduction property, any such `Type1` paths can be transformed in a post-processing stage into a finite sequence of `Type2` transit and transfer paths.

Capturing the Topology of $\mathcal{C}_G \cap \mathcal{C}_P$ via a Virtual Closed-Chain Mechanism: The main critical issue of the approach is to capture into a probabilistic roadmap the topology of $\mathcal{C}_G \cap \mathcal{C}_P$ which is a subset of the global configuration-space $\mathcal{C}$ with a lower dimension. The idea here is to explore $\mathcal{C}_G \cap \mathcal{C}_P$ as such. For this, we consider that $\mathcal{C}_G \cap \mathcal{C}_P$ is the configuration-space of a single system consisting of the robot together with the movable object placed at a stable position. Maintaining the stable placement while the movable object is grasped by the robot induces a virtual closed kinematics chain (see Figure 5.5).

We now explain how the virtual closed-chain mechanism is defined. A grasp of the movable object can be represented by a homogeneous transformation matrix ${}^{\mathcal{O}}T_{\mathcal{A}_e}$ defining the location of the end-effector $\mathcal{A}_e$ relative to the coordinate system of the object $F_{\mathcal{O}}$. For a fixed grasp, ${}^{\mathcal{O}}T_{\mathcal{A}_e}$ is a constant matrix. However this transformation can involve variables parameterizing a continuous set of grasps. We call q_{grasp} this set of parameters and we denote as ${}^{\mathcal{O}}T_{\mathcal{A}_e}(q_{grasp})$ the transformation depending on these variables. Given a transform matrix defining the location of the robot base in the world ${}^{\mathcal{W}}T_{\mathcal{A}_0}$ and the forward kinematics function for the robot ${}^{\mathcal{A}_0}T_{\mathcal{A}_e}(q_{rob})$, we can define a mapping:

$$G(q_{rob}, q_{grasp}) = {}^{\mathcal{W}}T_{\mathcal{A}_0} \cdot {}^{\mathcal{A}_0}T_{\mathcal{A}_e}(q_{rob}) \cdot {}^{\mathcal{O}}T^{-1}_{\mathcal{A}_e}(q_{grasp})$$

which determines the subset of locations of the object while grasped by the robot. Note that, in the previous sentences, we have intrinsically defined a parameterization for the

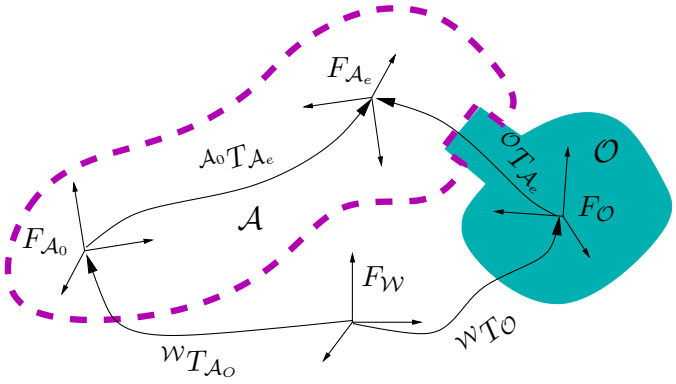


Figure 5.5: Schema of the virtual loop whose configuration-space matches with $\mathcal{C}_G \cap \mathcal{C}_P$.

submanifold $\mathcal{C}_G$. $\mathcal{C}_G$ corresponds to the subset of free configurations $q = \{q_{rob}, q_{obj}\}$ for which the configuration q_{obj} of $\mathcal{O}$ changes according to the grasp mapping induced by the forward kinematics of the robot and by the grasp of the object: $q_{obj} = G(q_{rob}, q_{grasp})$. $\mathcal{C}_G$ is therefore parameterized by the array of variables $\{q_{rob}, q_{grasp}\}$.

In a similar way, the subset of stable placements of $\mathcal{O}$ can be defined by a mapping:

$$P(q_{place}) = {}^{\mathcal{W}}T_{\mathcal{O}}(q_{place})$$

where q_{place} denotes the array of variable parameters defining of a continuous set of stable placements. The submanifold $\mathcal{C}_P$ corresponds to configurations where q_{obj} changes according to this mapping: $q_{obj} = P(q_{place})$.

A same configuration q_{obj} of the the movable object can be obtained from the two different mappings, $G(q_{rob}, q_{grasp})$ and $P(q_{place})$, *iff* $\mathcal{O}$ is simultaneously grasped by the robot and located at a stable placement. This condition characterizes configurations in the subset $\mathcal{C}_G \cap \mathcal{C}_P$. Then, configurations in $\mathcal{C}_G \cap \mathcal{C}_P$ can be defined by an array of variables $\{q_{rob}, q_{grasp}, q_{place}\}$ satisfying the constraint: $G(q_{rob}, q_{grasp}) = P(q_{place})$. Such a constraint establishes a relationship: $f(q_{rob}, q_{grasp}, q_{place}) = 0$, which has the same form than loop closure equations (1.9). Thus, we can conceive a virtual kinematic loop $\mathcal{L}$ whose joint variables are:

$$q_{\mathcal{L}} = \{q_{rob}, q_{grasp}, q_{place}\}$$

$\mathcal{L}$ involves the robot, the movable object and virtual joints between $\mathcal{O}$ and $\mathcal{A}_e$, and between $\mathcal{A}_0$ and $\mathcal{O}$, with joint variables q_{grasp} and q_{place} respectively. Hence, capturing the topology of $\mathcal{C}_G \cap \mathcal{C}_P$ requires to explore the variety of the configurations $q_{\mathcal{L}}$ satisfying closure constraints. A roadmap whose components encode the connectivity of collision-free subsets on this variety can be constructed by the approach explained in Chapter 3. In the next section we give details about this particular application.

Connections with Transit and Transfer Paths: Computing such connections requires to solve multiple point-to-point path planning problems, as for the case of manipulation planning with discrete sets of grasps and placements (e.g. [Nielsen 00]). Here, the issue is to provide efficient solutions for searching such collision-free transit (or transfer) paths in the various leaves of $\mathcal{C}_P$ (or $\mathcal{C}_G$).

Our approach exploits the fact that each planning problem has to be performed in a *partially modified environment*. The structure of $\mathcal{C}_{free}$ is in general slightly affected by variations of q_{obj}. A "static" roadmap is used as a global basis for solving these problems. It is combined with an incremental search algorithm that performs at a local level for solving problems occasioned by these slight changes in $\mathcal{C}_{free}$. This planning technique is also further explained in the section below.

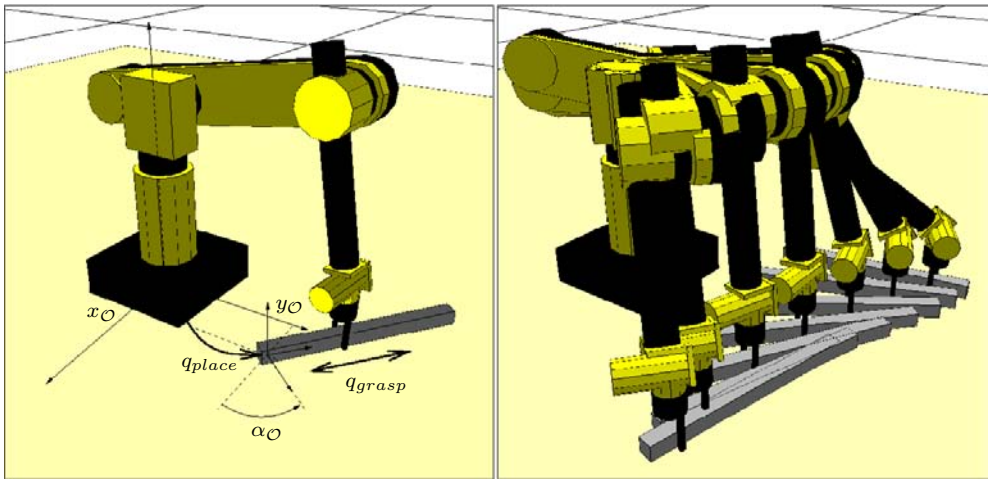


Figure 5.6: Virtual closed-chain system and a feasible motion in $\mathcal{C}_G \cap \mathcal{C}_P$: the bar moves on the floor while sliding into the gripper's jaws.

5.4 The Planning Techniques

We now detail the planning techniques developed to implement the approach. The two basic primitives required for computing the `Type1` and `Type2` are described first. Then, we explain how both primitives are combined by the algorithm used to build the manipulation roadmap.

Closed-Chain Planner for Type1 Motions: As explained above, our approach requires to apply motion planning techniques for a single-loop mechanism in order to capture the topology of $\mathcal{C}_G \cap \mathcal{C}_P$. The extended PRM-based planner explained in Chapter 3 is a very suitable tool for this aim. Indeed, the practical efficacy of our manipulation planning approach results from the good performance of this algorithm. We apply the extended version of the Visibility-PRM algorithm for capturing $\mathcal{C}_G \cap \mathcal{C}_P$. The interest of this particular algorithm is first to control the quality of the roadmap in term of coverage; and second, to capture the connectivity of possibly complex spaces into a small data structure. We believe that the small size of the visibility-roadmaps, combined with the proposed structuring of $\mathcal{C}_G \cap \mathcal{C}_P$ contributes to the overall efficiency of our approach by limiting the number of costly path-planning queries to be performed during the second stage, when searching the connections with collision-free transfer or transit paths.

The virtual loop $\mathcal{L}$ has kinematic features that incite to make the next selection of parameters for the RLG algorithm (Algorithm 3.1):

$$q^a_{\mathcal{L}} = \{q^a_{rob}, q_{grasp}, q_{place}\}$$
$$q^p_{\mathcal{L}} = q^p_{rob}$$

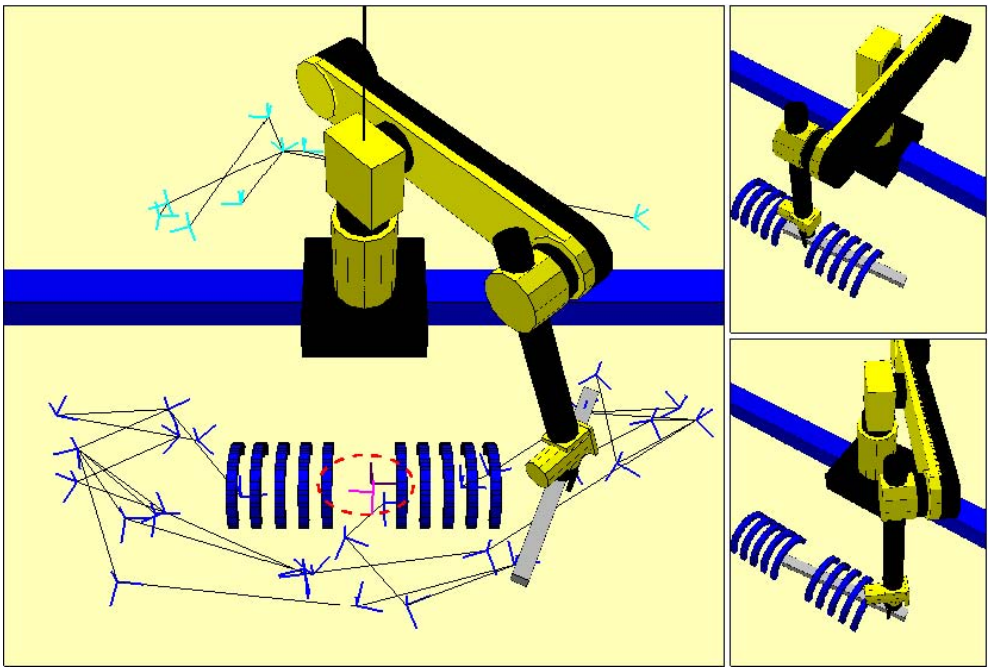


Figure 5.7: A visibility-roadmap computed on $\mathcal{C}_G \cap \mathcal{C}_P$ (left) and two configurations inside two different connected components of $\mathcal{C}_G \cap \mathcal{C}_P$ (right).

In the example illustrated by Figure 5.1, the robotic manipulator is non-redundant. Thus, there are no parameters $q^a_{\mathcal{L}}$. On the contrary, for a mobile manipulator, the configuration parameters of the mobile base should be in $q^a_{\mathcal{L}}$.

Figure 5.6 shows the virtual closed-chain system for this example. The bar moves in contact with the floor while sliding within the gripper. The sliding motion of the gripper results from an additional degree of freedom q_{grasp} introduced in the system to characterize the infinite set of grasps. In this example q_{grasp} is chosen to allow a translation of the parallel jaw gripper along the bar. Similarly, the set of stable placements corresponds to the planar motions parameterized by three variables into q_{place} (two horizontal translations and a vertical rotation), that maintain the contact of the bar with the floor. The motion shown in the right image of Figure 5.6 is a feasible motion in $\mathcal{C}_G \cap \mathcal{C}_P$. It is not admissible for a direct execution. However, thanks to the reduction property it can be transformed into a finite sequence of feasible transit and transfer paths.

Figure 5.7 shows the visibility-roadmap encoding $\mathcal{C}_G \cap \mathcal{C}_P$ for the example of Figure 5.1. The computed roadmap has four connected components: two main components separated by the long static obstacle, and two other small components that correspond to placements of the movable object inside the cage obstacle while it is grasped by the arm through the open passage in the middle of the cage. These two small components (inside the dashed circle of the left image) correspond to locations of the bar having similar position but

different orientations, about 180° apart. The associated configuration of the closed-chain system is shown in the top-right image. The bottom-right image corresponds to a node of the main component with the bar placed at the same position, but using a different grasp. Connecting this node to the small component is not possible (into $\mathcal{C}_G \cap \mathcal{C}_P$) because of the cage obstacle that limits the continuous change of grasp. Such re-grasping requires the computation of collision-free paths outside $\mathcal{C}_G \cap \mathcal{C}_P$ as explained below.

Connection Planner for Type2 Motions: Computing `Type2` connections requires a basic routine to find elementary collision-free transit and transfer paths. Each of the planning problems corresponds to a particular grasp or placement of the movable object. Then, the queries have to be performed in a partially modified environment. The motivation of the two-stage method used by the connection planner is simply to amortize the cost of dealing with such partial changes by re-using at each query some of the paths pre-computed during the first stage regardless of the movable object.

First, we compute a roadmap for the robot and the static obstacles, without considering the presence of the movable object. Then, before to solve a given (transit or transfer) path query, the roadmap is updated by checking whether each edge is collision-free with respect to the current position of the movable object. Colliding edges are labeled as blocked in the roadmap.

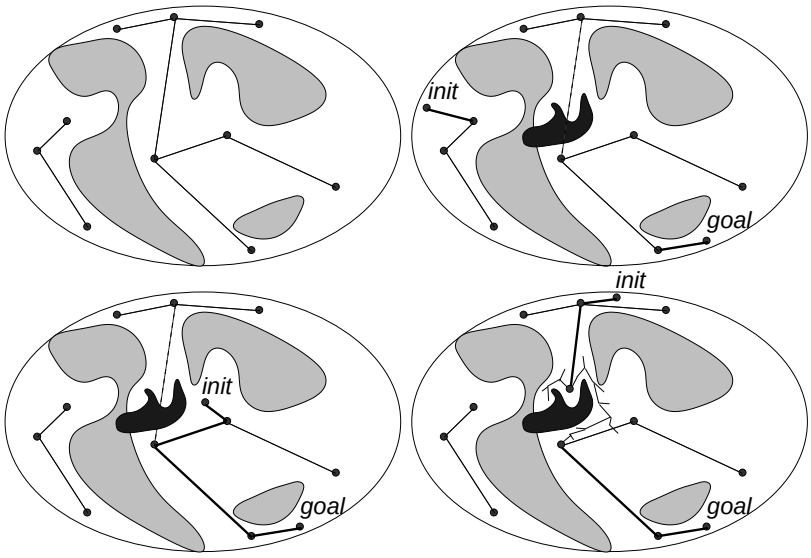


Figure 5.8: A static roadmap is computed in the configuration space of the robot (top, left). During queries, it is labeled according to collisions including $\mathcal{O}$. If the query fails, there is no solution (top, right). Otherwise, either there exists a solution path in the roadmap avoiding labeled edges (bottom, left) or not (bottom, right). In the later case the colliding part of the path is locally updated using an RRT-like technique.

The search for a given path is then performed within the labeled roadmap. As illustrated by Figure 5.8, three cases possibly occur. When the search fails, this means that no path exists even in the absence of the movable object; the problem has no solution. Similarly, when the computed path does not contain any blocked edge (dashed edges in Figure 5.8) then a solution is found. Now let us consider the intermediate situation where the solution path necessarily contains blocked edges. In such case, the algorithm tries to solve the problem locally using a RRT-based planner to connect the endpoints of the blocked edges. Such a call to the RRT planner can be seen a dynamic updating of the roadmap. The main interest of using the RRT approach is that it performs well locally. This means that the approach quickly finds connections between nearby nodes.

Manipulation Planning Algorithm: The algorithm incrementally constructs the manipulation roadmap MG by interleaving the two steps of the approach: computing $\mathcal{C}_G \cap \mathcal{C}_P$ connected components (`Type1` adjacency) and linking them (`Type2a-b` adjacencies). Following the principle of Visibility-PRM, the algorithm stops when it is not able to expand the graph after a given number of tries. This number of failures is related to an estimated coverage of the search-space [Siméon 00] (in our case, the $\mathcal{C}_G \cap \mathcal{C}_P$ subset). The function EXPAND_MG (in Algorithm 5.1) performs one expansion step of MG. Candidate nodes are first sampled in $\mathcal{C}_G \cap \mathcal{C}_P$ and the different types of connections to the graph are then tested.

The algorithm possibly considers several classes of continuous grasps and/or placements. For each one, the corresponding transform matrices (${}^{\mathcal{O}}T_{\mathcal{A}_e}$ or ${}^{\mathcal{W}}T_{\mathcal{A}_0}$) and sets of parameters (q_{grasp} or q_{place}) are defined as input data of the manipulation problem MP. Each couple *grasp-placement* induces a particular closed-chain system $\mathcal{L}_{g-p}$, and consequently, a class of node in $\mathcal{C}_G \cap \mathcal{C}_P$. A candidate node is generated as follows by the function RANDOMFREECONFIGURATION for one of these virtual loops, chosen at random. Then, ADJACENCYCHOICE selects a type of adjacency (`Type1` or `Type2`) by performing

Algorithm 5.1: EXPAND_MG

```
input  : the problem data MP, the manipulation graph MG
output : the manipulation graph MG
begin
    q_new ← RandomFreeConfiguration(MP);
    adj_type ← AdjacencyChoice(MP, MG);
    n_linked_comp ← TestConnections(q_new, adj_type, MG);
    if n_linked_comp ≠ 1 then
        AddNewNodeAndEdges(q_new, adj_type, MG);
        UpdateGraph(MG);
    else return Failure;
end
```

a biased random choice. The bias for this choice depends on the evolution of the size of MG. The aim is to minimize the computing time for constructing the manipulation graph (see [Siméon 03] for further details).

The function TestConnections checks the connection between the new (potential) node and each connected component of MG using the selected type of adjacency. When the expansion step is performed using `Type1` motions, candidate nodes for checking connection must belong to the same class grasp-placement, i.e. they have been generated for the same loop $\mathcal{L}_{g-p}$. Following the visibility principle, the candidate node is added to the graph only if the random sample q_{new} was linked to none or to more than one connected component. In the second case, the linked components are merged.

Once MG built, manipulation planning queries can be performed using the three following steps. First, the start and goal configurations are connected to MG using the planner for `Type2` motions, and the manipulation graph is searched for a path between both configurations. Then, `Type1` portions of the solution path are decomposed into sequences of transit and transfer paths by a simple dichotomic procedure detailed in [Sahbani 02, Siméon 03]. Finally, the solution is smoothed by a procedure that eliminates unnecessary motions (see also in [Sahbani 02]).

5.5 Results

In this section, we will only comment results of tests performed on the example with the manipulator, the bar and the cage, which has illustrated this chapter. More results of the application of our manipulation planning approach are presented in [Sahbani 02, Siméon 03].

The manipulation problem shown in Figure 5.1 is particularly complicated. Several consecutive re-graspings motions through the middle of the cage are necessary to move the bar to a position where it can be grasped by its extremity. The planner automatically computes the required configurations from only one continuous placement domain (the floor) and one grasping zone all along the bar. The path to get the bar out of the cage is found in the subset $\mathcal{C}_G \cap \mathcal{C}_P$, and then transformed during the post-processing step in a sequence of transit and transfer paths. Using other state-of-the-art planners (e.g. [Ahuactzin 98, Nielsen 00]), the user should determine and define "by hand" all the punctual placements and grasps of the movable object required to solve the manipulation task. Such discrete placements and grasps are rather close and precise in this example.

We introduced a tuning parameter in the algorithm Expand_MG in order to analyze the importance of exploring continuous $\mathcal{C}_G \cap \mathcal{C}_P$ regions within our approach. This parameter, that we call $\alpha \in [0, 1)$, modifies the behavior of the function AdjacencyChoice. With α set to zero, the roadmap builder only considers connections of MG's nodes with transit/transfer paths. In this case the algorithm behaves as the discrete approaches in which punctual placemets and grasps of the movable object are randomly sampled in a

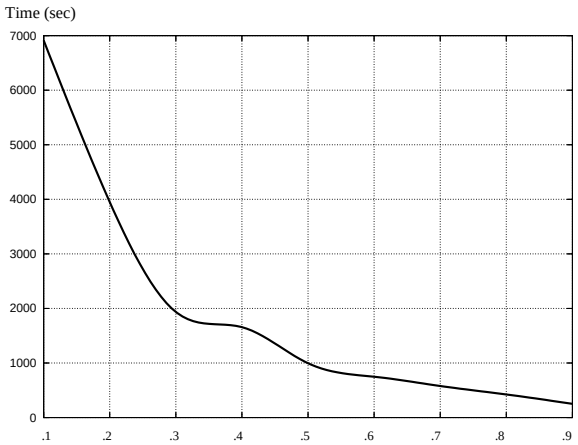


Figure 5.9: Performance of the algorithm depending on the percentage of $\mathcal{C}_G \cap \mathcal{C}_P$ exploration (`Type1` paths) wrt. to transit/transfer (`Type2` paths) used to build the manipulation roadmap for the example of Figure 5.1. The abscissa corresponds to the parameter α.

given domain. When α tends toward 1, the algorithm rarely selects such `Type2` connections before a sufficient coverage of $\mathcal{C}_G \cap \mathcal{C}_P$ has been reached.

The curve displayed in Figure 5.9 plots the time [4] spent by the algorithm to build the manipulation roadmap allowing to solve our illustrative problem. As one can note onto the curve, the computation time significantly decreases for runs performed with higher values of α. This performance improvement can be explained by the fact that many searches of collision-free motions along the leaves of $\mathcal{C}_P$ and $\mathcal{C}_G$ are avoided thanks to the direct exploration of the subset $\mathcal{C}_G \cap \mathcal{C}_P$. Note however that when α tends towards 1, the probability of selecting `Type2` adjacencies remains very low until a sufficient coverage of $\mathcal{C}_G \cap \mathcal{C}_P$ with `Type1` adjacencies has been reached. Since `Type2` adjacencies are required to link the $\mathcal{C}_G \cap \mathcal{C}_P$ connected components, the performance decreases again when $\alpha \rightarrow 1$. The reason is that the algorithm spares time to reach such good coverage inside $\mathcal{C}_G \cap \mathcal{C}_P$ instead of trying connections outside this subset. In all the experiments performed with the planner, this degradation of performance was observed to become significant for values of α close to 1 (see [Sahbani 02]). The experimental study conducted on the difficult manipulation problem of Figure 5.1 tends to show that when the problem is rather constrained, it is qualitatively advantageous to spend time on the connectivity of the $\mathcal{C}_G \cap \mathcal{C}_P$ subset before checking connections with feasible manipulation paths. As shown by the curve, the gain can be very important in such constrained situations. It is however observed to be less significant on simpler problems. As often with the randomized methods, the choice of the best value for this parameter remains an issue that would need to be further investigated. In our experiments with the planner, runs are generally performed with a value of α set to .9 .

[4] Each time value was averaged over ten runs performed using different seeds to initialize the random generator.

For $\alpha = .9$, this difficult manipulation problem was solved in less than 2 minutes. Less than 25% of the computing time was spent in exploring $\mathcal{C}_G \cap \mathcal{C}_P$, while the few connections by transit/transfer paths still remain the most expensive process. The final path contains 20 elementary paths with 8 re-graspings of the movable object.

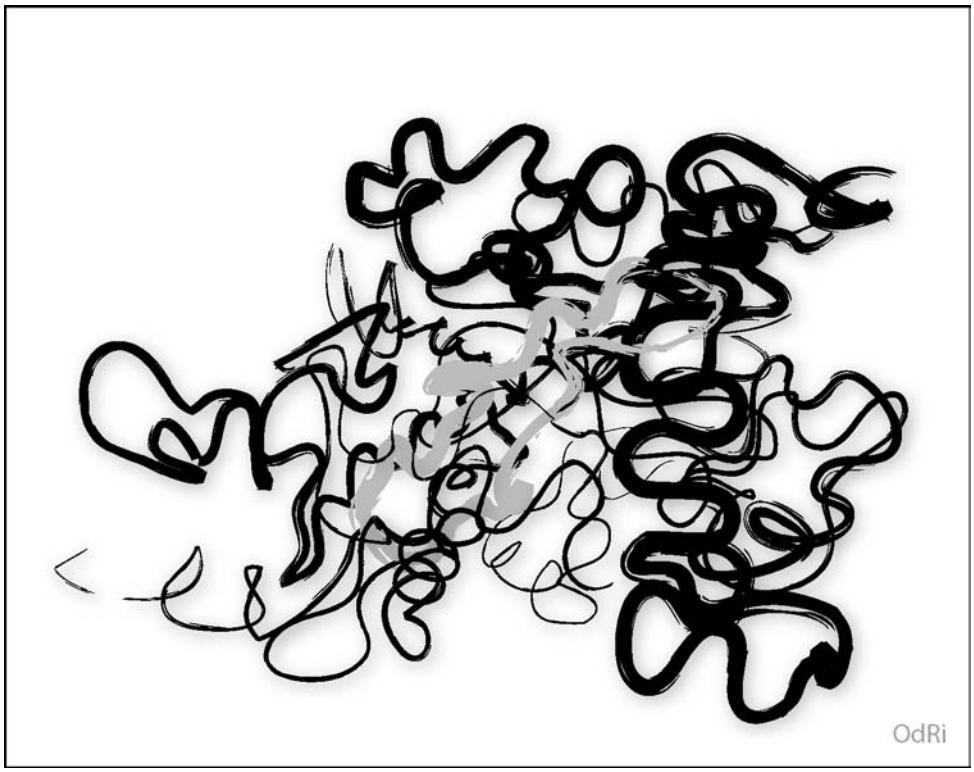
5.6 Discussion and Prospects

We have presented a new approach to manipulation planning. The power of the approach lies in the fact that it can deal with continuous settings of the manipulation problem. It relies on a structuring of the search-space allowing to directly capture into a probabilistic roadmap the connectivity of the submanifolds that correspond to the places where transit paths and transfer path can be connected. This structuring allows us to design a manipulation planner that automatically generates inside continuous domains, the particular grasps and placements that make the problem solvable. Experimental results show the effectiveness of the approach for solving difficult manipulation problems.

There remain several possible improvements, in particular to improve the performance of the connection planner, which is the most costly operation. This addresses the issue of improving the efficiency of PRM planners when facing dynamic changes of the environment. Also, although the approach as the potential to handle general models of the grasp and placements spaces, the planner is currently implemented for the particular case of planning *pick and place* operations for polyhedral objects. One could however imagine applications requiring to consider other models. It would therefore be interesting to further investigate how the manipulation planner can be extended to handle richer models of the grasp and placement spaces. Finally, our manipulation planner is currently restricted to a single movable object manipulated by a single robot. Considering the case of multiple movable objects and robots first requires studying the conditions under which the reduction property can be extended to such situations. Another possibility to solve such more general instances of the manipulation planning problem under investigation is to combine geometric approaches with a symbolic task planning level [Gravot 02, Gravot 03]. The approach explained in this chapter (as the techniques in Chapters 3 and 4) is suitable for direct integration into these new hybrid algorithms.

Chapter 6

Geometric Exploration of Molecular Motions



Prime techniques in structural investigations of molecules require the exploration of their conformational space $\mathcal{C}$ [1]. *Conformational search* methods [Leach 96] explore $\mathcal{C}$ aiming to identify the stable structures of molecules, which determine their properties and functions. *Molecular simulations* [Frenkel 96] explore $\mathcal{C}$ while performing conformational changes on a molecule when the environmental conditions are modified. The analysis of such variations of the molecular structure is essential for the understanding of many biological processes.

Since the goal of the conformational search is to find minimum energy structures, the exploration is much more efficient when it is executed in a subset of $\mathcal{C}$ excluding energetically unacceptable conformations. Conformational changes explored in simulations can take place only if there is not a high energetic barrier to overcome. Therefore, families of approaches treating these problems will greatly benefit from efficient techniques able to provide samples and paths in that $\mathcal{C}$ avoid some unfeasible conformations.

The conformational analysis of a whole macromolecule is a very difficult problem. From a methodological point of view, two stages are usually necessary. The first stage corresponds to the identification of rigid regular segments (i.e. secondary structural elements) capable of participating in the molecular framework. Comparative-modeling methods [Contreras-Moreira 02] have clearly demonstrated that accurate models can be predicted for these regular portions of the structure. The second stage is devoted to the remaining segments, so called *loops*, assumed to be much more flexible. Because of their irregular structure, available techniques to predict low energy conformations of long loops are limited and much less efficient. Figure 6.1 shows two representations of amylosucrase from *Neisseria polysaccharea*, the protein on which we have worked (see Section 6.5). The left image is a space-filling model with atoms represented as spheres. The right image represents the protein structure: cylinders and arrows correspond to secondary structure elements, *α-helices* and *β-sheets* respectively; the rest are the loops.

When the global molecular architecture is assumed to be known and only portions (loops) are studied separately, the integrity of molecular chains must be maintained. The first and last atoms of the treated segment of a molecular chain must remain bonded with their neighbor atoms. Breaking these bonds requires a very high amount of energy. Hence, a strong constraint is imposed for the conformational exploration. This same constraint is present in the analysis of cyclic molecules. It is often referred to as the *loop closure constraint*, and basically, its formulation is the same as in Robot Kinematics (see Section 1.2). Three main kinds of method can be applied in Computational Biology and Chemistry for computing conformations satisfying loop closure: analytical (e.g. [Gō 70, Manocha 95, Wedemeyer 99]), optimization-based (e.g. [Shenkin 87, Zheng 93, Canutescu 03]) and database methods (e.g. [Oliva 97, van Vlijmen 97]). The difficulty of this problem increases with the length of the treated molecular chain, and most of the available techniques are limited, or at least strongly penalized, by this.

[1] A conformation for a molecule is the equivalent to a configuration for a robot. We designate both, the configuration-space and the conformational space, by $\mathcal{C}$.

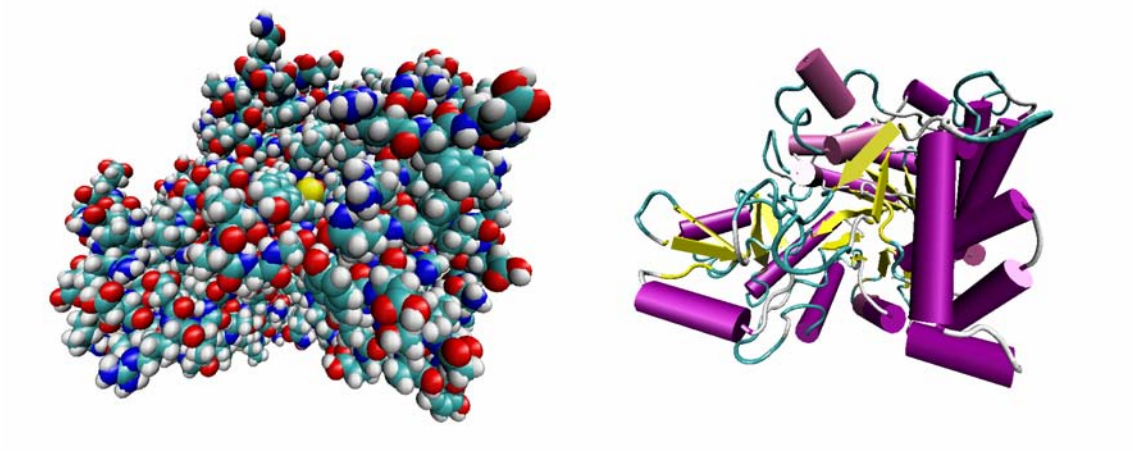


Figure 6.1: Amylosucrase from *Neisseria polysaccharea*: VdW model (left) and structure representation (right).

In addition to breaking bonds, another large amount of energy is required to get two non-bonded atoms significantly closer than the sum of their Van der Waals (VdW) radii. A violation of this condition is called *steric clash*. Feasible conformations of a molecular segment cannot contain either internal clashes, which we call *self-clashes*, or clashes with atoms of the rest of the molecule. A possible kind of filter for such unacceptable conformations consists of evaluating the repulsive term of the VdW energy and discarding them if they exceed a given cutoff value [Bruccoleri 87]. However, this energetic constraint can also be treated by geometric procedures. The use of "clash grids", computed from the distances between atoms, to achieve this filtering was proposed in [Moult 86]. An interesting alternative is the use of collision detection algorithms on a 3D model of the molecule [Lotan 02]. Obviously, the higher the number of atoms, the more critical the efficiency of the technique.

Note that the two above mentioned constraints, loop closure and collision avoidance, are the constraints affecting robot motions the we address from the beginning of this document. In this chapter we propose the application of our motion planning algorithms to molecular models. Although the method could be applied to any molecular segment or cyclic molecule, we are mainly interested in the application to long protein loops. This particular interest is discussed in Section 6.1. Then, in Section 6.2 we explain how molecules can be modeled like robots. We give some details about particularities of the application of our geometric algorithms to conformational sampling (Section 6.3), and to explore conformational changes of protein loops (Section 6.4). First experimental results (Section 6.5) demonstrate the efficacy of the approach handling a 17-residue loop with would play a very important role in the activity of an enzyme under current study by biologists. Finally, our objectives for future work are commented in Section 6.6.

6.1 Interest in Protein Loops

Loops play key roles in the function of proteins. They are often involved in active and binding sites. Therefore, when predicting a protein structure, an accurate loop modeling is necessary for determining its functional specificity.

Modeling loops in proteins is one of the main open problems in Structural Biology. Comparative-modeling methods (see [Contreras-Moreira 02] for a survey) often fail in the prediction of protein loop structures when the percentage of sequence identities between known and predicted protein family members is low. Indeed, it is well established that there is no reliable approach for modeling long loops (more than five residues) available at this time [Tramontano 01].

The alternatives to comparative-modeling are *de novo* (or *ab initio*) methods [Baker 01]. Such methods carry out a search of low energy conformations for a given amino-acid sequence. Many different approaches have been proposed for modeling protein loops. One of the most developed techniques is described in [Fiser 00]. This reference paper also provides a concise survey of loop modeling methods. The accuracy of *de novo* methods mainly depends on the energy function they use. Therefore, improvements in the results provided by these approaches require the design of fine energy models. Progress in the conformational exploration strategies may also be necessary in order to increase the efficiency of these techniques which are today computationally expensive.

Even more important than predicting the stable conformation of a loop for given environmental conditions, is determining the nature of feasible conformational changes. In many enzymes for example, surface loops undergo conformational changes to achieve catalysis [Osborne 01]. Furthermore, loop motions are, in general, important in any protein interaction. Introducing loop flexibility into docking approaches will provide the more accurate structural analysis required for predicting protein interactions [Janin 03].

Aim of our Approach

The techniques proposed in this paper aim to be new tools for the structural analysis of long polypeptide segments, and in particular of protein loops. The efficiency of geometric algorithms developed in the field of Robotics can relieve conformational exploration approaches of a part of the heavy energetic treatment.

First, a conformational sampling technique that provides random conformations achieving loop closure and clash avoidance constraints is proposed in Section 6.3. Families of approaches requiring conformational sampling, such as Monte Carlo algorithms [Metropolis 53] or Stochastic Roadmap techniques [Apaydin 02], would directly benefit from such filtered conformations. Our approach builds loop conformations depending not only on its structure but also on the conformation of the rest of the protein. A sufficient number of random samples uniformly distributed on the conformational space will provide

very useful information about the allowed conformations of this loop in its environment. For instance, this information could be represented and used in the form of Ramachandran plots [Ramachandran 68]. Approaches using this kind of statistical distribution (e.g. MODELLER [Fiser 00]) could improve their performance.

The geometric analysis can be pushed further. In Section 6.4, we propose an algorithm to explore the connectivity of the sub-space of geometrically feasible conformations. Given a starting conformation, the possible deformations maintaining loop closure and clash avoidance constraints are explored and encoded in a data structure. Information in this data structure should be useful for many existing conformational exploration approaches. Furthermore, new methods should be designed inspired by the basic principles of this geometric approach. In a similar direction, a conformational search method, that shares ideas with sampling-based motion planning algorithms for closed kinematic chains [LaValle 99], has been proposed in [LaValle 00] for small molecules (ligands) under geometric constraints. In this case, constraints are imposed on the relative position of atoms in order to match a determined pharmacophore.

6.2 Molecular Modeling

6.2.1 Kinematics Inspired Model

A molecule is a set of atoms $\mathcal{A}_i$ partially connected by bonds. Commonly, a point designates the position of an atom and a straight-line segment the bond between two atoms. A sequence of bonded atoms is called a *molecular chain*. Three parameters, usually called *internal coordinates*, define the relative position of consecutive atoms in a molecular chain:

- ***Bond length***: the distance between two consecutive (bonded) atoms $\mathcal{A}_{i-1}$ and $\mathcal{A}_i$.
- ***Bond angle***: the angle between two consecutive bonds $\mathcal{A}_{i-2}$-$\mathcal{A}_{i-1}$ and $\mathcal{A}_{i-1}$-$\mathcal{A}_i$.
- ***Dihedral angle***: for four consecutive atoms $\mathcal{A}_{i-2}$, $\mathcal{A}_{i-1}$, $\mathcal{A}_i$ and $\mathcal{A}_{i+1}$, the dihedral angle around the bond $\mathcal{A}_{i-1}$-$\mathcal{A}_i$ is the angle formed by planes $\mathcal{A}_{i-2}$-$\mathcal{A}_{i-1}$-$\mathcal{A}_i$ and $\mathcal{A}_{i-1}$-$\mathcal{A}_i$-$\mathcal{A}_{i+1}$.

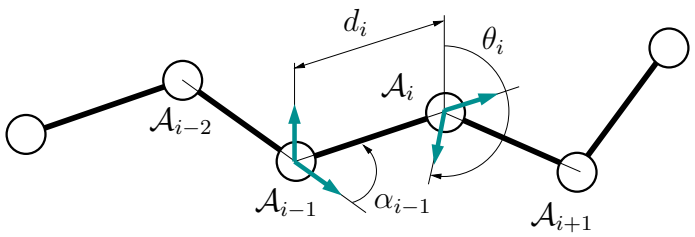


Figure 6.2: Molecular chain model. The mDH parameters are directly obtained from the internal coordinates.

The widely adopted *rigid geometry assumption* (see [Scott 66] as one of the first references) considers that only dihedral angles are variable parameters. Under this assumption, a molecule can be seen as an articulated mechanism with revolute joints whose axes correspond to bonds. The model of a molecular chain can be built from the internal coordinates using kinematic conventions. As we said in Chapter 1, we follow the modified-Denavit-Hartenberg (mDH) convention described in [Craig 89]. A Cartesian coordinate system F_i is attached to each atom $\mathcal{A}_i$. Then the relative location of consecutive frames F_{i-1} and F_i can be defined by the homogeneous transformation matrix (1.2), where d_i is the bond length, α_{i-1} is the supplement of the bond angle and θ_i is the dihedral angle as defined above, being $a_{i-1} = 0$ (see Figure 6.2).

A molecular chain between atoms $\mathcal{A}_0$ and $\mathcal{A}_n$ is then modeled as a kinematic chain, ${}^1\mathcal{K}_n$, in which joint variables correspond to dihedral angles. The conformation of the chain is determined by the array q of the θ_i.

Such a kinematic modeling to molecular chains has been proposed and applied by many scientists (e.g. [Manocha 95, Finn 98, LaValle 00]). Nevertheless, it is not the most efficient method for updating conformations. Often, some portions of molecular models are treated as rigid solids, peptide units for instance (see below). Then, attaching a coordinate system to each atom in the molecule yields superfluous operations. A recent work [Zhang 02] proposes a method for associating frames to rigid units, called *atomgroups* by the authors. Then, the relative location of atoms in an atomgroup only requires positional coordinates.

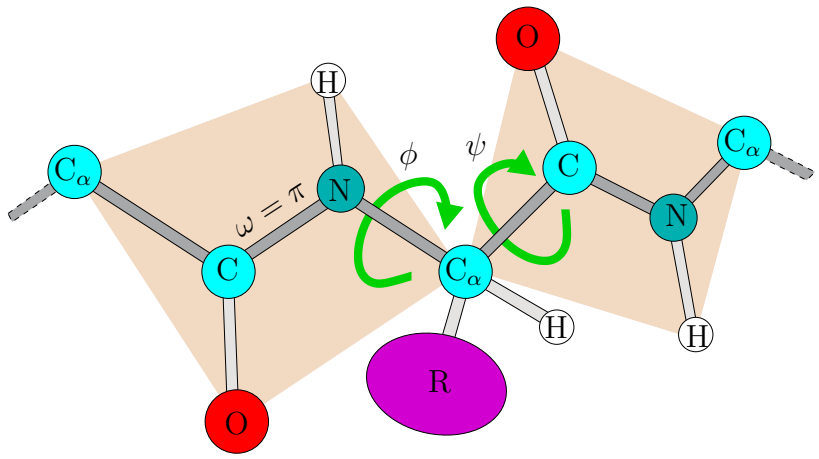


Figure 6.3: The elements in a protein are amino-acids connected via peptide bonds. Under the rigid geometry assumption, the peptide units are considered planar and only angles ϕ and ψ are variable.

Particularities of Proteins

A protein is formed by one or several *polypeptide chains* joined together. A polypeptide chain is a sequence of *amino-acids* (also called *residues*). The twenty amino-acids found in proteins have a similar structure. They have a common *backbone* of an organic carboxylic acid group and an amino group attached to a saturated carbon atom C_α; and a *side-chain* (marked as R in Figure 6.3) specific to each particular amino-acid. Side-chains have very different structure, and normally they also involve rotational bonds. Consecutive amino-acids in the polypeptide chain are held together by chemical bonds between the carboxy group of the one and the amino group of the other. The resulting C-N bond has a double-bond character making it particularly rigid. It is called a *peptide bond*. Under the rigid geometry assumption, the whole arrangement of the four C,O,N,H atoms as well as the two attached carbons C_α in a *peptide unit* is considered planar, i.e. the peptide bond is fixed with $\omega = 0$ or π [2]. Thus, the *protein backbone*, formed by the enchainment of amino-acid's backbones, only rotates around N-C_α bounds (angle ϕ) and C_α-C bounds (angle ψ). With a finest modeling, peptide bond angles ω can slightly variate. Figure 6.3 illustrates the structure of two-peptide-unit segment.

The kinematic model of a protein is then composed of a set of chains: the main-chain (the backbone) and the side-chains, which are built upon it. The conformation can be specified by an array:

$$q = \{q_{bkb}, q_{sch_1}, \ldots, q_{sch_n}\}$$

where q_{bkb} is the backbone conformation and q_{sch_i} is the conformation of each side-chain.

6.2.2 Van der Waals Model

The VdW model consists of a representation of the molecule by the union of solid spheres associated with atoms. A VdW radius is assigned to each atom type. This geometric model of the molecule is the simplest and most ordinary space-filling diagram [Edelsbrunner 98]. In molecular models treated by our approach, such spheres are the mobile bodies of the articulated polypeptide segment and the static obstacles corresponding to the rest of the atoms in the molecule, which compose what we call the environment. The left image in Figure 6.1 corresponds to the VdW model of amylosucrase.

From an energetic point of view, the sum of the VdW radii of two atoms represents the equilibrium distance of the electromagnetic interaction between them. This interaction is strongly repulsive in close proximity and weakly attractive for an intermediate range of distances. Distances between non-bonded atoms that are substantially shorter than the sum of their VdW radii are impossible.

The controversy when using such models concerns the choice of the VdW radii. Differ-

[2] When $\omega = \pi$ (like in Figure 6.3), the peptide unit is in form *trans*. This is the normal form found in proteins. The form *cis* (with $\omega = 0$) seldom appears in peptide units involving proline.

ent equilibrium distances are obtained from the different proposed equations of the VdW forces. Besides, such distances are associated with pairs of atoms, and thus a certain amount of ambiguity is introduced when determining values for individual atoms. Several slightly different tables of the VdW radii are available in literature (see [Bondi 64] for example).

Even more ambiguous can result the choice of the limiting contact distance between non-bonded atoms. This distance is obviously shorter than the sum of the VdW radii. Several tables of such distances are also available. In [Ramachandran 68] for example, authors propose *normal limits* and *extreme limits* for various inter-atomic contacts obtained from an analysis of a number of examples of crystal structures.

For our experiments, we model molecules using a percentage (usually 70%) of the VdW radii proposed in [Bondi 64]. This volume reduction is not only justified by the mentioned energetically possible penetrations between VdW spheres, but also by a relaxation of constraints imposed by the rigid geometry assumption. Collisions between such reduced VdW spheres must be avoided if they are separated by more than three bonds. This condition must be satisfied between the atoms of the articulated segment and between these atoms and the static atoms of the rest of the molecule.

6.3 Conformational Sampling

Algorithm 6.1 computes a random conformation of a polypeptide segment (the protein loop) achieving loop closure and clash avoidance constraints on the 3D model. First, the backbone conformation q_{bkb} is generated by the algorithm SingleLoop_RLG (Algorithm 3.1). We explain particularities of the treatment of protein models with RLG in Section 6.3.1. The n_{bkb_conf} conformations returned by SingleLoop_RLG are then tested for clashes of backbone atoms between themselves and with atoms in the environment. For each feasible conformation of the backbone, random conformations of the side-chains are tested. These chains are built iteratively until all of them are free of clashes. Section 6.3.2 explains the process.

6.3.1 Backbone Conformation

General explanations of the algorithm SingleLoop_RLG (see Section 3.2) stay for the application to a molecular chain model. Active and passive joint variables are chosen as consecutive dihedral angles in the backbone. For a polypeptide backbone model under the rigid geometry assumption, only dihedral angles ϕ an ψ are variable. The passive subchain, which has to involve 6 degrees of freedom, is then composed of the backbone of three residues. Although the passive subchain can be placed anywhere in the closed kinematic chain, in this case, it seems more suitable to place it in the middle. Next we comment particular considerations concerning the two main functions in the algorithm SingleLoop_RLG: Sample_q^a and Compute_q^p.

Algorithm 6.1: RANDOM_LOOP_CONFORMATION

```
input  : the loop A, the rest of the protein B
output : the conformations q[n_sol]
begin
    q_bkb[n_bkb_conf] ← SINGLELOOP_RLG(A.bkb);
    n_sol ← 0;
    for i = 1 to n_bkb_conf do
        if not CLASHCHECK(q_bkb(i), A.bkb, B) then
            if q_sch ← GENERATESIDECHAINS(q_bkb(i), A, B) then
                n_sol ← n_sol + 1;
                q(n_sol) ← COMPOUNDCONF(A, q_bkb(i), q_sch);
    if n_sol = 0 then return Failure;
end
```

SAMPLE_q^a : As we explained in Section 3.2.3, the crux of the RLG algorithm is to compute the bounding volumes RWS used for estimating the closure range of joint variables. We proposed a simple general procedure for obtaining the spherical shell dimension r_{ext} and r_{int}, corresponding to me maximum and minimum extension of the kinematic chain. This procedure is also appropriated for a polypeptide backbone. Particularities in the geometry of this chain allow to simplify some operations. Indeed, for segments containing more than three residues (the size of the passive subchain) r_{int} can be simply considered zero without decreasing the performance of the technique. Then, it only remains to obtain r_{ext}. We choose frames F_c, where the loop is broken, as the frames attached to atoms. The maximum distance between the extreme atoms of a segment of polypeptide backbone [3] is often obtained for a conformation with all the dihedral angles at π. We call this length r_π. However, this assumption is not always true, in particular if a slight rotation around peptide bonds is allowed. Thus, r_π is an accurate approximated value that we can use as $\tilde{r}_{ext}$. The upper bound of the maximum $\hat{r}_{ext}$, required for guaranteeing completeness, can be obtained as the sum of the distances between consecutive C_α atoms (i.e. the length of peptide units). Obviously, when the chain begins or ends with a fragment of peptide unit (i.e. only one or two of the three concerned atoms in the backbone are contained in the chain), the length of this portion is added. Then, we can sample r_{ext} from a distribution between r_π and $\hat{r}_{ext}$.

This approximated method to obtain r_{ext} is not dependent on a particular kind of geometry of peptide units. It can be applied to standard models or to structures acquired from the Protein Data Bank (PDB) [4]. Figure 6.4 illustrates the application to backbone segments with standard Pauling-Corey geometry [Pauling 60].

[3] Without proline. This case, studied apart, is not detailed here.

[4] PDB web site: `http://www.rcsb.org/pdb`

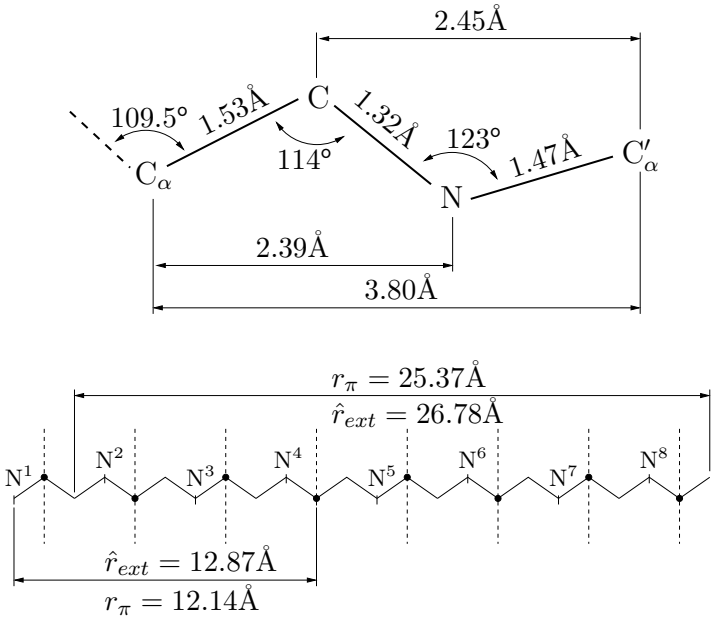


Figure 6.4: Maximum extension of polypeptide backbone with Pauling-Corey geometry.

COMPUTE_q^p : The kinematic model of the three-residue backbone corresponding to the passive subchain in our approach can be seen as a $6R$ manipulator of general geometry. The inverse kinematics problem for such a type of articulated mechanism has received the most attention in Robot Kinematics. Some authors have called it "the Mount Everest of Kinematics" [Roth 94]. The first attempt to solve this problem was made by Pieper in the 60's [Pieper 68]. Today, a variety of computationally efficient solutions are available [Nielsen 97].

We apply an algebraic elimination method to solve the general $6R$ inverse kinematics problem. Like many of related methods, it is inspired by the works of Lee and Liang [Lee 88b, Lee 88a]. The principle of this solution is described in [Renaud 00] [5]. The elimination of variables starts in a similar way to that in related works (e.g. [Raghavan 89, Manocha 94]). However, Renaud goes further in the elimination process, arriving at an 8×8 quadratic polynomial matrix in one variable instead of the 12×12 matrix in the referred methods. The problem can then be treated as a generalized eigenvalue problem (as previously proposed in [Manocha 94]), for which efficient and robust solutions are available [Kwakernaak 94]. Another important advantage of the method in relation to all previous approaches is that it requires a minimum number of divisions in the elimination process. In particular, divisions by zero are avoided in order to guarantee robustness.

[5] The author is currently working on an extended version with full technique details.

6.3.2 Side-Chain Conformation

The function GenerateSideChains builds the conformation of the side-chains upon a feasible backbone conformation. Random conformations are generated (by randomly sampling the the side-chain dihedral angles) and tested until one without forbidden overlappings is found. A progressive construction is carried out. Instead of rebuilding all the side-chains when the collision test is positive, only the conformation of a side-chain which clashes is resampled. The resampling and collision detection process is performed following an arbitrary order of the side-chains, intending to prevent a privileged conformational sampling. When two side-chains collide together, but self-clashes or clashes with the backbone and the environment do not exist, only one of them is resampled. The process is iterated a certain number of times before returning that a clash-free conformation of the side-chains cannot be found.

In our current implementation of the approach, clashes in a sampled conformation are checked by a generic collision detection algorithm [van Geem 01a], which operates well within geometrically complex 3D scenes.

6.4 Conformational Space Exploration

Sampling-based motion planning techniques explained in Chapter 3 can be applied to explore the subset of loop conformations satisfying closure and clash avoidance constraints. Such conformations correspond to the subset $\mathcal{C}_{free}$ in our formulation (see Chapter 1). In protein models, $\mathcal{C}_{free}$ is very reduced with relation to $\mathcal{C}$. The restricted free space between spheres of the VdW model makes feasible motions very constrained. Thus, the RRT approach is more suitable for this application.

The starting point q_{root} for the search can be a randomly sampled feasible conformation (e.g. generated by the technique explained in Section 1.3) or a known conformation (e.g. acquired from the PDB). Since $\mathcal{C}$ maybe composed of several disjoint manifolds and collision-free portions of each manifold can be also disjoint, the RRT algorithm can explore only a connected region in $\mathcal{C}_{free}$. If the aim is to explore the whole subset, then several starting points are required. An algorithm combining RRT and PRM techniques could be devised to achieve the exploration of the whole sub-space.

Note that the improvements of RRT-based algorithm we mentioned in Section 3.4.2 are particularly important for the current application. On the one hand, estimating the coverage of the explored manifold is necessary in order to determine a stop condition. On the other, limiting the number of nodes in the search tree while ensuring a good coverage increases the search speed and yields a data structure easier to handle.

Exploration with Flexible Geometry

Considering fixed values for bond lengths, bond angles and double bond torsion angles is a well accepted assumption that reduces the complexity of the structural analysis of molecules. However, it implies a severe restriction for conformational space exploration [Bruccoleri 85]. The rigid geometry assumption can be relaxed by allowing a slight variation of these parameters within given intervals. Handling these new variables is not a hard problem for our exploration algorithm, proceeding as follows. To generate a conformation q_{rand}, parameters d_i, α_i and ω_j of the molecular model (see Section 6.2) are first randomly sampled within the defined intervals. Then, the approach explained in Section 1.3 can be used. In the incremental variation of the selected conformation q_{near} toward q_{rand}, the new parameters are treated like the rest of the active variables (i.e. they are handled by the steering method).

6.5 First Results: Loop 7 Motions of Amylosucrase from Neisseria Polysaccharea

Amylosucrase (AS) is a glucansucrase that catalyzes the synthesis of an amylose-like polymer from sucrose. In the Carbohydrate-Active enZYme database (CAZy) [6], this enzyme is classified in family 13 of glucoside-hydrolases (GH), which mainly contains starch converting enzymes (hydrolases or transglycosidases). Remarkably, this enzyme is the only polymerase acting on sucrose substrate reported in this family, all the other glucansucrases being gathered in GH family 70. Which structural features are involved in AS specificity is an important fundamental question. Indeed, the structural similarity of AS to family 13 enzymes is high. The 3D structure reveals an organization in 5 domains [Skov 01]. Three of them are commonly found in family 13: a catalytic $(\beta/\alpha)_8$ barrel domain, a B domain between β-strand 3 and α-helix 3 (loop 3) and a C terminal Greek key domain. Two additional domains are found in AS only: a helical N-terminal domain and a domain termed B' formed by an extended loop between β-strand 7 and α-helix 7. Domain B' partially covers the active site located at the bottom of a pocket and is mainly responsible for this typical architecture. Recently, co-crystallization of AS with maltoheptaose revealed the presence of two maltoheptaose binding sites, the first one (OB1) in the main access channel to the active site and a second one (OB2) at the surface of domain B'. Soaking AS crystals with sucrose also revealed the presence of a second sucrose binding site (SB2) different from the active site initially identified [Skov 02]. The comparison of the various structures obtained suggests that the motion of a 17-residue fragment of domain B', starting at residue Gly^{433} and ending at residue Gly^{449}, consecutive to oligosaccharide binding, could facilitate sucrose translocation from SB2 to the active site. In the following part, this fragment will be called loop 7. This loop could play a pivotal role responsible for the structural change and the polymerase activity. In this context, molecular simulation of loop 7 motion appears

[6] CAZy web site: `http://afmb.cnrs-mrs.fr/~cazy/CAZY/index.htm`

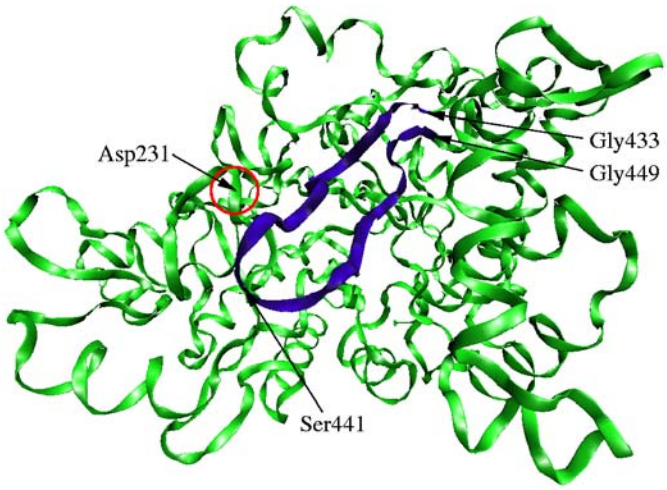


Figure 6.5: Loop 7 in amylosucrase from *Neisseria polysaccharea* and important residues in our study.

to be crucial to gain new insight into AS structure-function relationships.

Figure 6.5 shows location of loop 7 in the crystallographic structure of AS. This figure also indicates the location of the residues we mention in the next paragraphs. The model for our tests was created from the PDB file containing this structure (PDB ID: 1G5A), considering loop 7 as an articulated mechanism and the rest of the atoms as static elements. Atoms were modeled with 70% of their VdW radii. Images on the left in Figure 6.6 represent the articulated VdW model of the loop and a portion of its environment. Under our modeling assumptions, the results of the geometric exploration showed that only slight conformational variations of the loop are possible if the backbone integrity is maintained and steric clashes are avoided. The image on the right in Figure 6.6.a shows the skeleton of the articulated segment and a representation of one of the search trees computed for this test. Nodes of the tree are graphically represented by the positions explored by the C_α atom of Ser^{441}, the middle residue of the loop. This result contradicts pre-supposed significant loop fluctuations. Of course, our approach is not deterministic and therefore we cannot guarantee that such a motion does not exist. However, after several exhaustive tests, we can assert that the probability of its existence is very low. The average size of the constructed trees is of 1000 nodes (without visibility-pruning). Computing one of such trees required about 4000 iterations of the algorithm expanding the RRT (Algorithm 2.2), and more than 20000 complete collision tests were made. The average computing time was 1 hour [7]. It should be noted that computing time is mostly spent in collision detection.

[7]Tests were performed using a Sun Blade 100 Workstation with a 500-MHz UltraSPARC-IIe processor.

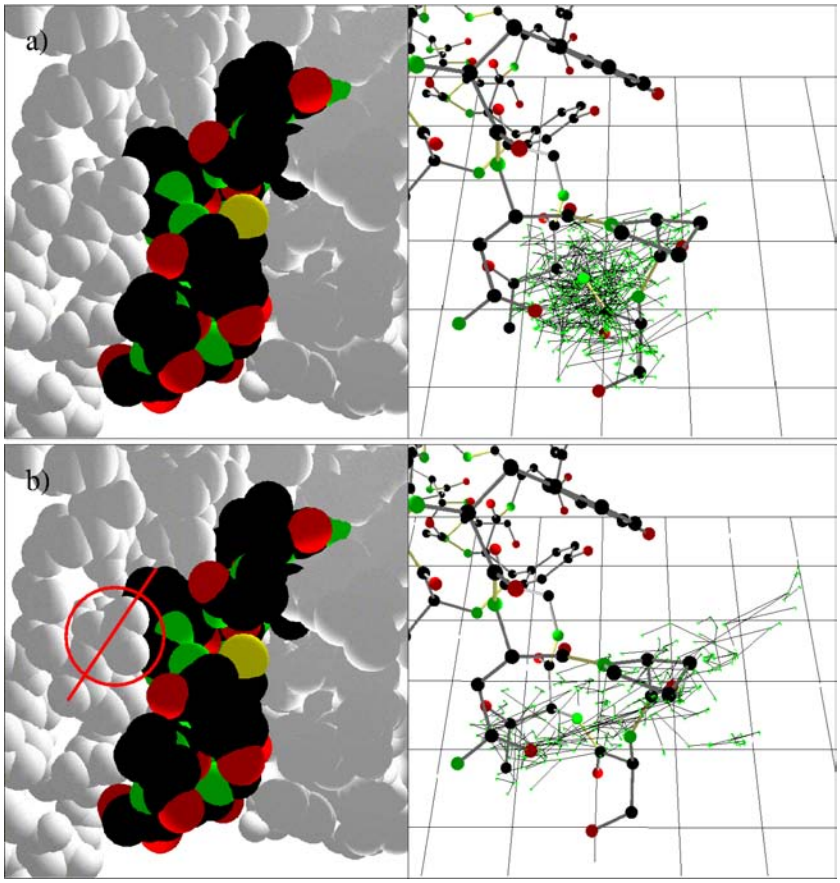


Figure 6.6: Exploration with (a) and without (b) the side-chain of Asp^{231}.

The generation of random conformations is very fast. For this loop, computing a conformation satisfying closure (including the update of all the frames and atom positions) attached to atoms) takes less than 0.1 seconds. The conformational sampling used by the exploration algorithm (i.e. a call to function SAMPLE_q^a) demands less than 0.01 seconds per conformation.

Several structural elements, and mainly loop 3 (residues 183-262), restrain the mobility of loop 7. Residue Asp^{231} was identified as the main "geometric lock", responsible for the loop 7 enclosing. The side-chain of this residue was removed from the model in order to simulate a possible conformational change of this chain or even of the whole loop 3. The conformational exploration in this case showed that the loop is able to effectuate the expected motions keeping geometric constraints. The C_α atom of Ser^{441} can be dislocated more than 9Å from its crystallographic position. Several tests were performed in order to see if the random nature of the approach could have an important influence on the nature of the results. Similar motions were obtained for all of them.

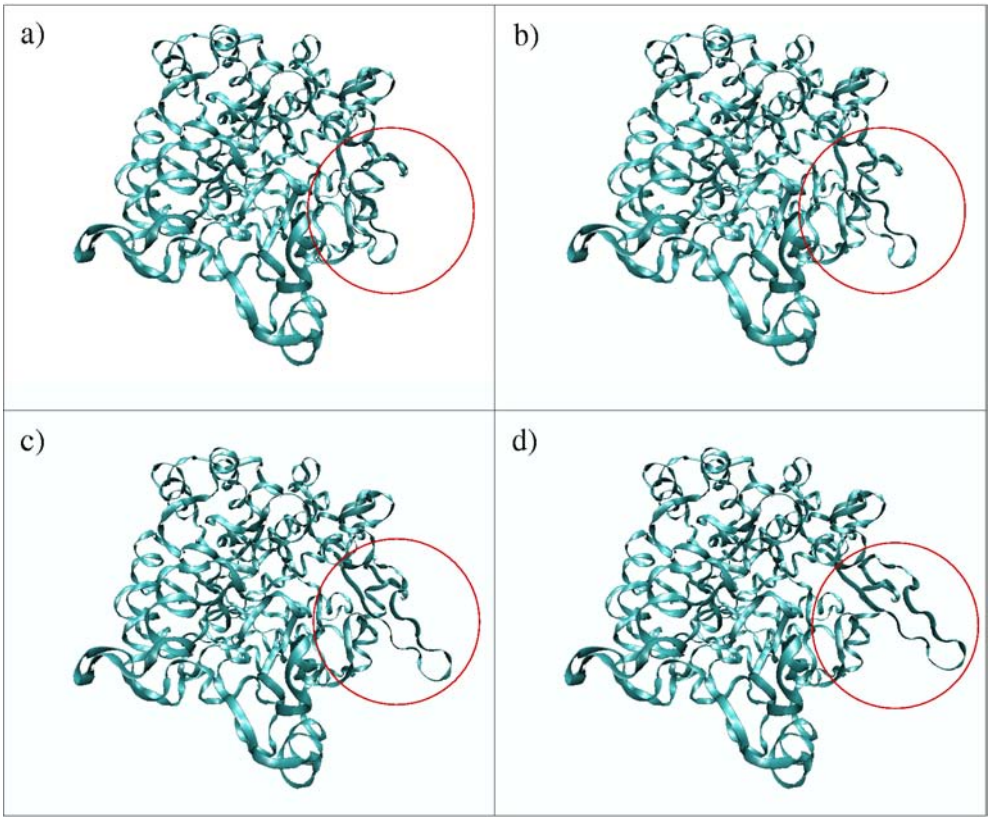


Figure 6.7: Simulated conformational gating of loop 7 in amylosucrase.

The loop moves almost as a rigid body with hinges at the extreme residues. Considerable variations of the backbone dihedral angles are concentrated in residues 433-436 and 446-449. Figure 6.6.b shows the representation of the search tree constructed in one of these tests. The images in Figure 6.7 correspond to four frames of the conformational change encoded in the RRT. Therefore, an "opening/closing" mechanism similar to other enzymes (e.g. [Derreumaux 98, Osborne 01]), termed *conformational gating*, is suspected for this loop. The role that residue Asp^{231} could play in this mechanism is being investigated. Directed mutagenesis experiments, replacing residue Asp^{231} by glycine, are currently being developed.

6.6 Discussion and Prospects

We have proposed techniques that provide powerful filters for conformational search methods. Our solution of the loop closing problem is complete (in the sense that no possible solution is missed), computationally efficient and its performance is only slightly affected

by the length of the molecular chain. Concerning the avoidance of steric clashes, collision detection algorithms combined with smart sampling techniques constitute an attractive alternative to methods producing optimization-based rearrangements.

Current improvements of the technique are related to clash avoidance. We are developing a tailored collision detection algorithm for molecular models which should perform faster tests. In addition, a different progressive process for building backbone conformations is going to be tried. In contrast to the described sampling approach, clashes between the backbone atoms and the static environment will be checked after each step of the RLG algorithm.

In our current implementation, values for all variable dihedral angles in the side-chains and the backbone are randomly sampled in the interval $(-\pi, \pi]$. As in other related techniques, our approach could handle information of the statistically preferred values of these angles (e.g. from Ramachandran plots by residue type). Using this information, many local steric clashes should be implicitly avoided.

Concerning the exploration technique, we are working on improved RRT-based algorithms (commented in Section 3.4.2). The visibility-pruning aims to resolve drawbacks of this approach, and preliminary results seem promising. We are also working on a stop condition for the algorithm depending on a estimation of the coverage.

Our algorithms treat conformations of a molecular segment in a static environment. The extension of these algorithms to handle the flexibility of side-chains in this environment could be done without difficulty. Handling several loops which share the same region of the space (e.g. antibody hypervariable loops [Bruccoleri 88]) is an interesting extension we expect to develop.

First results of the application of our robotic approach to molecular models show the potential of this technique. A fast geometric process can help to find the answer to important biochemical questions such as: which are the crucial residues in the biochemical reaction ? and what is the nature of conformational changes ?

Although our next goal is to improve this geometrically constrained exploration, the final aim is to incorporate the energetic analysis into the incremental search techniques. An energy function can easily be integrated into sampling-based motion planning algorithms. Indeed, impressive results have been obtained by conformational search methods inspired by these techniques applied to computer assisted drug design [Finn 98, LaValle 00], protein folding [Apaydin 02, Amato 02] and ligand-protein docking [Apaydin 01, Bayazit 01]. Given an energy function, geometrically feasible conformations generated by our approach could be evaluated and labeled, and then only the subset of the conformational space $\mathcal{C}_{feas}$ below a certain energetic limit should be explored.

Conclusions

We have presented an extended formulation of the motion planning problem under kinematic loop closure constraints and we have introduced a framework for the application of sampling-based algorithms on it. The use of sampling-based planners to solve this problem is strongly justified since no practical method is available yet to obtain a representation of the configuration-space of general closed-chain mechanisms. The guidelines and techniques that we have provided are general, independent of a particular implementation. In addition, we have given details on how to extend PRM-based and RRT-based algorithms to treat closed kinematic chains. The results obtained when solving difficult problems with these extended planners demonstrate the efficacy of the approach.

The algorithms that we have presented aim to be general tools, with application in many different domains. In the second part of this thesis we have discussed the application to the domains that we have already investigated. However, there are many other areas whose techniques should greatly benefit from the integration of our algorithms.

Our work on parallel mechanisms represents the first effective application of sampling-based motion planning algorithms to this kind of articulated structures. The generality of our approach is demonstrated by the complexity of the systems that it is able to treat, such as the model of the Logabex-LX4 (Figure 4.7), whose configuration-space is a 25-dimensional variety embedded in a 97-dimensional manifold. The same approach allows to solve difficult coordinated manipulation problems, such as the "robotized" version of the piano mover's problem (Figure 1), that remained unsolved.

The extended motion planning algorithms that we have explained have been integrated within a novel manipulation planning approach. The clever idea is to explore the connectivity of the subset where the manipulation sub-paths meet via a virtual closed-chain mechanism consisting of the robot together with the movable object placed at a stable position. Our manipulation planner automatically generates, among continuous sets, the grasps and the intermediate placements of the movable object required to solve complicated problems. It is the first general manipulation planner with this capability.

We have developed applications out of the field of Robotics. Motion planning techniques can be used as new tools that can help to solve important open problems in Computational Biology. The algorithms that we have presented can act as efficient filters for conformational search methods by making a geometric treatment of some strong energetic constraints. Techniques for predicting protein structures and protein interactions could greatly benefit from the integration of our geometric algorithms.

Future Research

Several points remain for future research. Some of them concern the RLG sampling algorithm. RLG has provided good results in all our experiments. Nevertheless, a deeper analytical work is necessary. The difficulty of this work arises from the great diversity of mechanical systems that should be studied in order to determine general properties. Because of the conservativeness of the approximations handled in the sampling process, the capacity of RLG to sample the whole configuration-space seems reasonable. However, we still have to work on a formal analysis of the technique. A characterization of the efficiency of RLG and the properties of samples (e.g. uniformity) depending on the accuracy of the bounds used by the algorithm also remains to be done.

The implemented version of RLG uses pseudo-random sequences to generate samples. The drawback of using such random samples within motion planning algorithms is that the performance of these algorithms is difficult to characterize. New forms of sampling that allow an easier analysis and control of algorithms must be investigated. Quasi-random sequences or multi-resolution grids, proposed in recent articles [Branicky 01, Lindemann 03, LaValle 03b], seems to be an interesting way to follow. The use of these other sampling methods within RLG should be studied in the future.

Another subject that requires future work concerns singularities. General techniques permitting to globally identify subsets of singular configurations should be a perfect complement to our algorithms. When the goal is to explore the connectivity of the whole configuration-space, they could allow to establish connections between disjoint subsets of regular configurations. For many practical applications (e.g. trajectory planning of parallel robots), they could allow to avoid paths passing through singularities. Unfortunately, as far as we know, such general global techniques are not available. Existing approaches are either local or dedicated to particular mechanisms (e.g. [Gosselin 90b, Alizade 85]). A general methodology for the treatment of singularities within sampling-based motion planning algorithms remains an open topic.

Recent interval methods [Merlet 01, Porta 03] appear to be another matter to study for the improvement of motion planning techniques for closed-chain mechanisms. Interval methods provide a complete approximated representation of the variety of the configurations satisfying loop closure equations. They compute a set of boxes that contain the continuum of solutions. Such a representation is very suitable for the application of sampling-based motion planning algorithms. Although the applicability of interval methods stays still limited by the number of variables that they can handle, future developments seem promising.

Concerning the applications, following the way that we have started on the development of algorithms for the conformational analysis of protein loops evokes a big interest. The first goal to attain is to get efficient algorithms that have some kind of guaranty (e.g. probabilistic or resolution completeness) to capture the whole geometrically feasible subset of the conformational space of one loop. Then, the aim will be to extend these

algorithms to several loops in the same protein. The final goal is to handle several proteins that interact while changing the conformation of loops on their surfaces. Although the principle for the efficacy of our approach is to stay in a geometric formulation, the use of simplified energetic potentials should be also studied. These energies could allow to consider important constraints for the conformational analysis that cannot be treated geometrically.

Another application with great demand in the present is graphic animation. The generation of realistic motions for human-like characters, requires further investigation on the combination of motion planning algorithms and other techniques, such as motion capture [Pettre 02]. Complex planners that consider closure/contact constraints or manipulation constraints must be devised for treating several characters able to interact with each other. Based on the approaches presented in this thesis, more sophisticated motion planning and manipulation planning algorithms could be developed for such multi-character systems, suitable for the integration of particular constraints that produce realistic motions.

Finally, in the fields of Robotics and virtual prototyping, another application of our algorithms would be to compute motions of objects in contact. Existing approaches for contact motion planning (e.g. [Ji 01]) do not consider the constraints (such as collision avoidance, workspace limits or singularities) imposed by the manipulator that handles the mobile object. Simultaneously considering contact constraints and these other motion constraints can be made via a virtual closed-chain mechanism, consisting of the mobile object and the manipulator. Constraints imposed in the motion of this composed mechanism in order to maintain a determined class of contact can be expressed like kinematic loop closure constraints. The techniques that we have presented, combined with other techniques to identify classes of contacts (e.g. [Xiao 01]), could solve this kind of problems.

Bibliography

[Agarwal 98] P. Agarwal, L.E. Kavraki & M. Mason. Robotics: The algorithmic perspective. WAFR1998. A.K. Peters, 1998.

[Ahuactzin 98] J.M. Ahuactzin, K. Gupta & E. Mazer. *Manipulation Planning for Redundant Robots: A Practical Approach.* International Journal of Robotics Research, vol. 17(7), pages 731–747, 1998.

[Ahuactzin 99] J.M. Ahuactzin & K. Gupta. *The Kinematic Roadmap: A Motion Planning Based Approach for Inverse Kinematics of Redundant Robots.* IEEE Transactions on Robotics and Automation, vol. 15(4), pages 653–669, 1999.

[Akinc 03] M. Akinc, K.E. Bekris, B.Y. Chen, A.M. Ladd, E. Plaku & L.E. Kavraki. *Probabilistic Roadmaps of Trees for Parallel Computation of Multiple Query Roadmaps.* Proc. 11th Int. Symp. on Robotics Research, 2003. In press.

[Alami 89] R. Alami, T. Siméon & J.-P. Laumond. *A Geometrical Approach to Planning Manipulation Tasks. The Case of Discrete Placements and Grasps.* Proc. 5th Int. Symp. on Robotics Research, pages 453–463, 1989.

[Alami 95] R. Alami, J.-P. Laumond & T. Siméon. *Two Manipulation Planning Algorithms.* In K. Goldberg, D. Halperin, J.-C. Latombe & R. Wilsonet, editors, Algorithmic Foundations of Robotics (WAFR1994), pages 109–125. A.K. Peters, 1995.

[Alizade 85] R.I. Alizade & G.N. Sandor. *Determination of the Condition of Existence of Complete Crank Rotation and of the Instantaneous Efficiency of Spatial Four-Bar Mechanisms.* Mechanism and Machine Theory, vol. 20(3), pages 155–163, 1985.

[Amato 98] N.M. Amato, O.B. Bayazit, L.K. Dale, C. Jones & D. Vallejo. *OBPRM: An Obstacle-Based PRM for 3D Workspaces.* In P. Agarwal, L.E. Kavraki & M. Mason, editors, Robotics: The Algorithmic Perspective (WAFR1998), pages 155–168. A.K. Peters, 1998.

[Amato 00] N.M. Amato, O.B. Bayazit, L.K. Dale, C. Jones & D. Vallejo. *Choosing Good Distance Metrics and Local Planners for Probabilistic Roadmap Methods.* IEEE Transactions on Robotics and Automation, vol. 16(4), pages 442–447, 2000.

[Amato 02] N.M. Amato, K.A. Dill & G. Song. *Using Motion Planning to Study Protein Folding Pathways.* Journal of Computational Biology, vol. 9(2), pages 149–168, 2002.

[Angeles 88] J. Angeles & C. Gosselin. *Détermination du Degré de Liberté des Chaînes Cinématiques.* Transactions of the Canadian Society of Mechanical Engineering, vol. 12(4), pages 219–226, 1988.

[Angeles 03] J. Angeles. Fundamentals of robotic mechanical systems: Theory, methods and algorithms. Springer-Verlag, 2003.

[Apaydin 01] M.S. Apaydin, A.P. Singh, D.L. Brutlag & J.-C. Latombe. *Capturing Molecular Energy Landscapes with Probabilistic Conformational Roadmaps.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 932–939, 2001.

[Apaydin 02] M.S. Apaydin, D.L. Brutlag, C. Guestrin, D. Hsu. & J.-C. Latombe. *Stochastic Roadmap Simulation: An Efficient Representation and Algorithm for Analyzing Molecular Motion.* Proc. RECOMB'02, pages 12–21, 2002.

[Artobolevski 77] I. Artobolevski. Théorie des mécanismes et des machines. Editions Mir, 1977.

[Atramentov 02] A. Atramentov & S.M. LaValle. *Efficient Nearest Neighbor Searching for Motion Planning.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 632–637, 2002.

[Baker 01] D. Baker & A. Šali. *Protein Structure Prediction ans Structural Genomics.* Science, vol. 295, pages 93–96, 2001.

[Barraquand 91] J. Barraquand & J.-C. Latombe. *Robot Motion Planning: A Distributed Representation Approach.* International Journal of Robotics Research, vol. 10(6), pages 628–649, 1991.

[Barraquand 93] J. Barraquand & J.-C. Latombe. *Nonholonomic Multibody Mobile Robots: Controllability and Motion Planning in the Presence of Obstacles.* Algorithmica, vol. 10, pages 121–155, 1993.

[Barraquand 94] J. Barraquand & P. Ferbach. *A Penalty Function Method for Constrained Motion Planning.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 1235–1242, 1994.

[Barraquand 97] J. Barraquand, L.E. Kavraki, J.-C. Latombe, T.Y. Li, R. Motvani & P. Raghavan. *A Random Sampling Scheme for Path Planning.* International Journal of Robotics Research, vol. 16(6), pages 759–774, 1997.

[Basu 00] S. Basu, R. Pollack & M.-F. Roy. *Computing Roadmaps of Semi-algebraic Sets on a Variety.* Journal of the American Mathematical Society, vol. 13(1), pages 55–82, 2000.

[Bayazit 01] O.B. Bayazit, G. Song & N.M. Amato. *Ligand Binding with OBPRM and User Input*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 954–959, 2001.

[Bedrossian 90] N.S. Bedrossian. *Classification of Singular Configurations for Redundant Manipulators*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 818–823, 1990.

[Bellman 57] R. Bellman. Dynamic programming. Princeton Univ. Press, 1957.

[Bennett 03] G.T. Bennett. *A New Mechanism*. Engineering, vol. 78, pages 777–778, 1903.

[Bessière 93] P. Bessière, J.M. Ahuactzin, E.-G. Talbi & E. Mazer. *The "Ariadne's Clew" Algorithm: Global Planning with Local Methods*. Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pages 1373–1380, 1993.

[Bliek 01] C. Bliek, P. Spellucci, L.N. Vicente, A. Neumaier, L. Granvilliers, E. Monfroy, F. Benhamou, E. Huens, P. van Hentenryck, D. Sam-Haroud & B. Faltings. *Algorithms for Solving Nonlinear Constrained and Optimization Problems: The State of the Art*. Progress Report of the COCONUT Project, 2001.

[Bochnak 98] J. Bochnak, M. Coste & M.-F. Roy. Real algebraic geometry. Springer, 1998.

[Bohlin 00] R. Bohlin & L.E. Kavraki. *Path Planning using Lazy PRM*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 521–528, 2000.

[Boissonnat 00] J.-D. Boissonnat, O. Devillers & S. Lazard. *Motion Planning of Legged Robots*. SIAM Journal on Computing, vol. 30(1), pages 218–246, 2000.

[Bondi 64] A. Bondi. *Van der Waals Volumes and Radii*. Journal of Physical Chemistry, vol. 68, pages 441–451, 1964.

[Boor 99] V. Boor, M.H. Overmars & A.F. van der Stappen. *The Gaussian Sampling Strategy for Probabilistic Roadmap Planners*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 1018–1023, 1999.

[Borrel 86] P. Borrel. *A Study of Manipulator Inverse Kinematic Solutions with Application to Trajectory Planning and Workspace Determination*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 1180–1185, 1986.

[Bostelman 01] R. Bostelman & J. Albus nad W.C. Stone. *Toward Next-Generation Construction Machines*. Proc. American Nuclear Society 9th Int. Topical Meeting on Robotics and Remote Systems, 2001. Available at `http://fire.nist.gov/bfrlpubs/build01/art002.html`.

[Branicky 01] M.S. Branicky, S.M. LaValle, K. Olson & L. Yang. *Quasi-Randomized Path Planning*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 1481–1487, 2001.

[Bruccoleri 85] R.E. Bruccoleri & M. Karplus. *Chain Closure with Bond Angle Variations*. Biopolymers, vol. 18, pages 2767–2773, 1985.

[Bruccoleri 87] R.E. Bruccoleri & M. Karplus. *Prediction of the Folding of Short Polypeptide Segments by Uniform Conformational Sampling*. Biopolymers, vol. 26, pages 137–168, 1987.

[Bruccoleri 88] R.E. Bruccoleri, E. Haber & J. Novotny. *Structure of Antibody Hypervariable Loops Reproduced by a Conformational Search Algorithm*. Nature, vol. 335, pages 564–568, 1988.

[Buchberger 82] B. Buchberger. *Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory*. In N. K.Bose, editor, Recent Trends in Multidimensional Systems Theory, pages 184–229. D. Reidel Publishing Company, 1982.

[Burdick 88] J.W. Burdick. *Kinematic Analysis and Design of Redundant Robot Manipulators*. PhD thesis, Stanford University, 1988.

[Burdick 89] J.W. Burdick. *On the Inverse Kinematics of Redundant Manipulators: Characterization of the Self-Motion Manifold*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 264–270, 1989.

[Candel 00] A. Candel & L. Conlon. Foliations 1. American Mathematical Society, 2000.

[Canny 88] J.F. Canny. The complexity of robot motion planning. MIT Press, 1988.

[Canutescu 03] A.A. Canutescu & R.L. Dunbrack Jr. *Cyclic Coordinate Descent: A Robotics Algorithm for Protein Loop Closure*. Protein Science, vol. 12(5), pages 963–972, 2003.

[Castellet 98] A. Castellet & F. Thomas. *An Algorithm for the Solution of Inverse Kinematics Problems Based on an Interval Method*. In M. Husty & A.J. Lenarcic, editors, Advances in Robot Kinematics, pages 393–403. Kluwer Academic Publishers, 1998.

[Celaya 93] E. Celaya & C. Torras. *On Finding the Set of Inverse Kinematics Solutions for Redundant Manipulators.* In J. Angeles, G. Hommel & P. Kovács, editors, Computational Kinematics, pages 85–94. Kluwer Academic Publishers, 1993.

[Chablat 98] D. Chablat. *Domaines d'Unicité et Parcourabilité pour les Manipulateurs Pleinement Parallèles.* PhD thesis, Ecole Centrale de Nantes, 1998.

[Chai 01] K.S. Chai & K. Young. *Designing a Stewart Platform-based Cooperative System for Large Component Assembly.* Proc. IEEE Int. Conf. on Methods and Models in Automation and Robotics, 2001. Available at `http://www.warwick.ac.uk/fac/sci/Eng/AMG/flextool/`.

[Chang 95] H. Chang & T.-Y. Li. *Assembly Maintainability Study With Motion Planning.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 1012–1019, 1995.

[Chang 00] K.-S. Chang, R. Holmberg & O. Khatib. *The Augmented Object Model: Cooperative Manipulation and Parallel Mechanism Dynamics.* Proc. IEEE Int. Conf. on Robotics and Automation., pages 470–475, 2000.

[Charentus 90] S. Charentus. *Modélisation et Commande d'un Robot Manipulateur Redondant Composé de Plusieurs Plateformes de Stewart.* PhD thesis, Université Paul Sabatier, 1990.

[Chatila 95] R. Chatila, S. Lacroix, T. Siméon & M. Herrb. *Planetary Exploration by a Mobile Robot: Mission Teleprogramming and Autonomous Navigation.* Autonomous Robots Journal, vol. 2, pages 1–12, 1995.

[Chatila 02] R. Chatila, R. Alami, T. Siméon, J. Pettre & L. Jaillet. *Safe, Reliable and Friendly Interaction between Humans and Humanoids.* 3rd IARP Int. Workshop on Humanoid and Human Friendly Robotics, pages 83–87, 2002.

[Chen 91] P.C. Chen & Y.K. Hwang. *Practical Path Planning Among Movable Obstacles.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 444–449, 1991.

[Cheng 01] P. Cheng & S.M. LaValle. *Reducing Metric Sensitivity in Randomized Trajectory Design.* Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pages 43–48, 2001.

[Cheng 02] P. Cheng & S.M. LaValle. *Resolution Complete Rapidly-Exploring Random Trees*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 267–272, 2002.

[Contreras-Moreira 02] B. Contreras-Moreira, P.W. Fitzjohn & P.A. Bates. *Comparative Modelling: An Essential Methodology for Protein Structure Prediction in the Post-Genomic Era*. Applied Bioinformatics, vol. 1(4), pages 177–190, 2002.

[Cortés 02a] J. Cortés & T. Siméon. *Deformable Handling Devices: Constrained Motion Planning and Closed Chain Systems*. In MOLOG Final Repport and Third Year Deliverables. 2002.

[Cortés 02b] J. Cortés, T. Siméon & J.-P. Laumond. *A Random Loop Generator for Planning the Motions of Closed Kinematic Chains using PRM Methods*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 2141–2146, 2002.

[Cortés 03a] J. Cortés & T. Siméon. *Probabilistic Motion Planning for Parallel Mechanisms*. Proc. IEEE Int. Conf. on Robotics and Automation (to appear), vol. To appear, 2003.

[Cortés 03b] J. Cortés, T. Siméon, M. Remaud-Siméon & V. Tran. *Geometric Algorithms for the Conformational Analysis of Long Protein Loops*. Journal of Computational Chemistry, 2003. Accepted with minor revisions.

[Craig 89] J.J. Craig. Introduction to robotics: Mechanics and control. Addison-Wesley, 1989.

[Crippen 92] G.M. Crippen. *Exploring the Conformational Space of Cycloalkanes by Linearized Embedding*. Journal of Computational Chemistry, vol. 13, pages 351–361, 1992.

[Dale 01] L.K. Dale & N.M. Amato. *Probabilistic Roadmaps - Putting It All Together*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 1940–1947, 2001.

[Dasgupta 98] B. Dasgupta & T.S. Mruthyunjaya. *Singularity-Free Path Planning for the Stewart Platform Manipulator*. Mechanism and Machine Theory, vol. 33(6), pages 711–725, 1998.

[Dasgupta 00] B. Dasgupta & T.S. Mruthyunjaya. *The Stewart Platform Manipulator: A Review*. Mechanism and Machine Theory, vol. 35(1), pages 15–40, 2000.

[DeMers 97] D. DeMers & K. Kreutz-Delgado. *Inverse Kinematics of Dextrous Manipulators*. pages 75–116. Academic Press, 1997.

[Derreumaux 98] P. Derreumaux & T. Schlick. *The Loop Opening/Closing Motion of the Enzyme Triosephosphate Isomerase*. Biophysical Journal, vol. 74, pages 72–81, 1998.

[Desai 99] J.P. Desai & V. Kumar. *Motion Planning for Cooperating Mobile Manipulators*. Journal of Robotic Systems, vol. 16(10), pages 557–579, 1999.

[Donald 93] B.R. Donald, P.G. Xavier, J.F. Canny & J.H. Reif. *Kinodynamic Motion Planning*. Journal of the ACM, vol. 40(5), pages 1048–1066, 1993.

[Donald 01] B.R. Donald, K.M. Lynch & D. Rus. Algorithmic and computational robotics: New directions. WAFR2000. A.K. Peters, 2001.

[Dress 88] A.W.M. Dress & T.F. Havel. *Shortest Path Problems and Molecular Conformation*. Discrete Applied Mathematics, vol. 19, pages 129–144, 1988.

[Edelsbrunner 98] H. Edelsbrunner. *Geometry for Modeling Biomolecules*. In P. Agarwal, L.E. Kavraki & M. Mason, editors, Robotics: The Algorithmic Perspective (WAFR1998), pages 265–277. A.K. Peters, 1998.

[Erdman 91] A.G. Erdman & G.N. Sandor. Mechanism design: Analysis and synthesis. Prentice Hall, 1991.

[Ferré 03] E. Ferré & J.-P. Laumond. *Maquette Numérique et Trajectoires de Désassemblage: Les Solutions Probabilistes*. Proc. MICAD'2003, pages 193–201, 2003.

[Finn 98] P.W. Finn, L.E. Kavraki, J.-C. Latombe, S. Venkatasubramanian, C. Shelton & A. Yao. *RAPID: Randomized Pharmacophore Identification*. Computational Geometry: Theory and Applications, vol. 10(4), pages 263–272, 1998.

[Fiser 00] A. Fiser, R.K. Do & A. Šali. *Modeling of Loops in Protein Structures*. Protein Science, vol. 9, pages 1753–1773, 2000.

[Fraichard 99] T. Fraichard. *Trajectory Planning In a Dynamic Workspace: A 'State-Time' Approach*. Advanced Robotics, vol. 13(1), pages 75–94, 1999.

[Frenkel 96] D. Frenkel & B. Smit. Understanding molecular simulation: From algorithms to applications. Academic Press, 1996.

[Garber 02] M. Garber & M.C. Lin. *Constraint-Based Motion Planning for Virtual Prototyping*. Proc. 7th ACM Symp. on Solid Model and Applications, pages 257–264, 2002.

[Gelfand 94] I.M. Gelfand, M.M. Kapranov & A.V. Zelevinsky. Discriminants, resultants and multidimensional determinants. Birkhäuser, 1994.

[Geraerts 02] R. Geraerts & M.H. Overmars. *A Comparative Study of Probabilistic Roadmap Planners*. Proc. Workshop on the Algorithmic Foundations of Robotics, 2002. In press.

[Gō 70] N. Gō & H.A. Scheraga. *Ring Closure and Local Conformational Deformations of Chain Molecules*. Macromolecules, vol. 3, pages 178–187, 1970.

[Goldberg 95] K. Goldberg, D. Halperin, J.-C. Latombe & R. Wilsonet. Algorithmic foundations of robotics. WAFR1994. A.K. Peters, 1995.

[Gorla 84] B. Gorla & M. Renaud. Modèles des robots manipulateurs: Application à leur commande. Cepadues, 1984.

[Gosselin 88] C. Gosselin. *Kinematic Analysis, Optimization and Programming of Parallel Robotic Manipulators*. PhD thesis, McGill University, 1988.

[Gosselin 90a] C. Gosselin. *Determination of the Workspace of 6-dof Parallel Manipulators*. ASME Journal of Mechanical Design, vol. 112, pages 331–336, 1990.

[Gosselin 90b] C. Gosselin & J. Angeles. *Singularity Analysis of Closed-Loop Kinematic Chains*. IEEE Transactions on Robotics and Automation, vol. 6(3), pages 281–290, 1990.

[Gottschalk 96] S. Gottschalk, M.C. Lin & D. Manocha. *OBB-Tree: A Hierarchical Structure for Rapid Interference Detection*. Computer Graphics, vol. 30, pages 171–180, 1996.

[Gottschlich 92] S.N. Gottschlich & A.C. Kak. *AMP-CAD: An Assembly Motion Planning System*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 2355–2360, 1992.

[Gough 56] V.E. Gough. *Contribution to Discussion of Papers on Research in Automobile Stability, Control and Tyre Performance*, 1956.

[Gravot 02] F. Gravot, R. Alami & T. Siméon. *Playing with Several Roadmaps to Solve Manipulation Problems*. Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pages 2311–2316, 2002.

[Gravot 03] F. Gravot, S. Cambon & R. Alami. *aSyMov: A Planner that Deals with Intricate Symbolic and Geometric Problems*. Proc. 11th Int. Symp. of Robotics Research, 2003. In press.

[Guibas 99] L.J. Guibas, D. Hsu & L. Zhang. *H-Walk: Hierarchical Distance Computation for Moving Convex Bodies*. Proc. 15th ACM Symp. on Computational Geometry, pages 265–273, 1999.

[Gupta 98] K. Gupta & A.P. del Pobil. Practical motion planning in robotics. John Wiley & Sons, 1998.

[Han 01] L. Han & N.M. Amato. *A Kinematics-Based Probabilistic Roadmap Method for Closed Kinematic Chains*. In B.R. Donald, K.M. Lynch & D. Rus, editors, Algorithmic and Computational Robotics: New Directions (WAFR2000), pages 233–245. A.K. Peters, 2001.

[Hansen 92] E.R. Hansen. Global optimization using interval analysis. Marcel Dekker, 1992.

[Henderson 97] D.W. Henderson & D. Taimina. Differential geometry: A geometric introduction. Prentice Hall, 1997.

[Hervé 78] J.M. Hervé. *Analyse Structurelle des Mécanismes par Groupe des Déplacements*. Mechanism and Machine Theory, vol. 13(4), pages 437–450, 1978.

[Hoffmann 89] C.M. Hoffmann. Geometric and solid modeling: An introduction. Morgan Kaufmann, 1989.

[Holleman 98] C. Holleman, L.E. Kavraki & J. Warren. *Planning Paths for a Flexible Surface Patch*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 21–26, 1998.

[Holleman 00] C. Holleman & L.E. Kavraki. *A Framework for Using the Workspace Medial Axis in PRM Planners*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 1408–1413, 2000.

[Hsu 97] D. Hsu, J.-C. Latombe & R. Motwani. *Path Planning in Expansive Configuration Spaces*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 2719–2726, 1997.

[Hsu 98] D. Hsu, L.E. Kavraki, J.-C. Latombe, R. Motwani & S. Sorkin. *On Finding Narrow Passages with Probabilistic Roadmap Planners*. In P. Agarwal, L.E. Kavraki & M. Mason, editors, Robotics: The Algorithmic Perspective (WAFR1998), pages 141–153. A.K. Peters, 1998.

[Hsu 99] D. Hsu, L.E. Kavraki, J.-C. Latombe & R. Motwani. *Capturing the Connectivity of High-Dimensional Geometric Spaces by Parallelizable Random Sampling Techniques*. In P.M. Pardalos & S. Rajasekaran, editors, Advances in Randomized Parallel Computing,

pages 159–182. Combinatorial Optimization Series, Kluwer Academic Publishers, 1999.

[Hsu 00] D. Hsu. *Randomized Single-Query Motion Planning in Expansive Spaces*. PhD thesis, Stanford University, 2000.

[Hsu 02] D. Hsu, R. Kindel, J.-C. Latombe & S. Rock. *Randomized Kinodynamic Motion Planning with Moving Obstacles*. International Journal of Robotics Research, vol. 21(3), pages 233–255, 2002.

[Hunt 78] K.H. Hunt. Kinematic geometry of mechanisms. Clarendon, 1978.

[Janin 03] J. Janin, K. Henrick, J. Moult, L.T. Eyck, M.J.E. Sternberg, S. Vajda, I. Vakser & S.J. Wodak. *CAPRI: A Critical Assessment of PRedicted Interactions*. Proteins: Structure, Function, and Genetics, vol. 52(1), pages 2–9, 2003.

[Ji 01] X. Ji & J. Xiao. *Planning Motion Compliant to Complex Contact States*. International Journal of Robotics Research, vol. 20(6), pages 446–465, 2001.

[Jiménez 98] P. Jiménez, F. Thomas & C. Torras. *Collision Detection Algorithms for Motion Planning*. In J.-P. Laumond, editor, Robot Motion Planning and Control, pages 305–343. Springer-Verlag, 1998.

[Jordan 99] D. Jordan & M. Steiner. *Configuration Spaces of Mechanical Linkages*. Discrete and Computational Geometry, vol. 22, pages 297–315, 1999.

[Kavraki 95a] L.E. Kavraki. *Random Networks in Configuration Space for Fast Path Planning*. PhD thesis, Stanford University, 1995.

[Kavraki 95b] L.E. Kavraki, J.-C. Latombe, R. Motwani & P. Raghavan. *Randomized Query Processing in Robot Motion Planning*. Proc. 27th ACM Symp. on Theory of Computing, pages 353–362, 1995.

[Kavraki 96] L.E. Kavraki, P. Svestka, J.-C. Latombe & M.H. Overmars. *Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces*. IEEE Transactions on Robotics and Automation, vol. 12(4), pages 566–580, 1996.

[Kavraki 97] L.E. Kavraki. *Geometry and the Discovery of New Ligands*. In J.-P. Laumond & M.H. Overmars, editors, Algorithms for Robotic Motion and Manipulation (WAFR1996), pages 435–448. A.K. Peters, 1997.

[Kavraki 98] L.E. Kavraki, M.N. Kolountzakis & J.-C. Latombe. *Analysis of Probabilistic Roadmaps for Path Planning*. IEEE Transactions on Robotics and Automation, vol. 14(1), pages 166–171, 1998.

[Khatib 86] O. Khatib. *Real-Time Obstacle Avoidance for Manipulators and Mobile Robots.* International Journal of Robotics Research, vol. 5(1), pages 90–98, 1986.

[Khatib 88] O. Khatib. *Object Manipulation in a Multi-Effector Robot System.* In R. Bolles & B. Roth, editors, Robotics Research 4, pages 137–144. MIT Press, 1988.

[Klein 83] C.A. Klein & C.H. Huang. *Review of Pseudo-Inverse Control for use with Kinematically Redundant Manipulators.* IEEE Transactions on Systems, Man and Cybernetics, vol. 13(3), pages 245–250, 1983.

[Koga 92] Y. Koga & J.-C. Latombe. *Experiments in Dual-Arm Manipulation Planning.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 2238–2245, 1992.

[Koga 94] Y. Koga & J.-C. Latombe. *On Multi-Arm Manipulation Planning.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 945–952, 1994.

[Koga 95] Y. Koga, K. Kondo, J.J. Kuffner & J.-C. Latombe. *Planning Motions with Intentions.* Proc. SIGGRAPH'94, pages 395–408, 1995.

[Kovács 93] P. Kovács & G. Hommel. *On the Tangent-Half-Angle Substitution.* Proc. Int. Workshop on Computational Kinematics, pages 27–40, 1993.

[Kuffner 99] J.J. Kuffner. *Autonomous Agents for Real-Time Animation.* PhD thesis, Stanford University, 1999.

[Kuffner 00] J.J. Kuffner & S.M. LaValle. *RRT-Connect: An Efficient Approach to Single-Query Path Planning.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 995–1001, 2000.

[Kwakernaak 94] H. Kwakernaak & M. Sebek. *Polynomial J-Spectral Factorization.* IEEE Transactions on Automatic Control, vol. 39(2), pages 315–328, 1994.

[Ladd 02] A. Ladd & L.E. Kavraki. *Generalizing the Analysis of PRM.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 2120–2125, 2002.

[Lamiraux 97] F. Lamiraux. *Robots Mobiles à Remorque: De la Planification de Chemins à Exécution de Mouvements.* PhD thesis, Institut National Polytechnique de Toulouse, 1997.

[Lamiraux 01a] F. Lamiraux & L.E. Kavraki. *Planning Paths for Elastic Objects under Manipulation Constraints*. International Journal of Robotics Research, vol. 20(3), pages 188–208, 2001.

[Lamiraux 01b] F. Lamiraux & J.-P. Laumond. *Smooth Motion Planning for Car-Like Vehicles*. IEEE Transactions on Robotics and Automation, vol. 17(4), pages 498–501, 2001.

[Lamiraux 03] F. Lamiraux, J.-P. Laumond, C. van Geem, D. Boutonnet & G. Raust. *Trailer-Truck Trajectory Optimization for Airbus A380 Component Transportation*. IEEE Robotics and Automation Magazine, 2003. In press.

[Latombe 91] J.-C. Latombe. Robot motion planning. Kluwer Academic Publishers, 1991.

[Latombe 99] J.-C. Latombe. *Motion Planning: A Journey of Robots, Molecules, Digital Actors, and Other Artifacts*. International Journal of Robotics Research, vol. 18(11), pages 1119–1128, 1999.

[Laumond 86] J.-P. Laumond. *Feasible Trajectories for Mobile Robots with Kinematic and Environment Constraints*. Proc. Int. Conf. on Intelligent Autonomous Systems, pages 346–354, 1986.

[Laumond 97] J.-P. Laumond & M.H. Overmars. Algorithms for robotic motion and manipulation. WAFR1996. A.K. Peters, 1997.

[Laumond 98a] J.-P. Laumond. Robot motion planning and control. Springer, 1998.

[Laumond 98b] J.-P. Laumond, S. Sekhavat & F. Lamiraux. *Guidelines in Nonholonomic Motion Planning for Mobile Robots*. In J.-P. Laumond, editor, Robot Motion Planning and Control, pages 1–53. Springer-Verlag, 1998.

[Laumond 01] J.-P. Laumond & T. Siméon. *Notes on Visibility Roadmaps and Path Planning*. In B.R. Donald, K.M. Lynch & D. Rus, editors, Algorithmic and Computational Robotics: New Directions (WAFR2000), pages 317–328. A.K. Peters, 2001.

[LaValle 98] S.M. LaValle. *Rapidly-Exploring Random Trees: A New Tool for Path Planning*. TR 98-11, Computer Science Dept., Iowa State University, 1998.

[LaValle 99] S.M. LaValle, J.H. Yakey & L.E. Kavraki. *A Probabilistic Roadmap Approach for Systems with Closed Kinematic Chains*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 1671–1676, 1999.

[LaValle 00] S.M. LaValle, P.W.Finn, L.E. Kavraki & J.-C. Latombe. *A Randomized Kinematics-Based Approach to Pharmacophore-Constrained Conformational Search and Database Screening.* Journal of Computational Chemistry, vol. 21(9), pages 731–747, 2000.

[LaValle 01a] S.M. LaValle & P. Konkimalla. *Algorithms for Computing Numerical Optimal Feedback Motion Strategies.* International Journal of Robotics Research, vol. 20(9), pages 729–752, 2001.

[LaValle 01b] S.M. LaValle & J.J. Kuffner. *Randomized Kinodynamic Planning.* International Journal of Robotics Research, vol. 20(5), pages 378–400, 2001.

[LaValle 01c] S.M. LaValle & J.J. Kuffner. *Rapidly-Exploring Random Trees: Progress and Prospects.* In B.R. Donald, K.M. Lynch & D. Rus, editors, Algorithmic and Computational Robotics: New Directions (WAFR2000), pages 293–308. A.K. Peters, 2001.

[LaValle 02] S.M. LaValle. *From Dynamic Programming to RRTs: Algorithmic Design of Feasible Trajectories.* In A. Bicchi, H.I. Christensen & D. Prattichizzo, editors, Control Problems in Robotics, pages 19–37. Springer-Verlag, 2002.

[LaValle 03a] S.M. LaValle. Planning algorithms. 1999-2003. Available at `http://msl.cs.uiuc.edu/planning/`.

[LaValle 03b] S.M. LaValle, M.S. Branicky & S.R. Lindemann. *On the Relationship Between Classical Grid Search and Probabilistic Roadmaps.* International Journal of Robotics Research, 2003. In press.

[Lazarevic 97] Z. Lazarevic. *Feasibility of a Stewart Platform with Fixed Actuators as a Platform for CABG Surgery Device.* Master's thesis, Columbia University, 1997.

[Leach 96] A.R. Leach. Molecular modeling: Principles and applications. Longman, 1996.

[Lee 88a] H.Y. Lee & C.G. Liang. *Displacement Analysis of the General Spatial 7-Link 7R Mechanism.* Mechanism and Machine Theory, vol. 23(3), pages 219–226, 1988.

[Lee 88b] H.Y. Lee & C.G. Liang. *A New Vector Theory for the Analysis of Spatial Mechanisms.* Mechanism and Machine Theory, vol. 23(3), pages 209–217, 1988.

[Lee 96] D.S. Lee & G.S Chirikjian. *A Combinatorial Approach to Trajectory Planning for Binary Manipulators.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 2749–2754, 1996.

[Lien 03] J.-M. Lien, S.L. Thomas & N.M. Amato. *A General Framework for Sampling on the Medial Axis of the Free Space*. Proc. IEEE Int. Conf. on Robotics and Automation, 2003. In press.

[Lin 03] M.C. Lin & D. Manocha. *Collision and Proximity Queries*. In Handbook of Discrete and Computational Geometry: Collision Detection. 2003. To appear. Available at `http://www.cs.unc.edu/~dm/`.

[Lindemann 03] S.R. Lindemann & S.M. LaValle. *Incremental Low-Discrepancy Lattice Methods for Motion Planning*. Proc. IEEE Int. Conf. on Robotics and Automation, 2003. In press.

[Lindemann 04] S.R. Lindemann & S.M. LaValle. *Steps Toward Derandomizing RRTs*. Preliminary draft, submitted to IEEE Fourth International Workshop on Robot Motion and Control, 2004.

[Lotan 02] I. Lotan, F. Schwarzer, D. Halperin & J.-C. Latombe. *Efficient Maintenance and Self-Collision Testing for Kinematic Chains*. Proc. 18th ACM Symp. on Computational Geometry, pages 43–52, 2002.

[Lozano-Perez 92] T. Lozano-Perez, J.L. Jones, E. Mazer & P.A. O'Donnell. Handey: A robot task planner. MIT Press, 1992.

[Lozano-Pérez 83] T. Lozano-Pérez. *Spatial Planning: A Configuration Space Approach*. IEEE Transactions on Computers, vol. 32(2), pages 108–120, 1983.

[Lück 93] C.L. Lück & S. Lee. *Self-Motion Topology for Redundant Manipulators with Joint Limits*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 626–631, 1993.

[Lynch 95] K. Lynch & M.T. Mason. *Stable Pushing: Mechanics, Controllability and Planning*. In K. Goldberg, D. Halperin, J.-C. Latombe & R. Wilsonet, editors, Algorithmic Foundations of Robotics (WAFR1994), pages 239–262. A.K. Peters, 1995.

[Ma 91] O. Ma & J. Angeles. *Architecture Singularities of Platform Manipulators*. Proc. IEEE Int. Conf. on Robotics and Automation, pages 1542–1547, 1991.

[Manocha 94] D. Manocha & J.F. Canny. *Efficient Inverse Kinematics of General 6R Manipulators*. IEEE Transactions on Robotics and Automation, vol. 10(5), pages 648–657, 1994.

[Manocha 95] D. Manocha, Y. Zhu & W. Wright. *Conformational Analysis of Molecular Chains Using Nano-Kinematics*. Computer Applications of Biological Sciences, vol. 11(1), pages 71–86, 1995.

[Masory 95] O. Masory & J. Wang. *Workspace Evaluation of Stewart Platforms*. Advanced Robotics Journal, vol. 9(4), pages 443–461, 1995.

[McCarthy 00] J.M. McCarthy. Geometric design of linkages. Springer-Verlag, 2000.

[Merlet 92] J.-P. Merlet. *Geometry and Kinematic Singularities of Closed-Loop Manipulators*. Journal of Laboratory Robotic and Automation, vol. 4, pages 85–96, 1992.

[Merlet 94] J.-P. Merlet. *Trajectory Verification in the Workspace for Parallel Manipulators*. International Journal of Robotics Research, vol. 13(4), pages 326–333, 1994.

[Merlet 95] J.-P. Merlet. *Determination of the Orientation Workspace of Parallel Manipulators*. Journal of Intelligent and Robotic Systems, vol. 13, pages 143–160, 1995.

[Merlet 98] J.-P. Merlet, C. Gosselin & N. Mouly. *Workspaces of Planar Parallel Manipulators*. Mechanism and Machine Theory, vol. 33(1/2), pages 7–20, 1998.

[Merlet 99] J.-P. Merlet. *Parallel Robots: Open Problems*. Proc. 9th Int. Symp. of Robotics Research, pages 27–32, 1999.

[Merlet 00] J.-P. Merlet. Parallel robots. Kluwer Academic Publishers, 2000.

[Merlet 01] J.-P. Merlet. *A Parser for the Interval Evaluation of Analytical Functions and its Applications to Engineering Problems*. Journal of Symbolic Computation, vol. 31, pages 475–486, 2001.

[Merlet 02] J.-P. Merlet. *Still a Long Way to Go on the Road for Parallel Mechanisms*. Proc. ASME 27th Biennial Mechanisms and Robotics Conf., 2002. Available at `http://www-sop.inria.fr/coprin/equipe/merlet/ASME/asme2002.html`.

[Metropolis 53] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller & E.Teller. *Equation of State Calculations by Fast Computing Machines*. Journal of Chemical Physics, vol. 21, pages 1087–1092, 1953.

[Milgram 02] R.J. Milgram & J.C. Trinkle. *The Geometry of Configuration Spaces for Closed Chains in Two and Three Dimensions*. Preliminary draft, 2002.

[Mishra 97] B. Mishra. *Computational Real Algebraic Geometry*. In J. E. Goodman & J. O'Rour, editors, Discrete and Computational Geometry, pages 537–556. CRC Press, 1997.

[Moore 79] R.E. Moore. Methods and applications of interval analysis. SIAM, 1979.

[Mortenson 97] M.E. Mortenson. Geometric modeling. John Wiley & Sons, 1997.

[Motwani 95] R. Motwani & P. Raghavan. Randomized algorithms. Cambridge University Press, 1995.

[Moult 86] J. Moult & M.N.G. James. *An Algorithm for Determining the Conformation of Polypeptide Segments in Proteins by Systematic Search*. Proteins: Structure, Function, and Genetics, vol. 1, pages 146–163, 1986.

[Neiderreiter 92] H. Neiderreiter. Random number generation and quasi-monte carlo methods. SIAM, 1992.

[Nielsen 97] J. Nielsen & B. Roth. *Formulation and Solution for the Direct and Inverse Kinematics Problem for Mechanisms and Mechatronic Systems*. Proc. NATO Advanced Study Institute on Computational Methods in Mechanisms, vol. 1, pages 233–252, 1997.

[Nielsen 00] C.L. Nielsen & L.E. Kavraki. *A Two Level Fuzzy PRM for Manipulation Planning*. Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pages 1716–1722, 2000.

[Nilsson 69] N.J. Nilsson. *A Mobile Automation: An Application of Artificial Intelligence Techniques*. Proc. 1st Int. Joint Conf. on Artificial Intelligence, pages 509–520, 1969.

[Nissoux 99] C. Nissoux. *Visibilité et Méthodes Probabilistes pour la Planification de Mouvement en Robotique*. PhD thesis, Université Paul Sabatier, 1999.

[Oliva 97] B. Oliva, P.A. Bates, E. Querol, F.X. Aviles & M.J.E. Sternberg. *An Automated Classification of the Structure of Protein Loops*. Journal of Molecular Biology, vol. 266(4), pages 814–830, 1997.

[Osborne 01] M.J. Osborne, J. Schnell, S.J. Benkovic, H.J. Dyson & P.E. Wright. *Backbone Dynamics in Dihydrofolate Reductase Complexes: Role of Loop Flexibility in the Catalytic Mechanism*. Biochemistry, vol. 40(33), pages 9846–9859, 2001.

[Overmars 95] M.H. Overmars & P. Svestka. *A Probabilistic Learning Approach to Motion Planning*. In K. Goldberg, D. Halperin, J.-C. Latombe & R. Wilsonet, editors, Algorithmic Foundations of Robotics (WAFR1994), pages 19–37. A.K. Peters, 1995.

[Pan 99] V.Y. Pan. *Solving a Polynomial Equation: Some History and Recent Progress*. SIAM Review, vol. 39(2), pages 187–220, 199.

[Parsons 94] D. Parsons & J. Canny. *Geometric Problems in Molecular Biology and Robotics*. Proc. 2nd Int. Conf. on Intelligent Systems for Molecular Biology, pages 322–330, 1994.

[Paul 81] R. Paul. Robot manipulators: Mathematics, programming and control. MIT Press, 1981.

[Pauling 60] L. Pauling. The nature of the chemical bond. Cornell University Press, 1960.

[Pettre 02] J. Pettre, T. Siméon & J.-P. Laumond. *Planning Human Walk in Virtual Environments*. Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pages 3048–3053, 2002.

[Pieper 68] D.L. Pieper. *The Kinematics of Manipulators under Computer Control*. PhD thesis, Stanford University, 1968.

[Porta 02] J.M. Porta, L. Ros, F. Thomas & C. Torras. *Solving Multi-Loop Linkages by Iterating 2D Clippings*. In A.J. Lenarcic & F. Thomas, editors, Advances in Robot Kinematics, pages 255–264. Kluwer Academic Publishers, 2002.

[Porta 03] J.M. Porta, L. Ros, F. Thomas & C. Torras. *A Branch-and-Prune Algorithm for Solving Systems of Distance Constraints*. Proc. IEEE Int. Conf. on Robotics and Automation, 2003. In press.

[Primrose 86] E.J.F. Primrose. *On the Input-Output Equation of the General 7R Mechanism*. Mechanism and Machine Theory, vol. 21(6), pages 509–510, 1986.

[Raghavan 89] M. Raghavan & B. Roth. *Kinematic Analysis of* 6*R Manipulator of General Geometry*. Proc. 5th Int. Symp. Robotics Research, pages 314–320, 1989.

[Rajan 99] K. Rajan & N. Deo. *A Parallel Algorithm for Bound-Smoothing*. Proc. 13th Int. Parallel Processing Symp. and 10th Symp. on Parallel and Distributed Processing, pages 645–652, 1999.

[Ramachandran 68] G.N. Ramachandran & V. Sasisekharan. *Conformation of Polypeptides and Proteins*. Adv. Prot. Chem, vol. 23, pages 283–438, 1968.

[Rao 98] R. S. Rao, A. Asaithambi & S. K. Agrawal. *Inverse Kinematic Solution of Robot Manipulators Using Interval Analysis*. ASME Journal of Mechanical Design, vol. 120, pages 147–150, 1998.

[Reif 79] J.H. Reif. *Complexity of the Mover's Problem and Generalizations*. Proc. 20th IEEE Symp. on Foundations of Computer Science, pages 421–427, 1979.

[Renaud 00] M. Renaud. *A Simplified Inverse Kinematic Model Calculation Method for all* $6R$ *type Manipulators.* Proc. Int. Conf. Mechanical Design and Production, pages 15–25, 2000.

[Ricard 98] R. Ricard & C.M. Gosselin. *On the Determination of the Workspace of Complex Planar Robotic Manipulators.* ASME Journal of Mechanical Design, vol. 120, pages 269–278, 1998.

[Roth 94] B. Roth. *Computational Advances in Robot Kinematics.* In A.J. Lenarcic & B.B. Ravani, editors, Advances in Robot Kinematics and Computational Geometry, pages 7–16. Kluwer Academic Publishers, 1994.

[Sahbani 02] A. Sahbani, J. Cortés & T. Siméon. *A Probabilistic Algorithm for Manipulation Planning under Continuous Grasps and Placements.* Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pages 1560–1565, 2002.

[Sahbani 03] A. Sahbani. *Planification de Tâches de Manipulation en Robotique par des Approches Probabilistes.* PhD thesis, Université Paul Sabatier, 2003.

[Schneider 92] S.A. Schneider & R.H. Cannon. *Object Impedance Control for Cooperative Manipulation: Theory and Experimental Results.* IEEE Transactions on Robotics and Automation, vol. 8(3), pages 383–394, 1992.

[Schwartz 83] J.T. Schwartz & M. Sharir. *On the Piano Movers' Problem II: General Techniques for Computing Topological Properties of Real Algebraic Manifolds.* Advances in Applied Mathematics, vol. 4, pages 298–351, 1983.

[Schwartz 84] J.T. Schwartz & M. Sharir. *On the Piano Movers' Problem V: The Case of a Rod Moving in Three-Dimensional Space Amidst Polyhedral Obstacles.* Communications on Pure Applied Mathematics, vol. 37, pages 815–848, 1984.

[Schwarzer 02] F. Schwarzer, M. Saha & J.-C. Latombe. *Exact Collision Checking of Robot Paths.* Proc. Workshop on the Algorithmic Foundations of Robotics, 2002. In press.

[Sciavicco 00] L. Sciavicco & B. Siciliano. Modelling and control of robot manipulators. Springer-Verlag, 2000.

[Scott 66] R.A. Scott & H.A. Scheraga. *Conformational Analysis of. Macromolecules. II. The Rotational Isomeric States of the Normal Hydrocarbons.* Journal of Chemical Physics, vol. 44, pages 3054–3069, 1966.

[Sekhavat 98] S. Sekhavat, P. Svestka J.-P. Laumond & M.H. Overmars. *Multi-Level Path Planning for Nonholonomic Robots using Semi-Holonomic Subsystems.* International Journal of Robotics Research, vol. 17(8), pages 840–857, 1998.

[Shenkin 87] P.S. Shenkin, D.L. Yarmush, R.M. Fine, H. Wang & C. Levinthal. *Predicting Antibody Hypervariable Loop Conformation. I. Ensembles of Random Conformations for Ringlike Structures.* Biopolymers, vol. 26, pages 2053–2085, 1987.

[Sherbrooke 93] E.C. Sherbrooke & N.M. Patrikalakis. *Computation of the Solutions of Nonlinear Polynomial Systems.* Computer Aided Geometric Design, vol. 10(5), pages 379–405, 1993.

[Siciliano 90] B. Siciliano. *Kinematic control of redundant robot manipulators: A tutorial.* Journal of Intelligent and Robotic Systems, vol. 3, pages 201–212, 1990.

[Siméon 00] T. Siméon, J.-P. Laumond & C. Nissoux. *Visibility-Based Probabilistic Roadmaps for Motion Planning.* Advanced Robotics Journal, vol. 14(6), pages 477–494, 2000.

[Siméon 01a] T. Siméon, J. Cortés, A. Sahbani & J.-P. Laumond. *A Manipulation Planner for Pick and Place Operations under Continuous Grasps and Placements.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 2022–2027, 2001.

[Siméon 01b] T. Siméon, J.-P. Laumond & F. Lamiraux. *Move3D: a Generic Platform for Path Planning.* Proc. IEEE Int. Symp. on Assembly and Task Planning, pages 25–30, 2001.

[Siméon 01c] T. Siméon, J.-P. Laumond, C. van Geem & J. Cortés. *Computer Aided Motion: Move3D within MOLOG.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 1494–1499, 2001.

[Siméon 03] T. Siméon, J.-P. Laumond, J. Cortés & A. Sahbani. *Manipulation Planning with Probabilistic Roadmaps.* International Journal of Robotics Research, 2003. Accepted with minor revisions.

[Skov 01] L.K. Skov, O. Mirza, A. Henriksen G.P. de Montalk, M. Remaud-Siméon, P. Sarçabal, R.M. Willemot, P. Monsan & M. Gajhede. *Amylosucrase, a Glucan-synthesizing Enzyme from the α-Amylase Family.* Journal of Biological Chemistry, vol. 276, pages 25273–25278, 2001.

[Skov 02] L.K. Skov, O. Mirza, D. Sprogoe, I. Dar, M. Remaud-Simeon, C. Albenne, P. Monsan & M. Gajhede. *Oligosaccharide and Sucrose Complexes of Amylosucrase. Structural Implications for the*

Polymerase Activity. Journal of Biological Chemistry, vol. 277, pages 47741–47747, 2002.

[Sánchez 02] G. Sánchez & J.-C. Latombe. *On Delaying Collision Checking in PRM Planning - Application to Multi-Robot Coordination.* International Journal of Robotics Research, vol. 21(1), pages 5–26, 2002.

[Sánchez 03] G. Sánchez & J.-C. Latombe. *A Single-Query Bi-Directional Probabilistic Roadmap Planner with Lazy Collision Checking.* In R. Jarvis & A. Zelinsky, editors, Robotics Research, the Tenth International Symposium, pages 403–417. Springer Tracts in Advanced Robotics, Springer, 2003.

[Stewart 65] D. Stewart. *A Platform with Six Degrees of Freedom.* Proc. of the Institution of Mechanical Engineers, vol. 180(15), pages 371–386, 1965.

[Svestka 97a] P. Svestka. *Robot Motion Planning using Probabilistic Roadmaps.* PhD thesis, Universiteit Utrecht, 1997.

[Svestka 97b] P. Svestka & M.H. Overmars. *Motion Planning for Car-like Robots, a Probabilistic Learning Approach.* International Journal of Robotics Research, vol. 16(2), pages 119–143, 1997.

[Svestka 98] P. Svestka & M.H. Overmars. *Probabilistic Path Planning.* In J.-P. Laumond, editor, Robot Motion Planning and Control, pages 255–304. Springer-Verlag, 1998.

[Talpaert 93] Y. Talpaert. Leçons et applications de géométrie différentielle et de mécanique analytique. Cepadues, 1993.

[Thomas 93] F. Thomas. *The Self-Motion Manifold of the N-bar Mechanism.* In J. Angeles, G. Hommel & P. Kovács, editors, Computational Kinematics, pages 95–107. Kluwer Academic Publishers, 1993.

[Tolani 00] D. Tolani, A. Goswami & N.I. Badler. *Real-Time Inverse Kinematics Techniques for Anthropomorphic Limbs.* Graphical Models, vol. 62(5), pages 353–388, 2000.

[Tombropoulos 99] R.Z. Tombropoulos, J.R. Adler & J.-C. Latombe. *CARABEAMER: A Treatment Planner for a Robotic Radio-Surgical System with General Kinematics.* Medical Image Analysis, vol. 3(3), pages 237–264, 1999.

[Tramontano 01] A. Tramontano, R. Leplae & V. Morea. *Analysis and Assessment of Comparative Modeling Predictions in CASP4.* Proteins: Structure, Function, and Genetics, vol. Suppl.5, pages 22–38, 2001.

[Trinkle 02] J.C. Trinkle & R.J. Milgram. *Complete Path Planning for Closed Kinematic Chains with Spherical Joints.* International Journal of Robotics Research, vol. 21(9), pages 773–789, 2002.

[van Geem 01a] C. van Geem & T. Siméon. *KCD: A Collision Detector for Path Planning in Factory Models.* Rapport LAAS N°01073, 2001.

[van Geem 01b] C. van Geem, T. Siméon & J. Cortés. *Progress Report on Collision Detection.* In Second Year Deliverables of the MOLOG Project. 2001.

[van Vlijmen 97] H.W.T. van Vlijmen & M. Karplus. *PDB-based Protein Loop Prediction: Parameters for Selection and Methods for Optimization.* Journal of Molecular Biology, vol. 267(4), pages 975–1001, 1997.

[Wampler 90] C.W. Wampler, A.P. Morgan & A.J. Sommese. *Numerical Continuation Methods for Solving Polynomial Systems Arising in Kinematics.* ASME Journal of Mechanical Design, vol. 112, pages 59–68, 1990.

[Wedemeyer 99] W.J. Wedemeyer & H.A. Scheraga. *Exact Analytical Loop Closure in Proteins Using Polynomial Equations.* Journal of Computational Chemistry, vol. 20(8), pages 819–844, 1999.

[Welman 93] C. Welman. *Inverse Kinematics and Geometric Constraints for Articulated Figure Manipulation.* Master's thesis, Simon Fraser Universit, 1993.

[Wenger 98] P. Wenger & D. Chablat. *Workspace and Assembly modes in Fully-Parallel Manipulators: A Descriptive Study.* 6th Int. Workshop on Advances in Robot Kinematics, pages 117–126, 1998.

[Wilfong 88] G. Wilfong. *Motion Planning in the Presence of Movable Obstacles.* Proc. 4th ACM Symp. on Computational Geometry, pages 279–288, 1988.

[Wilmarth 99] S. Wilmarth, N.M. Amato & P.Stiller. *MAPRM: A Probabilistic Roadmap Planner with Sampling on the Medial Axis of the Free Space.* Proc. IEEE Int. Conf. on Robotics and Automation, pages 1024–1031, 1999.

[Xiao 01] J. Xiao & X. Ji. *On Automatic Generation of High-level Contact State Space.* International Journal of Robotics Research, vol. 20(7), pages 584–606, 2001.

[Yakey 00] J.H. Yakey. *Randomized Path Planning for Linkages with Closed Kinematic Chains.* Master's thesis, Iowa State University, 2000.

[Yakey 01] J.H. Yakey, S.M. LaValle & L.E. Kavraki. *Randomized Path Planning for Linkages with Closed Kinematic Chains.* IEEE Transactions on Robotics and Automation, vol. 17(6), pages 951–958, 2001.

[Yim 03] M. Yim, K. Roufas, D. Duff, Y. Zhang, C. Eldershaw & S. Homans. *Modular Reconfigurable Robots in Space Applications.* Autonomous Robots Journal, vol. 14, pages 225–237, 2003.

[Yu 00] Y. Yu & K. Gupta. *Sensor-Based Probabilistic Roadmaps: Experiments with an Eye-in-Hand System.* Advanced Robotics Journal, vol. 14(6), pages 515–536, 2000.

[Zhang 02] M. Zhang & L.E. Kavraki. *A New Method for Fast and Accurate Derivation of Molecular Conformations.* Journal of Chemical Information and Computer Sciences, vol. 42(1), pages 64–70, 2002.

[Zheng 93] Q. Zheng, R. Rosenfeld, S. Vajda & C. DeLisi. *Loop Closure Via Bond Scaling and Relaxation.* Journal of Computational Chemistry, vol. 14, pages 556–565, 1993.

Résumé :

Un système robotique agit par le mouvement dans un monde physique. La capacité de planification de mouvement est donc une composante essentielle de l'autonomie du système et constitue un domaine de recherche très actif en Robotique. Le champ d'application de ces méthodes dépasse aujourd'hui le cadre de la Robotique et intéresse des domaines aussi diversifiés que la CAO, la logistique industrielle, l'animation graphique ou la biologie moléculaire. Dans tous ces domaines on est confronté au mouvement de systèmes complexes contenant des chaînes cinématiques fermées. Cette thèse traite de la planification de mouvement pour de tels systèmes.

La première partie présente notre contribution théorique et technique. Après avoir proposé une formulation générale de la planification de mouvement sous contrainte de fermeture cinématique, nous décrivons une méthode qui s'inscrit dans le cadre des techniques d'exploration par échantillonnage. Les outils algorithmiques que nous proposons permettent une application efficace de ces techniques à des systèmes mécaniques complexes.

La deuxième partie traite de l'utilisation de ces outils pour la résolution de divers problèmes. En Robotique, nos algorithmes ont été appliqués à la synthèse de mouvement de mécanismes parallèles, à la manipulation coordonnée ainsi qu'à la planification de tâches de manipulation d'objets. Enfin, nous abordons une application originale à la biologie structurale pour l'étude des capacités de mobilité de boucles protéiques. Les résultats obtenus à travers ces applications montrent l'efficacité et la généralité de notre approche.

Summary :

A robot essentially acts by moving in a physical world. The motion planning capability is thus a fundamental issue for the autonomy of the system, and represents a very active area of research in Robotics. Furthermore, the interest in motion planning techniques goes beyond robotic applications. Currently, these techniques are applied in other very different domains such as: CAD/CAM, industrial logistics, graphic animation, or computational Biology. Complex articulated mechanisms containing closed kinematic chains appear in all these domains. This thesis treats motion planning for such systems.

The first part contains our theoretical and technical work. We introduce sampling-based planners into a general formulation of the motion planning problem in presence of kinematic closure constraints. The algorithmic tools that we propose allow an efficient application of these techniques to complex closed-chain mechanisms.

The second part deals with the different fields of application that we have investigated. In Robotics, our approach has been applied to motion synthesis of parallel robots, coordinated manipulation and manipulation planning. Finally, we discuss an original application to Structural Biology for the conformational analysis of protein loops. The results of our experiments prove the efficacy and the generality of the approach.