(Dis)assembly path planning for complex objects and applications to structural biology.

Duc Thanh Le

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National Polytechnique de Toulouse (INP Toulouse)

Discipline ou spécialité :

Image, Information, Hypermédia

Présentée et soutenue par :
DUC THANH LE

le : 28/09/2010

Titre :
Algorithmes pour le (dés)assemblage d'objets complexes
et applications à la biologie structurale
(Dis)assembly path planning for complex objects
and applications to structural biology

JURY
Frédéric CAZALS INRIA, Sophia Antipolis Rapporteur
Philippe DERREUMAUX IPBC, Paris Rapporteur
Thierry SIMEON LAAS-CNRS, Toulouse Directeur de thèse
Juan CORTES LAAS-CNRS, Toulouse Co-Directeur de thèse
Rachid ALAMI LAAS-CNRS, Toulouse Membre Invité

Ecole doctorale :
Mathématiques Informatique Télécommunications (MITT)

Unité de recherche :
Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS-CNRS)

Directeur(s) de Thèse :
Thierry SIMEON
Juan CORTES

Rapporteurs :
Frédéric CAZALS
Philippe DERREUMAUX

UNIVERSITÉ TOULOUSE III - PAUL SABATIER
ÉCOLE DOCTORALE MATHÉMATIQUES, INFORMATIQUE, TÉLÉCOMMUNICATIONS DE TOULOUSE

THÈSE

en vue de l'obtention du

Doctorat de l'Université de Toulouse
délivré par l'Institut Nationale Polytechnique de Toulouse

Spécialité: Image, Information et Hypermédia

(Dis)assembly path planning for complex objects and applications to structural biology

DUC THANH LE

Préparée au Laboratoire d'Analyse et d'Architecture des Systèmes
sous la direction de M. Thierry SIMÉON et M. Juan CORTÉS

Jury

Frédéric CAZALS	INRIA, Sophia Antipolis	Rapporteur
Philippe DERREUMAUX	IBPC, Paris	Rapporteur
Rachid ALAMI	LAAS-CNRS, Toulouse	Membre Invité
Isabelle ANDRÉ	LISBP, Toulouse	Membre Invité
Thierry SIMÉON	LAAS-CNRS, Toulouse	Directeur de Thèse
Juan CORTÉS	LAAS-CNRS, Toulouse	Co-directeur de Thèse

To my parents!

Acknowledgements

During three years of my thesis work, there were many people who were quite willing to give me a help. Without their guidance, help, and patience, I would have never been able to accomplish my thesis. I would like to gratefully acknowledge all of them who have accompanied me along this important period.

Above all, I would like to thank my two great supervisors Dr. Thierry SIMÉON and Dr. Juan CORTÉS, whose important guidance and continuing encouragements have allowed me to make much progress in my studies over the multidisciplinary domains.

My greetings go also to Dr. Frédéric CAZALS and Prof. Philippe DERREUMAUX who honored me by accepting to be reviewers of my thesis. I also thank Dr. Rachid ALAMI and Dr. Isabelle ANDRÉ for taking part of the jury. All their comments and suggestions are very useful.

During three years of working in the LAAS-CNRS, I have had chance to collaborate with many nice colleagues, particularly: Léonard JAILLET, Mokhtar GHARBI, Romain IEHL, Yi LI, Isabell ANDRÉ, and Sophie BARBE. Without their help, it would be much more difficult for me to finish this thesis.

Finally, I would like to mention all the encouragement of my family, my girlfriend Bao Ngoc and my friends, that gave me the great motivation to make my dream come true.

Abstract

Understanding and predicting structure-function relationships in proteins with fully *in silico* approaches remain today a great challenge. Despite recent developments of computational methods for studying molecular motions and interactions, dealing with macromolecular flexibility largely remains out of reach of the existing molecular modeling tools. The aim of this thesis is to develop a novel approach based on motion planning algorithms originating from robotics to better deal with macromolecular flexibility in protein interaction studies.

We have extended a recent sampling-based algorithm, ML-RRT, for (dis)-assembly path planning of complex articulated objects. This algorithm is based on a partition of the configuration parameters into *active* and *passive* subsets, which are then treated in a decoupled manner. The presented extensions permit to consider different levels of mobility for the passive parts that can be pushed or pulled by the motion of active parts. This algorithmic tool is successfully applied to study protein conformational changes induced by the diffusion of a ligand inside it.

Building on the extension of ML-RRT, we have developed a novel method for simultaneously (dis)assembly sequencing and path planning. The new method, called Iterative-ML-RRT, computes not only the paths for extracting all the parts from a complex assembled object, but also the preferred order that the disassembly process has to follow. We have applied this general approach for studying disassembly pathways of macromolecular complexes considering a scoring function based on the interaction energy. The results described in this thesis prove not only the efficacy but also the generality of the proposed algorithms.

Keywords : Motion Planning, Disassembly Sequencing, Protein-Ligand Interactions, Protein Complex Disassembly.

Résumé

La compréhension et la prédiction des relations structure-fonction de protéines par des approches *in sillico* représentent aujourd'hui un challenge. Malgré le développement récent de méthodes algorithmiques pour l'étude du mouvement et des interactions moléculaires, la flexibilité de macromolécules reste largement hors de portée des outils actuels de modélisation moléculaire. L'objectif de cette thèse est de développer une nouvelle approche basée sur des algorithmes de planification de mouvement issus de la robotique pour mieux traiter la flexibilité moléculaire dans l'étude des interactions protéiques.

Nous avons étendu un algorithme récent d'exploration par échantillonnage aléatoire, ML-RRT pour le désassemblage d'objets articulés complexes. Cet algorithme repose sur la décomposition des paramètres de configuration en deux sous-ensembles actifs et passifs, qui sont traités de manière découplée. Les extensions proposées permettent de considérer plusieurs degrés de mobilité pour la partie passive, qui peut être poussée ou attirée par la partie active. Cet outil algorithmique a été appliqué avec succès pour l'étude des changements conformationnels de protéines induits lors de la diffusion d'un ligand.

A partir de cette extension, nous avons développé une nouvelle méthode pour la résolution simultanée du séquençage et des mouvements de désassemblage entre plusieurs objets. La méthode, nommée Iterative-ML-RRT, calcule non seulement les trajectoires permettant d'extraire toutes les pièces d'un objet complexe assemblé, mais également l'ordre permettant le désassemblage. L'approche est générale et a été appliquée pour l'étude du processus de dissociation de complexes macromoléculaires en introduisant une fonction d'évaluation basée sur l'énergie d'interaction. Les résultats présentés dans cette thèse montrent non seulement l'efficacité mais aussi la généralité des algorithmes proposés.

Mots-clefs : Planification de Mouvement, Séquençage de Désassemblage, Interactions Protéine-Ligand, Désassemblage des Complexes Protéiques.

Contents

1 Introduction

This thesis deals with atomic-scale modeling of protein motions and deformations that are necessary for their interactions with other molecules. Proteins are essential biological macromolecules that participate in most of the processes within cells. For instance, some types of proteins catalyze biochemical reactions, other proteins play structural roles. They are also involved in cell signaling processes, and participate to the cell-division cycle. The function of proteins is closely related with their structure and with their ability to undergo conformational changes. A better understanding of this relationship at the atomic level will lead to major scientific advances, with important applications in medicine, pharmacology and biotechnology. However, available tools to provide structural information on how proteins interact with other molecules are very limited. Experimental methods such as X-ray crystallography or nuclear magnetic resonance (NMR) are able to obtain an atomic-resolution representation of proteins, but provide null or poor information about how proteins deform to achieve their function. The limits of experimental methods are even more evident when the interactions to be analyzed involve large-amplitude (slow-timescale) conformational changes of proteins. Computational methods are therefore necessary to complement experimentation.

Computational methods in structural biology have been developed since several decades ago. Great advances on the understanding of protein function have been achieved thanks to molecular simulations. However, these methods, that mainly rely on Molecular Dynamics (MD) simulations, require huge computational resources for accessing to the time-scales of many protein interaction processes.

In this thesis, we have studied new computational methods, which can be considered as alternative or complementary tools, to address problems in structural biology involving large-amplitude protein motions. The main goal is to simulate such motions with low computational cost. For this, a geometric model of molecules is considered, where proteins are represented as articulated mechanisms. Then the problem of simulating relative motions between interacting molecules is formulated as a (dis)assembly problem for the

articulated objects. Inspired by recent research in robotics, we propose motion-planning-based algorithms to efficiently solve these complex (dis)assembly problems, which may involve hundreds of degrees of freedom.

This thesis is addressed both to researchers in structural biology, who would be interesting in applying or combining the methods we propose, and to researchers in robot motion planning, who could develop new methods inspired from ours. Therefore, Chapter 2 introduces some basic notions in structural biology that are straightforward for readers from the former group, but that will facilitate the reading of some chapters to "roboticians". Next, Chapter 3 presents a state-of-the-art on molecular modeling methods and motion planning algorithms, mainly focusing on their application to the simulation of protein motions and interactions. The next four chapters present the contribution of this thesis. This contribution is two-fold. First, we propose general methods that aim to be applicable to any type of mobile system. And second, we investigate the particular application to problems in structural biology. Chapters 4 and 6 involve the methodological contribution, while Chapters 5 and 7 deal with the applications. More precisely, Chapter 4 deals with disassembly path planning for two objects with articulated parts. A method is proposed that extends previous work to handle very complex systems. Chapter 5 investigates the application of this method to compute protein motions induced (or required) by the diffusion of a ligand inside it. Results presented in this chapter show the potential interest of the method. Based on an extension of the method described in Chapter 4, Chapter 6 presents a novel method for simultaneously (dis)assembly sequencing and path planning. The method is able to to solve general (dis)assembly planning problems involving objects with arbitrary shapes, and possibly requiring non-monotonic (dis)assembly sequences. Next, the application of this method to protein complexes involving multiple sub-units is presented in Chapter 7. Preliminary results are provided as a proof of concept. Nevertheless, the implementation needs to be improved in order to treat more accurately the physicochemical characteristics of protein complex dissociation. Finally, Chapter 8 concludes and discusses possible directions for future research.

2

Basics Notions in Structural Biology

Structural biology not only describes the structure at the atomic scale of living molecules, but also tries to create links between their structures and their biological functions by using physical, chemical and computational techniques such as X-ray crystallography, Nuclear Magnetic Resonance (NMR), Cryo-electron microscopy, and molecular modeling. This chapter gives some general notions for understanding molecular structures, conformational changes, and molecular interactions.

2.1 Protein Structure

Proteins are biological macromolecules consisting of chains of amino acids linked by peptide bonds. They are essential parts of organisms and participate in every process within cells such as gene expression, catalysis, storage, transport, signal transmission, etc. There are four organizational levels of protein structures:

- Primary structure: the sequence of amino acids, also called polypeptide chain.
- Secondary structure: some portions of the polypeptide chain that are arranged into regular structures called α-helices and β-sheets as a result of short distance interactions (i.e., hydrogen bond).
- Tertiary structure: the folded form of the polypeptide chain. This structure is a result of packing under a determined topology of the secondary structures.
- Quaternary structure: an association of several polypeptide chains (identical or not).

These notions are further explained next.

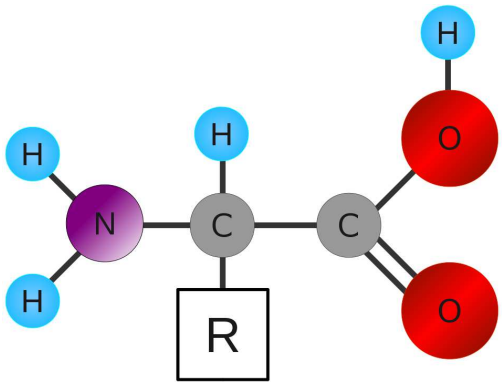


Figure 2.1: General amino acid structure: **R** represents a specific side-chain to each amino acid.

Amino Acid

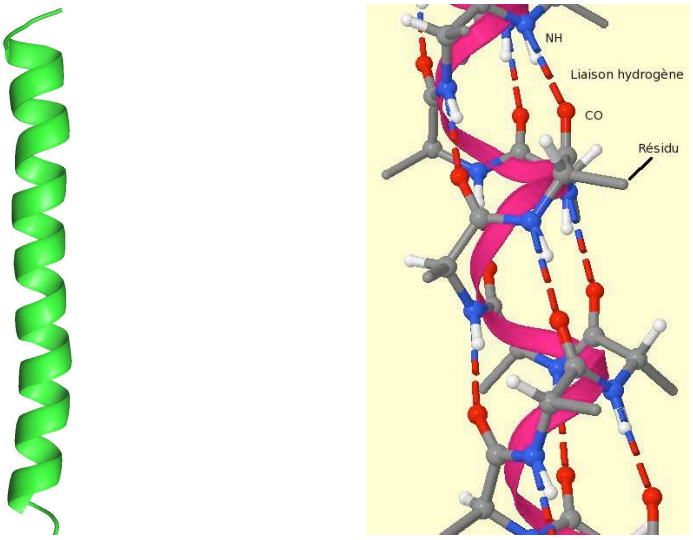
An amino acid is a molecule consisting of an asymmetric carbon (called α-carbon) attached to a carboxyl group (-COOH), an amine group (-NH2), a hydrogen H and a radical R also called side-chain (see Fig. 2.1). There are twenty natural amino acids in proteins. Each amino acid is given a name according to the nature of its radical. Each side-chain has some particular physico-chemical properties such as hydrophobicity, polarity, acidity, flexibility, steric congestion, etc.

Primary Structure

The primary structure is a sequence of amino acids starting from the N-terminal end (NH2-group) to the C-terminal end (-COOH group). This structure of a protein is determined by the gene associated with the protein. Two successive amino acids are connected by a peptide bond. This connection makes a spatial constraint: the C and the O atoms of carboxyl group of the first amino acid create a plane with the N and the Cα of the next amino acid. A protein includes between 30 and 30000 amino acids. The chain of the N, Cα, C, and O of all amino acids is called protein backbone. The amino-acid chain is also called polypeptide chain.

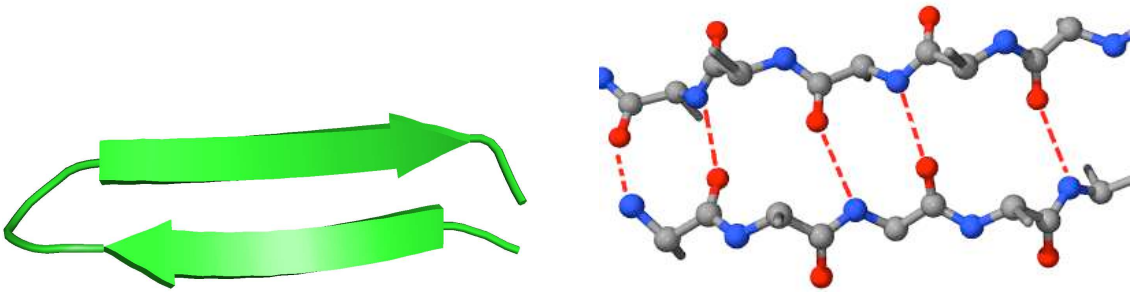
Secondary Structure

The secondary structure corresponds to a regular arrangement of amino acids along an axis. These structures are stabilized by the interactions of hydrogen bond network between non neighbor amino acids of the polypeptide chain. There are two main types of secondary structures: α-helices and β-sheets. These two structures minimize not only the steric hindrances and electrostatic repulsion between chains, but also maximize the number of hydrogen bond. Generally, the secondary structures contains approximately half of all amino acids while the other half is located in the loops that link secondary structure elements. The α-helix conformation is stabilized by hydrogen bonds of its amino acids. This kind of structure is stable and isolated. The hydrogen bonds guarantee the cohesion between amino acids of the sequence (see Fig. 2.2). On the contrary, the β-sheet usually contains two strands that form a sheet. The two strands can have the same direction or can be positioned in opposed directions (see Fig. 2.3). Many studies have been done to



(a) (b)

Figure 2.2: α-helix structure: (a) Cartoon Representation (b) Atomic Representation



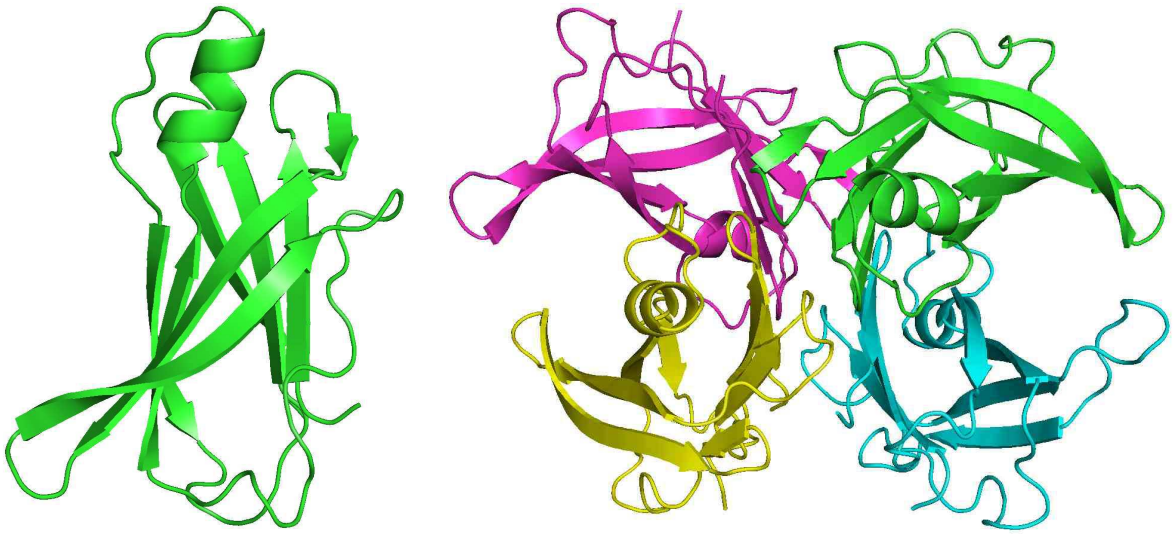
(a) (b)

Figure 2.3: β-sheet structure: (a) Cartoon Representation (b) Atomic Representation

predict secondary structures [Chiang et al. 2007; Moll et al. 2007]. This prediction may give some hints about the nature of folding, helps to localize residues of the active site, and may even give a hint about the protein localization in the cell (e.g., membrane proteins).

Tertiary Structure

The tertiary structure is the description of the polypeptide chain folded in its functional form (see Fig. 2.4.a). This form may involve covalent bonds (disulfide bridges), ions, or more complex co-factors (such as heme, flavin adenine dinucleotide, etc.). Tertiary structures can be complex and different. Long polypeptide chains (i.e., more than 200 amino acids) generally fold into several functional regions called domains. The protein folding is based primarily on long-range interactions taking place between two residues that are close in space but far in the sequence [Gromiha and Selvaraj 2002].



(a) (b)

Figure 2.4: Structure of human transthyretin complex: (a) Tertiary Structure - One subunit, (b) Quaternary Structure - Four subunits

Quaternary Structure

The quaternary structure is a geometry of the association of several protein sub-units (see Fig. 2.4.b). These sub-units can be either identical or not. Their association is stabilized by short distance interactions like in the tertiary structure. Complexes of two or more polypeptides (i.e., multiple subunits) are called multimers.

2.2 Molecular Forces

Due to the nature and structure of a biological molecules, the type and strength of molecular forces can be very different. The main forces that maintain the stability of protein structure and that stabilize protein-ligand, or protein-protein interactions are detailed next.

- **Covalent bonds** are a type of chemical bond, which are formed by sharing electrons between atoms. This force hold atoms together. Disulfide bridges, that occur when two Cysteines are oxidized to form covalent S–S bonds, are a particular type of covalent bond between atoms in non-consecutive residues.
- **Electrostatic forces** involve permanent dipoles interactions and salt bridges. Permanent dipoles occur when two atoms in a molecule have substantially different electro-negativity. One atom attracts electrons more than another, becoming more negative, while the other atom becomes more positive. Salt bridges correspond to the interactions between charged regions determined by electrostatic laws. In normal physiologic pH condition, positive charged residues are Lysines, Arginines, Histidines, and N-terminal end of the polypeptide chain, while negative charged residues are Aspartic acid, Glutamic acid.

- **Van der Waals forces** involve attractions between atoms, molecules, and surfaces. Differ from covalent and ionic bonding, these forces contain a long-range attractive component named London dispersion force and a very short-range repulsive component. The London forces arise from the interactive forces between temporary multipoles in molecules without permanent multipole moments due to the fluctuations of electron density. Since the energy bonding between the atoms that have 3-4 $\AA$ distance is only 1 kcal/mol, but with a huge number of atoms, these interactions play an important role in complex stabilization.
- **Hydrogen bonds** result from the electrostatic interaction between one hydrogen atom (H) covalenty linked with electronegative atom (O, N, S - donor) and other electronegative atom that has a pair of non-shared electrons (acceptor). This bonding energy can be estimated between 3 and 9 kcal/mol. Some polar amino acids such as Serine, Threonine, Tyrosine, Asparagine and Glutamine can also form hydrogen bonds between them or with water molecules.

Besides the aforementioned forces, the **Hydrophobic interactions**, that are of entropic nature , occur when non-polar molecules form aggregates of molecules in water and analogous intramolecular interactions [Chandler 2005]. These interactions play an important role for protein folding, protein-protein interactions, formation of lipid bilayer membranes, nucleic acid structures, and protein-small molecule interactions. Some hydrophobic amino acids such as Alanine, Valine, Leucine, Isoleucine, Proline, Phenylalanine, Tryptophan and Methionine possess non-charged and non-polar side chains. Like Van der Waals interactions, hydrophobic effects are responsible for dense compression in macromolecular assembly interfaces.

2.3 Protein Motions

Protein flexibility is crucial in a majority of cellular mechanisms. Proteins can deform to adapt to their partner to avoid steric hindrances and improve surface complementary for hydrogen bonding. Proteins can also change their conformation due to enzymatic reactions. Protein motions can be characterized by their amplitude. Weak-amplitude motions reflect the equilibrium inside an ensemble of conformers around its most stable state in the energy funnel. This motion type can be modeled by molecular dynamics simulations or observed by NMR experiments. Large-amplitude motions, in which two or more parts of protein move in relation to others, usually occur during the catalytic or allosteric reactions [Kumar et al. 1999]. Large-amplitude motions can not be observed with current experimental methods, and their simulation remains very computationally expensive. Protein motions can also be classified depending on the size of the involved protein portion: side-chain motions, loop motions, domain motions, and folding motions. These four types of motions are detailed next.

Side-chain Motions

Side-chains involve only a few atoms but may play an essential role in protein functions. In particular, in ligand-protein binding problem, even small magnitudes of side-chain rotations can lead to the binding of small organic molecules in protein pockets [Zavodszky and Kuhn 2004].

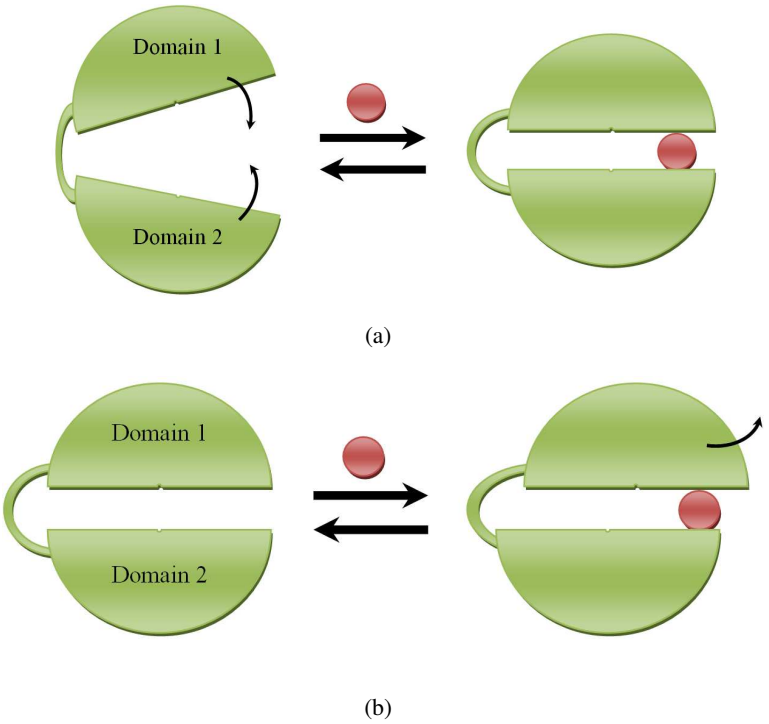


Figure 2.5: Common mechanisms for domain motions [Teague 2003]: (a) Hinge bending motion, (b) Shear bending motion.

Loop Motions

Protein surface loops containing about 6 to 20 residues and they are the most flexible regions of proteins [Fetrow 1995]. They are often involved in protein function and molecular recognition[Fetrow 1995]. Loops may play a key role in enzymatic catalysis, protein stability, or protein folding. Loop motions can be also observed during protein-protein or protein-DNA interactions.

Domain Motions

Domain motions are of two basic types: hinge motions and shear motions, [Gerstein et al. 1994; Teague 2003]. Hinge motions occur when two domains move perpendicularly to the interface surface (see Fig. 2.5.a). Shear motions occur when two domains move parallel to the interface surface(see Fig. 2.5.b). Domain motions are usually induced by ligand binding.

Folding/Unfolding Motions

Protein folding is a process by which a sequence of amino acids folds into its characteristic and functional three-dimensional structure, known as the native state. Once the protein has its correct structure, it can play its physicochemical function. Otherwise, in case of misfolding, extracellular or intracellular aggregates

that initiate profound cellular dysfunction can be formed. There are several diseases (e.g., Alzheimer and Parkinson) which are caused from these disorders of folding process [Selkoe 2003]. Proteins can be also unfolded (or denatured) by high-temperature, acidic or basic pH or certain non-aqueous solvents [Leach 2001b]. This unfolding is often reversible and proteins can be therefore refolded to its native state [Finkelstein 1997].

2.4 Protein Interactions

Proteins play a key role in molecular recognition at the heart of all processes of life. They interact with the other components of the cell (e.g., small molecules, nucleic acids, membranes, and other proteins) to build higher-order assemblies or to perform their functions (e.g., chemical catalysis, signaling, regulation, etc.). This thesis deals with protein-ligand and protein-protein interactions.

Protein-Ligand Interactions

A ligand can be an atom, an ion or a small molecule, that is able to bind to a protein to serve a biological functionality. Ligand can be substrate, inhibitor, activator, or neurotransmitter. A substrate that binds to the enzyme's active site, is transformed into products by an enzymatic mechanism. An inhibitor/activator binds to a protein to reduce/increase its activity.

Protein-Protein Interactions

Classes of protein-protein interactions can be defined according to their lifetime and binding affinity: obligate permanent interactions involving homo or hetero obligomers and non-obligate transient interactions involving Enzyme-inhibitor or non Enzyme-inhibitor. The permanent complexes (e.g., oligomeric complexes, virus capsids and muscular fibers) have a lifetime of about 10^6 seconds. On the contrary, the transient complexes associate and dissociate rapidly. Their lifetime is between 10^{-3} s and 1 s, such as the complexes in the cellular adhesion and in the signal transduction. There are also antibodies/antigens or enzymes/inhibitors complexes that are more stable ($\sim 10^3$ s). The reviews of [Nooren and Thornton 2003; Park et al. 2009] give a detailed description of different protein-protein interactions.

3

State of The Art

Computational methods are more and more important to complement experimental studies in structural biology. Although great progress has been done, the capacities of these methods are still limited. Probably the major challenge to be faced is the treatment of protein flexibility, which needs to be considered for an accurate study of molecular interactions. In the last years, new methods inspired by algorithms originally developed for robot motion planning have been proposed as alternative or complementary computational tools to treat problems involving flexibility and interactions of biological macromolecules.

This chapter provides a brief state-of-the-art on methods related with the thesis. First, Section 3.1 describes basic models and computational methods to study proteins. Section 3.2 gives an overview on motion planning algorithms, with a particular focus on sampling-based methods. Several applications of these robotic methods to probelms in structural biology are then presented in Section 3.3. Finally, the contribution of this thesis is outlined in Section 3.4.

3.1 Molecular Interaction Studies

This section gives a broad overview of models and basic methods used to simulate biological systems such as proteins, focusing on the most related techniques and problems with respect to the contribution of this thesis.

3.1.1 Models and Methods

This sub-section summarizes several models and methods which have been developed for decades to treat molecular simulation problems.

3.1.1.1 Models

Quantum mechanics (QM) uses mathematical tools to describe interactions between energy and matter at the atomic and subatomic scales (i.e., nuclei and electrons). This model allows a large variety of computational approaches to treat chemical structure and reactivity problems with great accuracy. However, due to its computational cost, approaches based on QM have only dealt with small upto medium-size molecules [Bakowies and Thiel 1996].

Molecular mechanics (MM) or Force field-based models consider atoms as spheres of different mass depending on the element, and connected together by chemical bonds. In order to describe the energy of the molecular system, different types of interactions (i.e., bond stretching, angle bending, dihedral rotation, van der Waals forces, hydrogen bonding, and electrostatic terms) are taken into account. The greatest advantage of MM over QM is a better computational speed in studying large molecular systems [Burkert and Allinger 1982]. Besides, since MM does not deal directly with electrons and orbitals, it does not allow to study chemical reactions or molecular reactivities [Bakowies and Thiel 1996].

Face to the limits of both quantum and molecular mechanics, a hybrid approach **QM/MM** has been proposed to combine their advantages. The basic idea of this model is to partition the large system into an electronically important fragment that is treated by quantum mechanics approach and a remainder that is described by a force field. This partitioning into two subsystems allows a complex system to be modeled in an accurate and affordable manner. With current improvements in computer hardware, the QM/MM approach has been applied to numerous biochemical problems [Bakowies and Thiel 1996; Sherwood 2000; Thiel 2009]. This model is suitable not only for simulating chemical reactions in the active site of a large system, but also for studying other localized electronic processes. Although QM/MM geometry optimization method can deal with large systems with 20000-30000 atoms, the active site size has still to limit around 1000 atoms to better investigate localized electronic event while the remainder must be fixed at the initial geometry [Thiel 2009].

In order to investigate large systems' interactions (e.g., protein-protein docking, rearrangement upon ligand binding, folding, etc), all above models that use an all-atom representation can not be directly applied with current computational resources. Therefore, coarse-grained models have been proposed to reduce the number of particles of the system [Tozzini 2005]. The main idea is to simplify the simulated systems by clustering several sub-components into one component in order to reduce the computational complexity by removing both degrees of freedom (DOF) and interactions inside the "coarse" component [Zacharias 2003; Derreumaux and Mousseau 2007; Marrink et al. 2007; Loriot et al. 2009]. Despite the simplification, physically correct properties of the biological system can be still conserved. Such a representation can speed up the computational time 1-2 orders of magnitude over the all-atom simulations [Freddolino et al. 2009].

3.1.1.2 Methods

Two main classes of methods for simulating biological macromolecular systems are Molecular Dynamics and Monte Carlo algorithms. **Molecular dynamics** simulations (MD) have been developed to investigate the physical basis of the structure and the function of biological macromolecules [Karplus and McCammon 2002]. These methods try to provide detailed information concerning the motion of each particle in the

system as a function of time. Therefore, dynamical properties of the biological systems such as transport coefficient, time-dependent responses to perturbations, rheological properties, and spectra can be addressed [Allen 2004]. Three types of applications of this technique can be classified in the macromolecular area [Karplus and McCammon 2002]. The first uses simulation to determine or refine data on the system's structure. The second tries to describe the system at equilibrium while considering the structural and motional properties along with thermodynamic parameters. The third aims to examine the actual dynamics to correctly represent the development of the system over time. Many tools based on MD are widely used such as CHARMM and AMBER [Weiner and Kollman 1981; Brooks et al. 1983; Brooks et al. 2009].

Monte Carlo methods (MC), which rely on statistical mechanics, have been widely used to deal with conformational search problems. Instead of evaluating forces to determine incremental atomic motions like MD methods, Monte Carlo simulations consider only large motions of the system and determines whether or not a sampled conformation is energetically feasible at the simulated temperature. Because MC simulations sample only in the conformation space, they can not provide time-dependent quantities. However, they are more efficient than MD methods in estimating average thermodynamic properties with many sampled conformations [Leach 2001a; Frenkel 2004].

Such two above methods present however a major drawback for computing large-amplitude motions since the conformational exploration tends to get trapped into the many local minima of the complex molecular energy landscape. Other classes of methods such as Normal Mode Analysis and Genetic Algorithms have been developed to treat particular problems. **Normal Mode Analysis** (NMA), which is based on an harmonic approximation of the molecular force field, is an efficient method to determine collective, large-amplitude motions of macromolecules [Cui and Bahar 2006; Ma 2005]. **Genetic Algorithms** (GA), which are stochastic optimization methods, try to mimic natural evolutionary method of adapting to a changing environment [Devillers 1975]. They provide a powerful random search tools in a large problem space. A wide range of applications which gain advantage from the use of GAs have been proposed in ligand-protein docking, drug design, or the structural alignment of molecules, etc. [Terfloth and Gasteiger 2001; Jones et al. 1997; Devillers 1996; Morris et al. 1998]

3.1.2 Protein Motions

Protein motions that range from high-frequency vibration to large-scale conformational changes play an essential role in many biochemical processes. Despite the new knowledge of structural and functional information, the understanding of protein movement is still very limited. Many techniques have been presented for modeling protein flexibility and mobility. They can be classified into three classes: force field-based, graph-based, and harmonic analysis-based methods [Ahmed et al. 2007].

In the first class, MD simulations are often used to model macromolecular movements by exploring the energy landscape of the macromolecule with an empirical force field. The trajectories of atoms are calculated by using Newton's Second Law and potential energy functions of atom–atom interactions [Leach 2001a; Schlick 2002; Hornak et al. 2006; Hornak and Simmerling 2007]. However, due to the computational cost, MD with all-atom models prohibits routine simulations of large-amplitude motions of macromolecules. The use of coarse-grained models together with efficient MD methods permits longer simulations with modest computational resources [Chebaro et al. 2009].

Monte Carlo (MC) algorithms have also been used to investigate protein flexibility. In particular, MC-based methods have been proposed for the exploring conformational energy surface of polypeptides. For example, a method called ARTIST [Yun et al. 2006] applies an effective activation-relaxation MC algorithm on internal-coordinates molecular models for exploring both small and large conformational changes in densely packed environments and for finding the global energy minima of medium-size proteins. A different approach to explore conformations of proteins is the building-up procedure that consists in performing MC moves on a discrete set of conformations of short polypeptides fragments that are encoded in fragment libraries computed from available data in the Protein Data Bank (PDB). This type of methods (e.g. [Simons et al. 1997; Hegler et al. 2009; Zhao et al. 2010]), initially proposed for protein structure prediction, can also be applied to represent conformational ensembles.

Alternative methods to determine protein flexibility are based on graph theory. These techniques allow to identify rigid and flexible regions of proteins from a single, static structure. Atoms and inter-atomic interactions are represented in a bond-bending network by modeling atoms as nodes and covalent bonds and non-covalent interactions as edges. A fast combinatorial algorithm then identifies the rigid, over-rigid, and flexible regions by counting bond-rotational degrees of freedom in the network [Ahmed et al. 2007]. FIRST is a popular program that implemented this technique to capture the essential conformational flexibility of the protein main-chain and side-chains from analysis of a single, static three-dimensional structure [Jacobs et al. 2001; Gohlke et al. 2004].

Large-amplitude molecular motions have also been studied by using normal mode analysis (NMA). A number of works used NMA to compute global macromolecular motions such as open-closed conformational transitions in proteins or domain motions [Brooks and Karplus 1985; Mouawad and Perahia 1993; Tama and Sanejouand 2001; Miyashita et al. 2003; Alexandrov et al. 2005; Kirillova et al. 2008]. Large-amplitude motions in macromolecules are shown to be associated with low-frequency normal modes, therefore demonstrating the ability of NMA-based methods to predict the direction of collective conformational changes.

3.1.3 Protein Ligand Interactions

In recent years, with the advantage of computing infrastructures, the development of *in silico* methods has been widely developed to better understanding protein-ligand interactions. Two types of problems are mostly analyzed: protein-ligand docking and access/exit pathways.

3.1.3.1 Protein-Ligand Docking

Powerful docking algorithms have been developed in the last ten years to efficiently explore and evaluate the huge number of possible ligand geometries for partner proteins, in terms of relative orientation and position. The problem increases in complexity when internal DOF also need to be explored, in which case the use of previously developed search algorithms leads to computational explosion. For detailed description of existing protein-ligand docking methods, a number of excellent review articles have been presented [Sousa et al. 2006; Warren et al. 2008; B-Rao et al. 2009; Henzler and Rarey 2010]. Overall, methods for protein-ligand docking are mainly built from three ingredients [Sousa et al. 2006]: (i) an adequate representation of

the proteins to be docked, which includes the definition of the DOF to be searched (global translations and rotations, side-chain rotations,...); (ii) a search algorithm to explore the conformational space accessible to the system; (iii) a scoring function to evaluate the strength of interaction and the quality of the generated complex.

Docking algorithms are becoming more and more sophisticated. The flexibility in both ligands and receptors is necessary to take into account. There are three main models of docking based on the flexibility of receptor: rigid protein docking, partial protein flexibility and full protein flexibility [Sousa et al. 2006; B-Rao et al. 2009]. Most of the existing methods assume that the conformation of the protein can not change. In the second model, protein is assumed to be flexible only at the binding site or it can be modeled by adding a few DOF in the protein binding site to the combined protein-ligand conformation space. These DOF represent only a very small fraction of the total conformational space that is available but should account for a significant difference in binding energy values. The last model takes into account all the DOF of protein when using a conformational ensemble of slightly different protein structures coexisting in a low energy region of the potential energy surface [Totrov and Abagyan 2008].

Three classes of search algorithm can be distinguished to deal with flexible ligand [Sousa et al. 2006]: systematic methods, random or stochastic methods, and simulation methods. In the first class, the methods try to explore all the DOF of the ligand. In order to reduce the combinatorial explosion, two usual ways were proposed: fragmentation of ligand and using of conformational ensembles. The fragmentation approach incrementally grows the ligand into the active site, either by docking several fragments into the active-site and then linking them covalently to recreate the initial ligand, or by docking firstly a rigid fragment of the ligand and then adding subsequently and successively other flexible fragments [Sousa et al. 2006]. Two popular programs use this approach are FlexX [Kramer et al. 1999], DOCK [Ewing and Kuntz 1997]. Another approach which used libraries of pre-generated conformations of flexible ligand was also presented as in FLOG [Miller et al. 1994]. In the second class, the MC and the GA methods are often used to sample randomly different conformations of the ligand. These conformations are then filtered by a predefined probability function. Some common programs of this class are ICM [Abagyan et al. 1994] using MC-based method, and AutoDock [Morris et al. 1998], GOLD [Jones et al. 1997] using GA-base methods. In the last class, MD methods are often chosen with some modifications such as using very high temperatures in some parts of the MD simulation, or using different start positions of the ligand. One example is CDOCKER [Vieth et al. 1998] which combines MD, MC and GA for docking ligand-receptor complexes. Despite the promising recent results, ligand-protein docking problem still holds several hidden weaknesses. In particular, more adequate scoring functions, able to efficiently combine both accuracy and speed, remain to be investigated.

3.1.3.2 Access/Exit Pathways

The simulation of access/exit pathways of ligand from the surface to the active-site of receptor has been less studied than the protein-ligand docking, although the knowledge of entry/exit pathways of a receptor should be very useful in discriminating between different ligands for computational drug design [Genest et al. 2008].

Different methods have been proposed to investigate the conformational changes of receptor due to

ligand exit. A MD-based variant, called Random Acceleration Molecular Dynamics (RAMD) [Ludemann et al. 2000], uses an additional force applied to the center of mass of the ligand, in addition to the standard MD force-field, in a randomly chosen direction in order to enhance the probability of ligand access to or exit from the active site. After a number of time steps, if the distance traveled by the ligand can not reach the threshold distance, then a new force direction is randomly changed. Otherwise, the force direction is maintained. This process is iterated until the expulsion of the ligand from the buried active site. Another MD variant, called Steered Molecular Dynamics (SMD) [Izrailev et al. 1998], also applies external forces to a ligand to facilitate its unbinding from a protein. In SMD simulations, the force direction is usually defined by the user through an haptic device. This direction is accepted or rejected depending on some factors such as conservation of secondary structure of the protein, deformation of the protein, the magnitude of the force applied, the average velocity of the ligand along the unbinding pathway. Although these methods have been shown to provide biologically relevant information, they remain yet computationally expensive. Besides, the artificial force introduced for accelerating the simulation may yield biased results about the induced conformational changes, so that the interest of simulating with an accurate molecular force field is partially lost.

Other computational techniques have been proposed in order to compute the access channel of ligands into proteins. CAVER [Petrek et al. 2006] performs systematic exploration of the protein interior. It is based on the construction of a vertex-weighted graph from a discrete three-dimensional grid model of the protein. The weights are computed from the distance to the protein atoms, the lowest weights corresponding to nodes with the highest clearance. A variant of Dijkstra's algorithm is applied to search for the shortest low-cost paths. Another similar method, called MOLE [Petrek et al. 2007], also applies a Dijkstra's path search algorithm, but it uses a Voronoi mesh of the space between protein atoms instead of computing a grid. MolAxis [Yaffe et al. 2008] computes the medial axis of the free space inside the protein using efficient computational geometry tools. The medial axis is represented by a collection of two-dimensional surface patches. All these techniques treat rigid proteins. Thus, molecular flexibility can only be indirectly treated by applying these techniques to a set of structures (e.g. samples of a molecular dynamics simulation).

Finally mention that a recent study [Burendahl et al. 2009] on the unbinding mechanism of ligands from estrogen receptor (ERα and ERβ) combined the strengths of RAMD, SMD, and CAVER methods. They applied RAMD simulation to enhance the molecular dynamics sampling by the addition of external forces to ligand, which facilitates the passing of energy barriers. SMD method and CAVER were then used to obtain unbinding pathways and their characterization.

3.1.4 Macromolecular Interactions

The formation of biological complexes between proteins plays a key role in many biological processes (i.e. cell signaling, gene transcription, the immune response). The affinity and the lifetime of protein complexes vary widely from obligate and permanent to transient. Most of the computational methods focus only on predicting bi-molecular association of transient complexes given the structure of the two sub-units. This problem is known as the protein-protein docking problem. Other methods have been developed to study, with more generally, protein association/dissociation processes. These two types of methods are detailed next.

3.1.4.1 Protein-Protein Docking

Since it is often more difficult to determine experimentally protein-protein complexes than their isolated components, the number of experimentally solved protein-protein complexes are still small. *In silico* approaches are often used to solve the protein-protein docking problem. Despite the variety of available methods, the accuracy of their predictions is still limited [Moreira et al. 2010]. A number of excellent reviews give an entire overview of existing protein-protein docking methods [Cherfils and Janin 1993; Halperin et al. 2002; Andrusier et al. 2008; Ritchie 2008; Moreira et al. 2010].

As in ligand-protein docking problem, protein-protein docking has also three main ingredients: a representation of the system, a conformational space search technique, and a ranking function of search results. For the search technique, a same approach has been chosen by many docking methods, in which a smaller protein is used as the probe, and the other as the receptor, which remains fixed during the docking process. A scoring function then ranks the complementarity of computed complexes (i.e., geometric, electrostatics, or hydrophobic, or all three).

Different search techniques have been used to successfully find out the possible binding interfaces between two proteins in docking algorithms. MC methods that randomly sample in the 6-dimensional space to create candidate locations are presented in [Gray et al. 2003]. Other methods, such as ATTRACT [Zacharias 2003; Zacharias 2005], apply systematic search combined with energy minimization. The Fast Fourier Transform (FFT) has been often used in rigid-body docking [Katchalski-Katzir et al. 1992; Gabb et al. 1997; Li et al. 2003; Kozakov et al. 2006]. A progam called ZDOCK [Li et al. 2003] uses an FFT-based algorithm along with an optimized scoring function, involving three major terms: pairwise shape complementarity, electrostatics, and desolvation, to treat rigid-protein docking. To overcome the limitation of treating unbound proteins as rigid bodies, ZDOCK softens the protein surfaces by allowing light overlaps between two protein interfaces to take into account possible conformational changes. However, these FFT-based methods could take CPU-days to execute, because even though the FFT is used to enumerate rapidly the translations of the probe, the rotations are still searched systematically. Other algorithms use matching of surface cubes or geometric hashing (i.e. the identification and matching of convex and concave protein surface regions) [Connolly 1986; Norel et al. 1994; Norel et al. 1995]. A representative program of this class is PatchDock [Duhovny et al. 2002] that docks unbound proteins by matching geometric patches (concave, convex, and flat surface pieces) of almost equal areas after detecting these patches by applying a segmentation algorithm in order to generate candidate transformations. Protein complementarity can also be determined by a Voronoi-diagram-based model of macromolecular interfaces [Cazals et al. 2006]. However, local shape feature matching based methods are usually not sufficient when the available structures of the proteins are unbound [Smith and Sternberg 2002].

Multi-stage approaches have been often proposed to treat flexibility within protein-protein docking methods. Proteins are usually treated as rigid bodies in the first stage, and the aforementioned search methods are applied to find candidate conformations of the complex. In a latter stage, these structures are refined and re-ranked by using more expensive exploration methods and more accurate energy function [Li et al. 2003; Lorenzen and Zhang 2007; Pierce and Weng 2008]. Nevertheless, flexibility can also be treated in the first stage by constructing multiple copies of the proteins [Bastard et al. 2006].

3.1.4.2 Protein-Protein Association and Dissociation

The assembly of proteins is a multi-scale process from the interactions of their surface atoms to the organization (i.e. the assembly order) of these proteins to form specific complexes. A deep understanding of protein-protein association is therefore essential for accurate modeling and prediction of protein meta-structures.

Different computational approaches have been proposed to investigate protein-protein association. The calculation of bimolecular association rate constants can be performed by simulation of the diffusional motion of the interacting particles using particle-based Brownian dynamics simulations [Northrup and Erickson 1992]. In this method, one of the interacting molecules is placed at the center of a sphere, while the other one starts Brownian moves at a given distance where there are no forces between the molecules at this separation or the forces are centrosymmetric. The rate constant for the molecules to approach the given distance is then computed. The association rate constant can be calculated based on thousands of generated trajectories. Using a simpler approach, the variation in the rate constant for bimolecular association is determined from the electrostatic interaction energy between proteins in transient intermediate configurations [Zhou 1997].

For the inverse problem that considers protein complex dissociation, the two points to be investigated are the interactions between proteins leading to the denaturation of complexes, and the order in which proteins are disassembled from the complexes. Very few computational methods addressing protein dissociation have been proposed. Pogorelov *et al.* [Pogorelov et al. 2007] used an all-atom steered molecular dynamics (SMD) simulation in combination with evolutionary studies and binding free analysis to explore disassociation pathways of one subunit from a membrane protein complex. Nesatyya *et al.* [Nesatyya and Laskinb 2002] presented a MC simulation of collisionally activated dissociation of non-covalent protein complexes in the collision cell of a triple quadrupole mass spectrometer. They used the hard-sphere and the diffuse scattering models to simulate the evolution of the translational and internal energies of activated precursor ions along the collision cell. Dissociation rate constants and ion survival probability were then calculated based on the estimated internal energy content of the excited ion.

Finally note that computational methods have been proposed to complement experimental data on the structure and the dynamics of protein assemblies. For instance, Cazals and Dreyfus [Cazals and Dreyfus 2009] introduced a novel method to build models of large assemblies of proteins from low-resolution structural data. In their model, a toleranced protein is presented as a collection of toleranced balls, and a toleranced assembly as a collection of toleranced proteins. Such an assembly, having a continuum of possible geometries, is encoded in a special graph called a Hasse diagram, which is a forest of trees representing the complex. The information extracted from this diagram can be used to investigate protein dynamics inside the complex.

3.2 Motion Planning Approaches

Motion planning is a fundamental problem in robotics and has been extensively studied during the past decades. In its original basic form, motion planning ignores typical concerns such as optimality and uncertainty, but instead focus on the computationally difficult problem of generating feasible collision-free

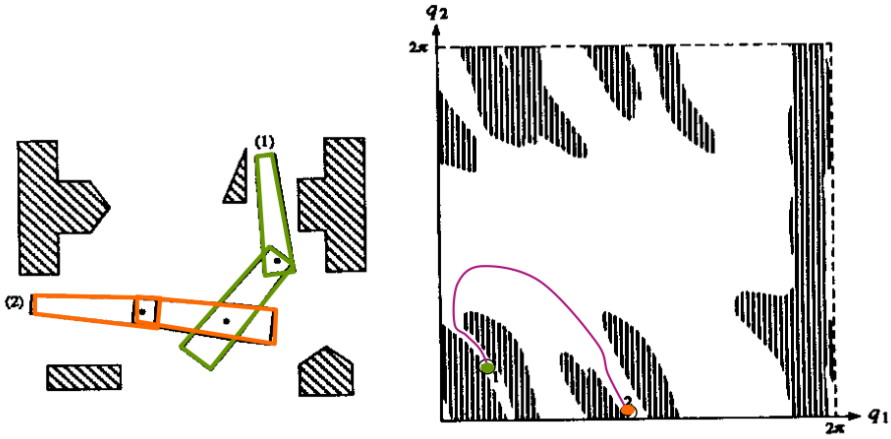


Figure 3.1: Illustration of the motion planning problem for a 2R manipulator [Latombe 1991]: (a) The manipulator and the obstacles in the workspace, (b) Representation of the manipulator and the obstacles in the configuration space.

paths that take the robot from a given start to a given goal configuration among a collection of workspace obstacles. The vast body of works related to motion planning can hardly be given adequate coverage in this brief review which focuses on techniques relevant with the planning needs of the thesis. Several surveys [Latombe 1991; Choset et al. 2005; LaValle 2006] can be consulted for further reading.

Motion planning problems are normally formulated in the configuration space of the robot (see Fig. 3.1). Obstacles in the environment translate into so-called C-obstacles in the configuration space, allowing a unified way to represent a robot motion as a curve in the configuration space that avoids the C-obstacles. The complexity of the C-obstacles is exponential in the dimension, meaning that an exact characterization in computationally too costly, even for robots with a relatively small number of DOF. At a broad level, motion planning approaches can be subdivided into three classes: cell-decomposition methods, roadmap methods, and potential field (or local) methods. **Cell-decomposition** methods divide the free part of the configuration space into a number of cells. Motion is then planned through these cells. Unfortunately, when the dimension of the configuration space get higher or when the complexity of the scene is large, the number of cells required becomes too large to be practical. **Roadmap methods** construct a network of roads through the configuration space along which the object can move without collision. This roadmap can be seen as a graph, and the problem is reduced to graph searching. Unfortunately, computing an effective roadmap in a deterministic way can only been achieved for rather low-dimensional configuration spaces with few obstacles. Finally, **Potential field methods** and other local methods steer the robot by determining a direction of motion based on local properties of the scene (e.g. move in the goal direction while being pushed away by nearby obstacles). These methods work well in relatively open area but, because only local properties are used, the robot can get stuck in deadlock position.

In the following, we will base ourselves on more recent Sampling-based roadmap methods [Kavraki et al. 1996; LaValle 1998] that have now emerged as a general and effective framework, successfully used in various application domains for articulated robots with possibly complex kinematics.

3.2.1 Sampling-based approaches

An important characteristic of sampling based planners is that they do not attempt to explicitly construct a model of the C-space. Instead, they directly capture the connectivity of the free C-space inside a roadmap of sampled configurations. The big advantage of this computational scheme is that its complexity tends to be dependent on the difficulty of the path, and much less on the global complexity of the scene or the dimension of the configuration space. The key idea is to use a collision detection algorithm as a black box, which allows the planner to ignore all issues associated with the precise geometric models of the robot and obstacles. The efficiency of available collision detection algorithm also contributes to the overall practical performance of sampling-based planners that also achieve some desired forms of probabilistic completeness.

Sampling-based approaches can be grouped in two main families: PRMs [Kavraki et al. 1996] use sampling for constructing a roadmap capturing the free-space connectivity while single-query variants (e.g. RRTs [LaValle 1998]) combine sampling within incremental search methods for searching for a particular path. The choice mainly depends on the application. PRM methods are more suitable when multiple motion planning queries must be solved in a same environment. Computing time is spent to preprocess in advance a roadmap data structure that can be used afterwards for fast query answering. In contrast, single query variants like RRT are incremental search methods that build a new roadmap from scratch for each new query. The roadmap needs only linking the query configurations and generally consists of two trees rooted at these configurations. Hence, single-query methods are in general faster than the PRM preprocessing stage since they do not require a whole roadmap reflecting the free-space connectivity. However, as they focus on solving a particular problem, the processed data structure is less appropriate for later use. Such techniques are particularly suited for solving constrained assembly problems [Chang and Li 1995] where one must check whether there exists a path to remove a part from an assembly.

Note that, it can sometimes be interesting to combine both kinds of methods. For instance, when only slight modifications are produced in the environment, a multiple-query method can be applied at a global level and then single-query methods can rapidly solve local problem arisen from these slight changes [Jaillet and Siméon 2004]. Also, the use of single-query planner within a multiple-query motion planning framework can be used for the efficient parallelization of the planning process [Akinc et al. 2003]. In the next sections, the two approaches will be detailed.

3.2.2 The PRM approach

The *Probabilistic RoadMap* (PRM) method is a popular approach to motion planning [Kavraki et al. 1996]. It has received a lot of attention in recent years [Amato and Wu 1996; Boor et al. 1999; Lucia K. Dale 2001; Wilmarth et al. 1999; Bohlin and Kavraki 2000; Siméon et al. 2000; Hsu et al. 2003; Burns and Brock 2005; Saha et al. 2005; Jaillet and Siméon 2006]. It is now used by many researchers considered the most widely applicable approach to motion planning. PRM is a general planning framework. The method has been successfully applied to many motion planning problems dealing with robot arms [Kavraki et al. 1996], car-like robots [Svestka 1996; Sekhavat et al. 1998], manipulation tasks [Alami et al. 1994; Ahuactzin and Gupta 1995; Nielsen and Kavraki 2000; Siméon et al. 2004], motion planning with uncertainty [Alterovitz et al. 2008], closed-kinematic loops [Han 2004; Cortés 2004] like two mobile robot arms that together hold

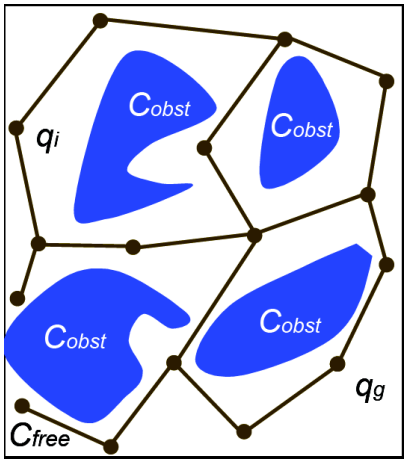


Figure 3.2: Illustration of the construction of the roadmap in PRM approach.

an object, and even flexible objects [Holleman et al. 1998; Lamiraux and Kavraki 2001]. In all these cases, the method is very efficient but, due to the probabilistic nature, its performance remains difficult to analyze (see e.g. [Kavraki et al. 1996]).

PRM is a roadmap technique but, rather than constructing the roadmap in a deterministic way, a probabilistic approach is used. Most of the work lies in the construction of the roadmap, but this can be done in a preprocessing phase. This is important for applications in which fast query answering is required. The overall principle is to capture the connectivity of the collision-free space by a set of one-dimensional curves stored in a pre-computed roadmap (see Fig. 3.2). The roadmap is obtained by sampling robot configurations and subsequently connecting promising samples with valid local paths generated by a simple and local planner. Sampled configurations and local paths are checked for collision using effective collision detection algorithms [Otaduy and Lin 2003]. Even thought the local planner might fail, once enough samples have been added to the graph, with high probability the roadmap will be able to solve motion-planning queries. Once the roadmap has been constructed, multiple path planning queries can be answered efficiently by simply connecting the query configurations and searching the augmented roadmap for a solution path, generally smoothed in a post-processing step to improve the quality of solution. This basic scheme has been extended in many ways to adapt the approach to different types of motions and/or to exploit certain properties of the environment.

While PRM is successful for robots with many DOF and probabilistic complete, its performance degrades in the presence of narrow passages that require a prohibitively high density roadmap. A number of variants and extensions have been proposed to alleviate this problem and improve PRM performance, e.g. biasing sampling around obstacles [Amato et al. 1998; Boor et al. 1999; Hsu et al. 2003] or towards the medial axis [Wilmarth et al. 1999; Holleman and Kavraki 2000; Lien et al. 2003], using free-space dilatation [Hsu et al. 2006; Saha et al. 2005], visibility-based filtering [Siméon et al. 2000] or adaptive sampling [Kurniawati and Hsu 2006; Rodriguez et al. 2006], exploiting search space information [Burns and Brock 2005] or delaying collision checks [Bohlin and Kavraki 2000; Sanchez and Latombe 2002]. Unfortunately, the different improvements listed above are difficult to compare . Some of them are limited to particular (e.g. free-flying) robots while others maintain the generality of the PRM framework. Also the effectiveness of the techniques is shown on different test cases, using a different implementation. Therefore

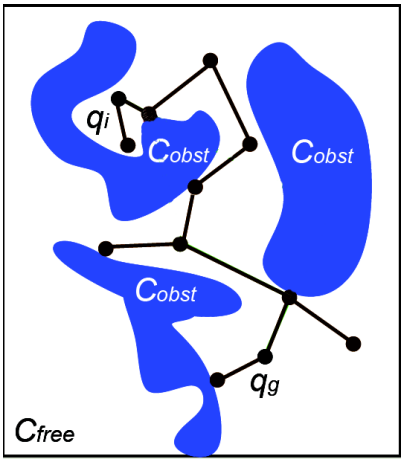


Figure 3.3: Illustration of the construction of the tree in RRT approach from q_i to q_g.

it is still rather unclear, what is the best technique under which circumstances. See [Lucia K. Dale 2001; Geraerts and Overmars 2002] for first studies on this issue.

An originality of Visibility-PRM [Siméon et al. 2000] is to produce compact roadmaps with a small number of nodes. The method relies on a free-space structuring into visibility domains (i.e. sets of configurations connectable to a given guard by a valid local path). Computed guards are linked together via connectors located in their overlapping visibility regions. Such roadmaps can be constructed using a simple PRM variant: each free sample is added to the roadmap only if it cannot be connected to any existing node (i.e. guard) or if it connects at least two components (i.e. connector). Visibility-PRM offers two advantages. First, the algorithm termination is controlled by the difficulty of adding a new guard, which relates to the quality of the roadmap in term of coverage. Secondly, the computed roadmap is a very small tree (i.e. no cycles) capturing the free space coverage with a limited number of nodes and edges. The tree restriction is relaxed with the Path Deformation Roadmap extension [Jaillet and Siméon 2009]. Finally, current PRM versions still need to spend some amount of time during query answering, both to find the correct path in the roadmap and to improve the path [Song et al. 2001]. Some work has been done in this direction for improving the query answering performance of PRM. The solution proposed in [Geraerts and Overmars 2005] aims at enhancing the preprocessing phase such that motions can be computed much more efficiently and the results are much better in terms of quality (e.g. smoothness, path length), removing the need for costly post-processing of the path.

3.2.3 Incremental Search Planners

Incremental search methods [Barraquand and Latombe 1991; Bessière et al. 1993; Hsu et al. 2000; LaValle 1998] are PRM variants introduced for single-query planning. They are more efficient for that purpose since their exploration strategy is biased to solve a particular query and not to obtain information about the exploration strategy about the whole configuration 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: *Unidirection* methods develop a single tree from one of the query configurations until the other configuration is reached, while *Bidirectional* methods involve two trees rooted at query

configurations. The motivation for using one or more trees depends on the characteristics of problem to be solved. For example, if the robot is highly constrained around the initial position and rather free to move around the goal, it will be more efficient to build only a tree rooted at this initial configuration. In other situations, a single tree may be trapped trying to find the exit through a narrow passage, while traveling in both directions may be easier.

The Rapidly-exploring Random Tree (RRT) approach, introduced in [LaValle 1998], has become now the most popular single-query motion planner. RRT-based algorithms were first developed for non-holonomic and kinodynamic planning problems [LaValle and Kuffner 2001a] where the space to be explored is the state-space. However, the RRT framework is general and can also be applied for problems without differential constraints [LaValle and Kuffner 2001b]. The key idea of the RRT expansion is to bias the exploration toward unexplored regions of the space. Hence, the probability that a node is selected for expansion is proportional to the area of its Voronoi region (i.e. set of points closer to this node than to the others). Therefore, RRTs are biased by large Voronoi regions to rapidly explore, before uniformly covering, the space.

RRTs combine a construction and a connection phase (see Fig. 3.3). For building a tree, a configuration is randomly sampled and the nearest node in the tree (given a distance metric in C-space) is expanded toward the sample. In the basic RRT algorithm (referred to as RRT-Extend), a single expansion step of fixed distance is performed. In a greedier variant, RRT-Connect, the expansion step is iterated while keeping feasibility constraints (e.g. collision-freeness) are satisfied. RRT-Connect has been shown to be more efficient than the single step version for systems without differential constraints [LaValle and Kuffner 2001b].

One weakness of the RRT algorithm comes from its high sensitivity to the distance metric used to select the tree node to be expanded. Several approaches [Cheng and LaValle 2001; Urmson and Simmons 2003; Cortés et al. 2007] have been proposed to address this issue and design more sophisticated metrics than the Euclidian metric used in basic RRT, e.g. by applying weights to balance the influence of the DOF. However, in high dimensional spaces, it is difficult to find the adequate metric-showing a good trade-off computational complexity and accuracy.

Other recent improvements concern configuration sampling, normally made using a uniform random distribution in the configuration-space. More sophisticated sampling strategies, e.g. iterative RRTs [Ferré and Laumond 2004] or sampling in dynamic domains that envelope the search tree [Yershova et al. 2005; Jaillet et al. 2005] have been shown to improve the RRT performance for constrained disassembly problems (see Figure 2.3). While these methods address (dis)assembly problems involving rigid objects, the extension presented in [Cortés et al. 2009] is particularly devised for the disassembly of complex objects with articulated parts. In this type of problem, configuration parameters generally play two different roles. Some of them are essential for the disassembly task, while others only need to move if they hinder the progress of the disassembly process. The proposed method is based on such a partition of the configuration parameters. Results show a remarkable performance improvement compared to standard RRT method.

Another interesting variant called Transition-based RRT (T-RRT) [Jaillet et al. 2010] has been recently proposed in order to compute good-quality paths with respect to a cost function defined over the configuration space. Basically, T-RRT introduces a state-transition test based on the Metropolis criterion, similarly to MC methods, in order to reject new states that will decrease the quality of the path. This

filtering, which relies on the steepness of the local motion to connect a given state to the RRT, makes the tree expansion tends to follow the valleys and the saddle points of the configuration-space costmap. This technique shows a good capacity to find good-quality paths in reasonably high-dimensional spaces constrained by obstacles.

3.3 Motion Planning Approaches to Problems in Structural Biology

Motion planning algorithms, originally developed in the field of robotics, are efficient tools for exploring constrained high-dimensional spaces. When applied to structural biology, they yield high-performance conformational search methods that are able to consider a wide range of DOF. This section gives a brief overview on the application of motion-planning-based methods to different problems in structural biology.

3.3.1 Protein Motions

3.3.1.1 Protein Loop Motions

In order to study large-amplitude motions of flexible molecules, an approach with two stages has been proposed [Cortés et al. 2005]. The driving idea is to separate the conformational search in two stages aiming to highly speed up the computation. The first stage consists in a geometric filtering operated by motion planning techniques applied on articulated hard sphere models. The second stage accounts for the energy-based accuracy only for selected solutions found at the previous stage. The interest of this geometric filtering is that high-dimensional conformational spaces can be globally explored in a continuous way.

Another work to model flexible protein loops has been carried out by Yao *et al.* [Yao et al. 2008]. The authors proposed two algorithms called *seed* and $deformation$ sampling. The seed algorithm tries to sample conformations broadly distributed over the space of closed collision-free conformations of a flexible protein loop. The deformation sampling operation then starts from a given conformation sampled by the previous algorithm and deforms this conformation without breaking closure constraint or making auto-collision while modifying the DOF of loops. This method shows its promising results on handling 5 to 25 residue long loops. They have also applied this method to interpret fuzzy regions in electron-density maps obtained from X-ray crystallography.

Shehu *et al.* have developed Fragment Ensemble Method (FEM) addressing equilibrium mobility in the loop modeling problem [Shehu et al. 2006]. The method combines a statistical mechanics formulation and a robotic-based exploration of the conformational space to model a loop fragment. Starting from an incomplete protein structure and the amino acid sequence of the missing loop, the proposed method generates an ensemble of low-energy loop conformations that completes the given protein structure.

3.3.1.2 Helix and Domain Motions

Del Carpio *et al.* [Carpio et al. 2005; Carpio et al. 2006] presented a novel algorithm that resolves the intra-molecular loop and domain motions during the interaction of proteins. This hybrid approach firstly maps regions or domains (including loops) with high flexibility in the interacting units by using a graph

theoretical method to account for protein flexibility reduction. The PRM algorithm is then used to generate different configurations pathways that provide information about the protein complex energy.

Enosh *et al.* have developed an automated method to predict and simulate motion in the transmembrane (TM) proteins of the α-helix bundle type by using an RRT-based motion planning algorithm [Enosh et al. 2006]. The resulting paths are ranked according to the quality of the van der Waals interactions between the TM helices (i.e. Lennard-Jones potential). They have focused on simple systems comprising pairs of α-helices, thus circumventing the complexities of modeling loops that connect pairs of helices.

Kirillova *et al.* have proposed an approach for computing large-amplitude domain motions by combining an path-planning method and normal mode analysis [Kirillova et al. 2008]. The main idea is to guide the conformational exploration, performed with a path planning algorithm on a purely geometric molecular model, using the directions of collective atomic motions given by the low-frequency normal modes. Indeed, the algorithm explores the space of the collective DOF provided by the low-frequency modes, which is a lower-dimensional sub-manifold of the conformational space. This combination allows the method to overcome the limitations of each individual method for computing large-amplitude conformational changes in proteins.

3.3.2 Protein-Ligand Docking and Access/Exit Pathways

Singh *et al.* [Singh et al. 1999] developed a novel technique for studying the dynamics of protein-ligand interactions based on motion planning algorithms. This algorithm uses electrostatic and van der Waals potentials to compute the most energetically favorable path between any given initial and goal ligand configurations. The PRM algorithm is used to sample the distribution of possible paths to a given goal configuration and to compute an energy-based "difficulty weight" for each path. By statistically averaging this weight over several randomly generated starting configurations, they computed the relative difficulty of entering and leaving a given binding configuration. This approach yields details of the energy contours around the binding site and can be used to characterize and predict good binding sites. They used two attributes to distinguish between true and predicted binding configurations: the absolute energy of the ligand and the average weights of all paths entering and leaving the configuration. They found that these average weights of the true binding configuration were significantly higher than the ones of all other low-energy configurations.

Based on the main idea of the previous method, Apaydin *et al.* proposed a novel approach for studying of computational mutagenesis and on binding sites [Apaydin et al. 2002]. They introduced the Stochastic Roadmap Simulation (SRS) to compute the escape time from a putative binding site which is the number of MC simulation steps starting from the binding site to reach a conformation outside its funnel of attraction [Camacho and Vajda 2001]. SRS allows to analyze all the paths in the roadmap simultaneously. In their first study, they measured the effects of mutations on the catalytic site of a protein. For 6 mutations, the escape times of the ligand in the bound conformation are computed and then compared with the wide-type's ones. These results obtained with SRS agreed with the biological interpretation of the mutation. In the second study, the escape time in 5 of 7 complexes from the funnel around the catalytic site is larger than any other putative binding sites' one by at least an order of magnitude.

Cortés *et al.* have presented a novel variant of RRT algorithm to treat the problem of ligand exit from

the active site of a protein by using a purely geometric approach in which the energetic terms are replaced by constraints: non-bonded atoms are separated by a minimum distance to account for van der Waals repulsion terms, privileged orientations and distances account for the possible formation of hydrogen bonds or hydrophobic interactions [Cortés et al. 2005; Cortés et al. 2007]. The proposed method, called Manhattan-like RRT (ML-RRT), addresses disassembly path planning of articulated objects. The particularity of this algorithm is that the motions of the different parts are decoupled. Indeed, for the present application, ligand and protein degrees of freedom (only protein side-chain flexibility is considered) are treated at different levels. The ligand motion is privileged, while flexible parts of the protein only move if they hinder the ligand progression. This new algorithm presents two advantages with respect to the basic RRT. First, the computing time and its variance are notably reduced. And second, but not less important, the flexible parts that have to move for finding a solution path are automatically identified. Thus, the planner is able to handle models involving hundreds of potential DOF, avoiding user intervention to select the important ones.

3.3.3 Protein folding/unfolding

For studying protein unfolding pathways, Song *et al.* have proposed a framework based on PRM motion planning algorithm [Song and Amato 2001; Song and Amato 2004]. The objective is to compute the most energetically favorable paths between the folded state and denatured states. A high-quality roadmap is constructed by using Gaussian sampling around the folded state with various standard deviations to create new conformations. The analysis of the computed unfolding pathway may provide information about the order of formation of secondary structure elements.

Stochastic Roadmap Simulation (SRS), developed by Apaydin *et al.* [Apaydin et al. 2002], is a general technique to study molecular motion. This technique is also applied to the protein folding problem [Apaydin et al. 2003]. A number of conformations is sampled randomly from the conformation space. These conformations of a protein are represented as nodes in a compact graph with the edges representing the probability of moving between neighboring states. Once the roadmap construction is finished, techniques from Markov-chain theory are used to calculate the probability of each node to tend toward a folded or an unfolded state. The overall calculation of such folding probabilities allows to determine which structures constitute the transition state of the folding process, and to estimate the folding time for the protein.

3.4 Contribution of this Thesis

Computational methods have been developed to complement the experimental studies and to better face the quantity of data issued from genomics and proteomics. In this context, understanding and predicting structure-function relationships in proteins with fully *in silico* approaches remain today a great challenge. Proteins are able to undergo possibly large conformational changes that affect their ability to interact with other molecules, and despite great advances achieved in the last years, dealing with macromolecular flexibility largely remains out of reach of the existing molecular modeling tools.

Our approach to deal with the molecular flexibility in studying molecular interactions is based on path planning algorithms proposed. Such algorithms are efficient tools for exploring constrained high-dimensional spaces. Applied to problems in structural biology, they yield high-performance conformational

search methods, able to consider a wide range of degrees of freedom. We present a new RRT variant for disassembly path planning of articulated objects (see Chapter 4). The new method was developed to circumvent the limitations of the basic RRT algorithm for dealing with disassembly problems involving complex articulated objects. The main idea is to facilitate the tree expansion by considering separately two types of conformational parameters, called active and passive. Active parameters are essential for the disassembly problem, and they are directly treated at each iteration of the algorithm. Passive parameters, however, only need to be treated when they hinder the expansion of active parameters. The advantage of this decoupled treatment, that favors the expansion of the active parameters, is to maintain the exploratory strength of the RRT algorithm while dealing with high-dimensional problems. Two extensions have also introduced to deal with more complex systems involving several levels of passive parameters, or requiring pushing and pulling motions of passive parts.

The technique presented in Chapter 4 is then applied for simulating ligand diffusion motions, together with the possibly induced conformational changes of the protein. Given an initial structure with the ligand docked inside the protein, the proposed method computes a path (i.e. continuous sequence of conformations) simulating the ligand exit. Such a path search problem is formulated as a mechanical disassembly problem, where the protein and the ligand are modeled as articulated mechanisms (see Chapter 5).

Generally, disassembly sequencing and path planning are parts of a whole problem, and ideally, they have to be treated simultaneously. Therefore, we propose a new method for simultaneously disassembly sequencing and path planning in a general framework (see Chapter 6). It extends the above disassembly path planning algorithm for two objects with articulated parts. The idea developed in this chapter consists in iterating the disassembly algorithm for extracting all the parts of a general assembly. This method can extract not only path motion for each part to be disassembled from the complex but also the preferred order that the disassembly process has to follow. A deep analysis of its performance is also carried out.

The disassembly sequencing method is then used to study the protein complex dissociation processes. The method presented in Chapter 6 requires to be extended, since it is necessary to take into account the interaction energy along the dissociation pathways between molecules. Therefore, a coarse-grained model has been used to facilitate the energy computation. This approach has been applied for several protein complex models, and it shows some promising preliminary results (see Chapter 7).

4

Disassembly Path Planning for Complex Articulated Objects

This chapter addresses the problem of automatically computing motions to disassemble objects. Assembly and disassembly planning are important problems in manufacturing engineering. Many techniques have been developed in this field for automatically generating (dis)assembly plans that optimize time, cost, etc. [Bourjault ; Homem de Mello and Lee 1991]. Most of these techniques are based on *relation graph models* of the assembly or *precedence graphs* and use graph theory and AI algorithms for computing disassembly sequences. The (dis)assembly problem can be formulated as a general path planning problem [Latombe 1991; LaValle 2006]. Indeed, path planning concepts and algorithms have been applied to solve different instances of the (dis)assembly planning problem. Sampling-based tree planners [Kavraki and Latombe 1998; LaValle 1998] have been shown to be efficient techniques for solving constrained disassembly problems between rigid objects. In this chapter, we present an algorithm that extends prior works to disassembly problem between complex articulated objects. This algorithm will be applied in Chapter 5 to simulate ligand pathways in flexible (articulated) protein receptors. We first provide a short state of the art in Section 4.1 and the problem formulation in Section 4.2. The general RRT-based algorithm will be next described in Section 4.3. Section 4.4 gives a detailed presentation of our variant along with its performance analysis (Sec. 4.5). Two extensions of the ML-RRT algorithm, permitting to tackle more complex problems, are described in Sections 4.6 and 4.7. The last section contributes to some discussions of our method (Sec. 4.8).

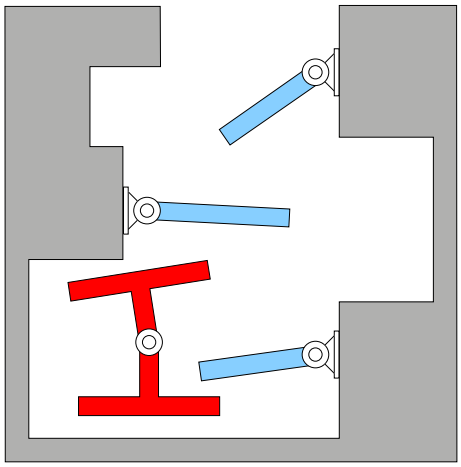


Figure 4.1: Disassembly path planning problem for two objects with articulated parts. The problem consists in finding a path to extract the small (red/dark) object from the big one.

4.1 Related Works and Overview of the Proposed Method

A number of approaches were presented to deal with assembly and disassembly planning problems. Geometric reasoning approaches have been proposed for reducing the combinatorial complexity of the problem, as well as the amount of information that has to be provided by the user. Wilson's pioneering work on geometric reasoning about mechanical assembly [Wilson 1992] introduces the *directional blocking graph* (DBG), which identifies which parts collide given an instantaneous displacement in a given direction, and the *non-directed blocking graph* (NDBG), which represents how parts are constraining each other, based on a partition of the space of allowed motions and on the associated DBGs. Many subsequent (dis)assembly sequencing methods have used these two concepts. The approach presented in [Lozano-Pérez and Wilson 1993] generalizes the solutions in [Wilson 1992] to arbitrary motions between parts. The method involves constructing configuration space diagrams that explicitly represent interferences between pairs of parts for every relative motion. A similar approach is developed in [Halperin et al. 2000], based on the concept of *motion space*, which is an extension of the notion of configuration space, and represents possible motions of sub-assemblies.

The assembly maintainability study [Chang and Li 1995] is a variant of the (dis)assembly problem. Given an assembled system, maintainability studies are conducted to determine if it is possible to remove a particular part, and if so, to obtain the disassembly path. Normally, such studies involve only one mobile part, and therefore, "standard" path planning algorithms could be applied. However, the workspace is usually extremely constrained in this context, and problem-specific algorithms are required for efficiently computing disassembly paths. A fast and effective algorithm for this kind of problems is presented in [Ferré and Laumond 2004]. The method is based on an iterative RRT-like algorithm that reconstructs some parts of the search tree while progressively increasing the size of the objects.

All the methods above address (dis)assembly problems involving rigid objects. The method we present in this chapter is well suited for assembly maintainability studies in which the disassembled objects have articulated parts. Figure 4.1 illustrates a simple two-dimensional example. Our algorithm is a variant of the RRT algorithm [LaValle 1998]. The particularity of the proposed variant is to introduce two types of

configuration parameters, labeled as *active* and *passive*, and to generate their motion in a decoupled manner. We call this variant Manhattan-like RRT (ML-RRT) because the computed paths look like Manhattan paths over these two sets of parameters that change alternatively. The partition of the configuration parameters into active and passive corresponds to their role in the disassembly problem. Active parameters are essential for the disassembly task, while passive parameters only need to move if they hinder the progress of the disassembly process. The ML-RRT algorithm presents two main advantages with respect to the basic RRT: (1) The computing time is notably reduced. (2) The passive parts that have to move for finding a solution path are automatically identified. Thus, the planner is able to handle models involving hundreds of potential degrees of freedom, avoiding user intervention to select the important ones.

4.2 Problem formulation

The disassembly path planning problem treated in this chapter can be formulated as a general path planning problem for a system with multiple mobile objects, using the notion of *configuration space* C [Lozano-Pérez 1983; Latombe 1991; LaValle 2006]. A *configuration* q is a minimal set of parameters defining the location of the mobile system in the world, and C is the set of all the configurations. Given the assembled configuration q_{init} and a goal configuration q_{goal} (any disassembled configuration) the problem consists in finding a feasible collision-free path in C that connects both configurations. Note that contact between parts is considered as a collision by our method. Therefore, contact motions can not be computed. The instance studied in this chapter considers two objects with possibly multiple articulated parts. Considering that the spatial location of one of the objects is fixed, then, the configuration parameters are those defining the pose of the reference frame attached to other (mobile) object plus the degrees of freedom associated with the articulated parts in both objects. Thus, the configuration vector is given by: $q = \{q^M, q^{Jm}, q^{Js}\}$, where q^M contains parameters defining the position and the orientation of the mobile reference frame, and q^{Jm} and q^{Js} represent the joint variables of the articulated parts in the mobile object and the static object respectively.

In the example illustrated by Fig. 4.1, the configuration vector is given by: $q = \{q^M, q^{rm}, q^{rs1}, q^{rs2}, q^{rs3}\}$, where q^M contains three parameters defining the position and the orientation of the mobile reference frame, and q^{rm}, q^{rs1}, q^{rs2}, q^{rs3} represent the joint angles of the articulated parts in the mobile object and the fixed object. Thus, considering joint limits, the configuration space is: $C = SE(2) \times \mathbb{R}^4$.

In general, the most significant parameters for the disassembly of articulated objects are those concerning the pose of the mobile object, q^M. The parameters associated with the articulated parts are relatively less important, since they only need to move if they hinder the progress of the mobile object toward the disassembled configuration. Such different role of the configuration parameters is generally true for the disassembly of two objects with articulated parts. Therefore, configuration parameters can be separated into two sets: $q = \{q^{act}, q^{pas}\}$, with $q^{act} = q^M$ representing the active parameters, and $q^{pas} = \{q^{Jm}, q^{Js}\}$ the passive parameters. The terms active and passive have been chosen in relation to how the algorithm described in Section 4.4 acts on them. This partition induces the corresponding sub-manifolds in the configuration space: $C = C^{act} \times C^{pas}$. Although the above described partition can be generally adopted, any other partition can be defined by the user. The mobile parts are separated into two lists L_{act} and L_{pas} containing the active and the passive parts respectively. For a given partition, q^{act} is the set of configuration parameters associated with the parts in L_{act} and q^{pas} is is the set associated with L_{pas}.

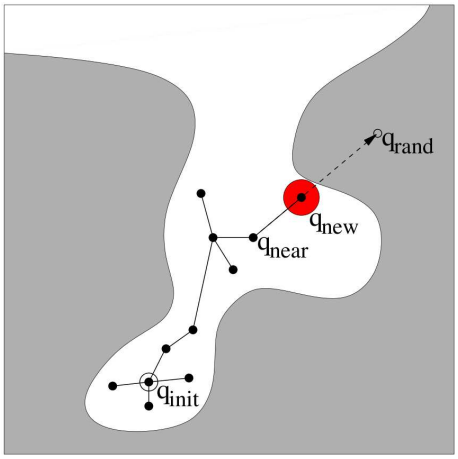


Figure 4.2: Illustration of one expansion step of an RRT search tree. The tree tends to cover C_{feas}: the feasible subset of the configuration space C.

4.3 Basic RRT Algorithm

The basic principle of the RRT algorithm [LaValle 1998] is to incrementally grow a random tree τ rooted at the initial configuration q_{init} in order to explore the reachable configuration space and to find a feasible path connecting q_{init} to a goal configuration q_{goal}. Fig. 4.2 illustrates the process and Algorithm 1 gives the pseudo-code for the RRT construction. At each iteration, the tree is expanded toward a randomly sampled configuration $q_{rand} \in C$. This random sample is used to simultaneously determine the tree node to be expanded and the direction in which it is expanded. Given a distance metric in the configuration space, the nearest node q_{near} in the tree to the sample q_{rand} is selected and an attempt is made to expand q_{near} in the direction of q_{rand}. For kinematically unconstrained systems, the expansion procedure can be simply performed by moving on the straight-line segment between q_{near} and q_{rand}. If the expansion succeeds, a new node q_{new} and a feasible local path from q_{near} are generated. The key idea of this expansion strategy is to bias the exploration toward unexplored regions of the space. Hence, the probability that a node will be chosen for an expansion is proportional to the volume of its Voronoi region (i.e. the set of points closer to this node than to any other node). Therefore, RRT expansion is biased toward large Voronoi regions enabling rapid exploration before uniformly covering the space.

Different strategies can be adopted for the design of RRT-based path planners [LaValle and Kuffner 2001b]. Configuration sampling (function `SampleConf`) is normally made using a uniform random distribution in the configuration space C. However, more sophisticated sampling strategies (e.g. sampling in dynamic domains that envelope the search tree [Yershova et al. 2005; Burns and Brock 2007]) may improve the performance of the RRT algorithm. Another technical point concerns the function `NearestNeighbor`. The basic RRT algorithm selects q_{near} as the nearest node to q_{rand} using an Euclidean metric[1] in C. Such a metric distance is very simple and easy to compute. However, since it does not consider motion constraints (e.g., obstacles, kinematic constraints), it may lead to a poor performance of the planner,

[1] We use a weighted metric for translation and rotation components, with 3D rotations represented by Euler angles. Note however that the use of unit quaternions will be more appropriate [Kuffner 2004].

Algorithm 1: Construct_RRT

```
input  : the configuration space C;
         the root q_init and the goal q_goal;
output : the tree τ;
begin
    τ ← InitTree(q_init);
    while not StopCondition(τ, q_goal) do
        q_rand ← SampleConf(C);
        q_near ← NearestNeighbor(τ, q_rand);
        q_new ← Expand(q_near, q_rand);
        if not TooSimilar(q_near, q_new) then
            AddNewNode(τ, q_new);
            AddNewEdge(τ, q_near, q_new);
end
```

by repeatedly selecting "exhausted" nodes for futile expansion. To avoid this problem, two modifications can be introduced in `NearestNeighbor`: (1) A node is no longer selected after its expansion fails a given number of consecutive times l. (2) q_{near} is selected at random among the k nearest neighbors[2]. The efficiency of these two modifications has been shown in related works [Cheng and LaValle 2001; Urmson and Simmons 2003; Cortés et al. 2007]. One can also choose a more or less greedy strategy for the expansion procedure (function `Expand` in Algorithm 1). In the basic RRT algorithm, a single expansion step of fixed distance is performed. Here we use the RRT-Connect variant [LaValle and Kuffner 2001b], which iterates the expansion step while feasibility constraints are satisfied. This variant is in general more efficient than the single-step version for systems without differential constraints, which are the type of systems considered in this chapter. The function `TooSimilar` rejects the new generated configuration if the distance between q_{near} and q_{near} in C can not exceed a predefined threshold.

4.4 Manhattan-like RRT Algorithm

This section presents a variant of the RRT algorithm that considers the active/passive partition of the configuration parameters introduced in Section 4.2. The algorithm, called Manhattan-like RRT (ML-RRT), computes the motion of the parts associated with both parameter types in a decoupled manner. We have called this variant Manhattan-like RRT (ML-RRT) because the paths computed by the algorithm look like Manhattan paths over these two sets of parameters that change alternatively. The ML-RRT algorithm considers two types of configuration parameters: *active* and *passive*. Active parameters are directly handled by the planner, while passive parameters are treated only when required to expand the tree. Indeed, passive parts only move if they hinder the motion of other mobile parts (active parts or other passive parts involved in the expansion).

The ML-RRT algorithm is schematized in Algorithm 2. At each iteration, the motion of active parts is computed first. The function `SampleConf` receives as argument the list of active parts P_{act} and it only samples the associated parameters q_{act}. Thus, this function generates a configuration q_{rand}^{act} in a sub-manifold

[2] In our implementation, l is a constant with default value equal to 10, and k is computed as $n_{\text{node}}/100$ rounded to the nearest upper integer, where n_{node} is the current number of nodes in the tree. These values have been empirically determined.

Algorithm 2: Construct_ML-RRT

input : the conformational space C;
the initial conformation q_{init};
the partition $\{P_{\text{act}}, P_{\text{pas}}\}$;

output : the tree τ;

begin

$\tau \leftarrow$ `InitTree`(q_{init});

while not `StopCondition`(τ) **do**

$q_{rand}^{act} \leftarrow$ `SampleConf`(C, P_{act});

$q_{near} \leftarrow$ `NearestNeighbor`(τ, q_{rand}^{act}, P_{act});

$(q_{new}, P_{\text{pas}}^{\text{col}}) \leftarrow$ `Expand`(q_{near},q_{rand}^{act});

while $P_{pas}^{col} \neq \emptyset$ **do**

$P_{\text{pas}}^{\text{mov}} \leftarrow$ `PartsToMove`($P_{\text{pas}}^{\text{col}}$);

$q_{rand}^{pas} \leftarrow$ `PerturbConf`(C, q_{new}, $P_{\text{pas}}^{\text{mov}}$, $q_{near}.n_{\text{fail}}$);

$(q'_{new}, P'^{\text{col}}_{\text{pas}}) \leftarrow$ `Expand`(q_{new}, q_{rand}^{pas});

$P_{\text{pas}}^{\text{col}} \leftarrow P'^{\text{col}}_{\text{pas}} \setminus P_{\text{pas}}^{\text{col}}$;

$q_{new} \leftarrow q'_{new}$;

if not `TooSimilar`(q_{near}, q_{new}) **then**

`AddNewNode`(τ, q_{new});

`AddNewEdge`(τ, q_{near}, q_{new});

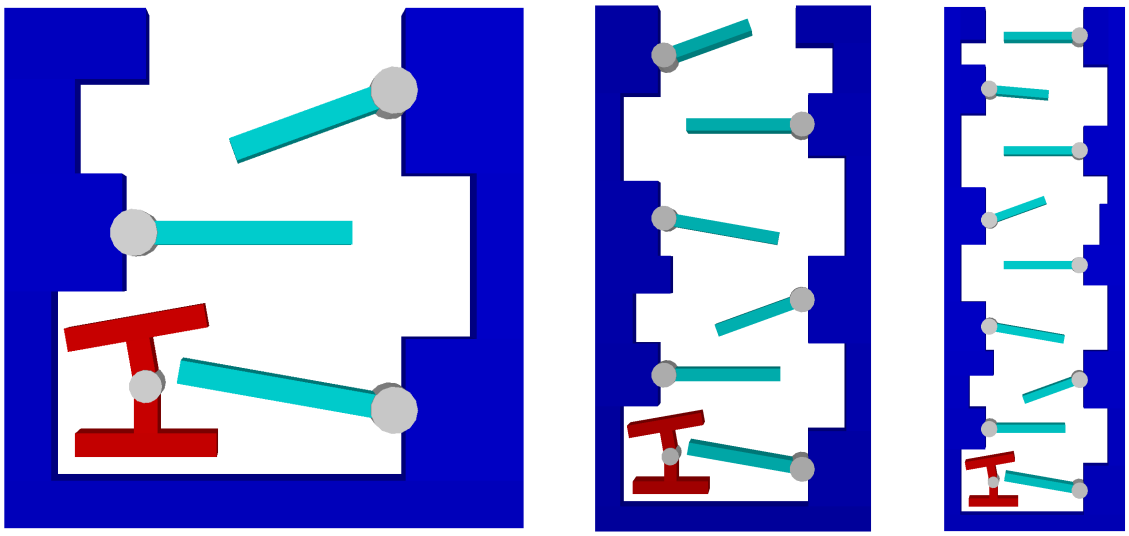
$q_{\text{near}}.n_{\text{fail}} \leftarrow 0$;

else $q_{\text{near}}.n_{\text{fail}} \leftarrow q_{\text{near}}.n_{\text{fail}} + 1$;

end

of the configuration space involving the active parameters, C^{act}. The function `NearestNeighbor` selects the node to be expanded q_{near} using a distance metric in C^{act}. Note that the function `NearestNeighbor` also integrates the basic improvements mentioned in Section 4.3. Then, `Expand` performs the expansion of the selected configuration by only changing the active parameters. A greedy strategy is used. The returned configuration q_{new} corresponds to the last valid point computed along the straight-line segment from q_{near} toward $\{q_{rand}^{act}, q_{near}^{pas}\}$. The function `Expand` also analyzes the collision pairs yielding the stop of the expansion process. If active parts in L_{act} collide with potentially mobile passive parts in P_{pas}, the list of the involved passive parts $P_{\text{pas}}^{\text{col}}$ is returned. This information is used in the second stage of the algorithm, which generates the motion of passive parts.

The function `PartsToMove` determines the list $P_{\text{pas}}^{\text{mov}}$ of passive parts to be moved at one iteration. This function receives as argument the list of colliding passive parts $P_{\text{pas}}^{\text{col}}$, and constructs a list with all the parts indirectly involved in the collision. The function `PerturbConf` generates a configuration q_{rand}^{pas} by randomly sampling the value of the passive parameters associated with $P_{\text{pas}}^{\text{mov}}$ in a ball around their configuration in q_{new}. An attempt is then made to further expand q_{new} toward $\{q_{new}^{act}, q_{rand}^{pas}\}$. Only the parts in $P_{\text{pas}}^{\text{col}}$ move during this tree expansion. The function `Expand` returns a list $P_{\text{pas}}^{\text{col}'}$ if the expansion is stopped by a collision involving passive parts. If this list contains new passive parts (in relation to $P_{\text{pas}}^{\text{col}}$), the process generating passive part motions is iterated. Such a possible cascade of passive part motions may be useful to solve problems where passive parts indirectly hinder the motion of the active ones because they block other passive parts. At the end of the iteration, if the expansion is not negligible, a new node and a new edge are added to the tree.



(a) (b) (c)

Figure 4.3: 2D examples: (a) Simple, (b) Double, (c) Triple.

Finally, note that although the active and passive parts move alternately in the path obtained by the ML-RRT algorithm, a randomized path smoothing post-processing[3] is performed in the composite configuration space of all the parameters, so that simultaneous motions are obtained in the final path.

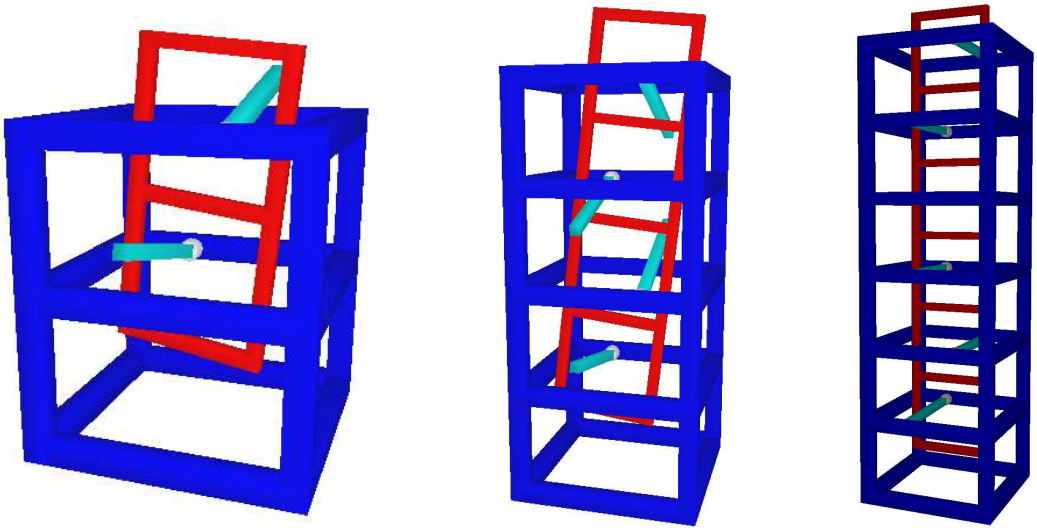
4.5 Empirical Performance Analysis

This section focuses on the performance comparison between the basic RRT algorithm and the ML-RRT variant which have both been implemented within the motion planning software Move3D [Siméon et al. 2001], and tested with a set of benchmark examples. The results reported in this section aim to illustrate both the generality and the good overall performance of the method. We start with the description of the models divided into two classes: two-dimensional and three-dimensional academic examples. Then, we analyze the intrinsic performance of the ML-RRT algorithm and we compare it with related methods. All reported results correspond to averaged values based on 10 runs of both methods. All the experiments of this chapter were performed on a AMD Opteron Processor 2222 3Ghz equipped with 8Gb of memory. The search process returned a failure if the tree size excessed 10000 nodes.

4.5.1 Benchmark Models

Two sets of two-dimensional and three-dimensional have been used. The two-dimensional benchmarks (see Fig. 4.3) involve a mobile H-like object with four d.o.f (i.e. two for translation, one for orientation, one for deformation) and several fixed sticks. The length of the static obstacle and the number of the fixed sticks are gradually multiplied to create more difficult problems (e.g., 3, 6 and 9 sticks along with more complex obstacles). The three-dimensional examples involve a freeflying object (6 d.o.f) constrained by a number of rotational sticks (see Fig. 4.4). The length of the mobile object is also extended along with the

[3] We use the *probabilistic path shortening* method [Sekhavat et al. 1998] for path smoothing.



(a) (b) (c)

Figure 4.4: 3D examples: (a) Simple, (b) Double, (c) Triple.

Table 4.1: NUMERICAL RESULTS.

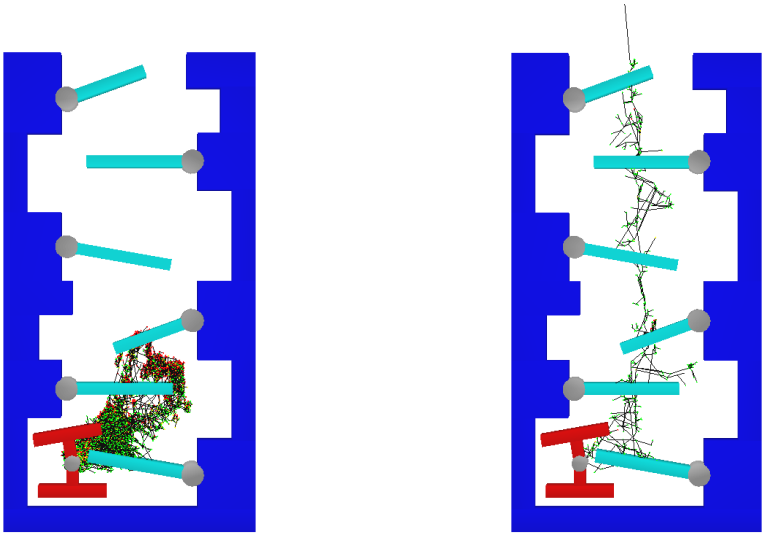
		2DSimple	2DDouble	2DTriple	3DSimple	3DDouble	3DTriple
	N_{DOF}	7	10	13	8	10	12
RRT	T(s)	48.64±17.28	$\rightarrow\infty$	$\rightarrow\infty$	0.35±0.36	29.36±25.82	$\rightarrow\infty$
	N_{node}	1266	$\rightarrow\infty$	$\rightarrow\infty$	94	465	$\rightarrow\infty$
	N_{samp}	22139	$\rightarrow\infty$	$\rightarrow\infty$	1390	42120	$\rightarrow\infty$
	N_{coll}	38882	$\rightarrow\infty$	$\rightarrow\infty$	6901	73768	$\rightarrow\infty$
MLRRT	T(s)	0.22±0.19	0.79±0.93	3.02±2.56	0.15±0.34	0.54±0.21	5.85±3.07
	N_{node}	262	444	1013	109	255	1365
	N^{act}_{samp}	883	1556	4537	179	1144	13077
	N_{coll}	3189	17110	22755	4552	11797	73145

static obstacle. In all the examples, the active parameters involve the translations and the orientations of the mobile object. The passive ones contain all rotational sticks and mobile object deformation joint (in the 2D models).

4.5.2 Performance Comparison

Table 4.1 displays the computing time (with standard deviation) and the number of nodes in the search trees, samples and collision tests required for solving all benchmark problems with both algorithms (i.e. RRT and ML-RRT). Note that, for ML-RRT, only the number of samples for active parameters N^{act}_{samp} is shown. These results show that ML-RRT clearly outperforms the basic RRT, and that the performance gain increases with the complexity of the problems. Note that the basic RRT is unable to solve the difficult versions of the problems (i.e. 2D-Double, 2D-Triple, 3D-Triple) in reasonable computing time, while the performance of ML-RRT is only slightly affected by the problem difficulty.

Figure 4.5 shows a projection of the search trees on the coordinates of the center of mass of the mobile



(a) (b)

Figure 4.5: Search tree development for 2D-Double problem obtained with: (a) the basic RRT algorithm, (b) the ML-RRT variant.

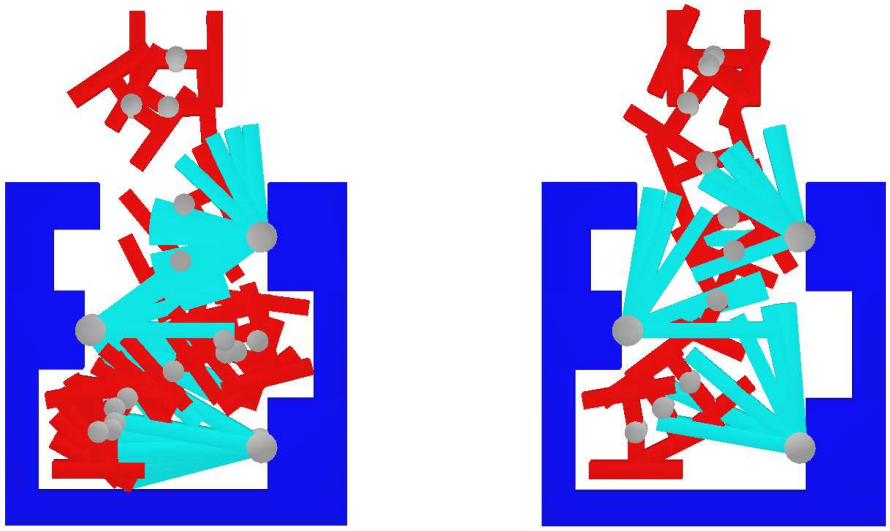
H-like object for example 2D-Double. The tree computed with the basic RRT algorithm contains 10000 nodes but all are concentrated in a small region of the search-space around the initial configuration. On the contrary, the tree obtained with the ML-RRT algorithm contains about 1000 nodes and yet better covers the search-space.

The main idea of the ML-RRT algorithm is to divide the configuration parameters into active and passive ones. The planner has to treat only the active parameters and a few passive ones if needed. When the number of passive parameters gradually increases along with more complex obstacles, this slightly affects the algorithm's performance. Results in Table 4.1 show that the performance of the basic RRT drastically degrades due to the complexity of problems. On the contrary, the ML-RRT method still maintains its performance with a relatively small increase of computing time.

Besides the computational efficiency, the solution paths obtained with both algorithms are also qualitatively different (see Fig. 4.6). The two traces of the solution paths obtained with the basic RRT and ML-RRT show this difference. In the path displayed by Fig. 4.6.a, computed by the basic RRT, the mobile H-like object circumvents the mobile parts of the fixed object by generating many small motions in the very constrained space. In the other path, obtained with ML-RRT (see Fig. 4.6.b), it is easy for the H-like robot to progressively "push" the passive rotational parts to exit. This type of solution path seems more natural for this kind of problem.

4.6 Multi-level passiveness

One extension of the ML-RRT algorithm is to take into account different mobility levels for the motion of different parts. Therefore, passive parameters will be moved according to their mobility. For example, Figure 4.7 illustrates the basic articulated disassembly problem (Fig. 4.7.a) and its extension (Fig. 4.7.b).



(a) (b)

Figure 4.6: Trace of solution paths for 2D-Simple problem obtained with: (a) the basic RRT algorithm, (b) the ML-RRT variant.

Instead of fixed obstacles in the basic example, the new extension involves two articulated walls with several associated sticks which are concatenated in a kinematic chain and attached to a fixed base. We suppose that the small sticks are easier to move than the walls.

The ML-RRT algorithm (see Algorithm 2) is general enough to permit such an extension. Only slight modifications of some functions are necessary. The function `PartsToMove` determines the list $P_{\text{pas}}^{\text{mov}}$ of passive parts to be moved at one iteration from the list of colliding passive parts $P_{\text{pas}}^{\text{col}}$. The extended version of this function determines the list of parts to move based on the kinematic diagram of the articulated mechanism. Figure 4.8 illustrates two typical situations. If the H-like object motion is hindered by a small sticks (Case 1 in the Figure 4.8), then, the list involves this stick and the articulated parts of the corresponding wall computed from the part containing the stick to the part attached to the base. When the H-like object is hindered by a part of the wall (Case 2 in the Figure 4.8), only the list of parts in the kinematic chain between this part and the base are returned.

A mobility coefficient $\delta \in (0,1]$ is assigned to each passive parameter. This coefficient is used to differentiate passive parts that are allowed to move easily (i.e. the sticks) from those that should be moved only if the solution path cannot be found otherwise (i.e. the walls). Then, the function `PerturbConf` modifies or not the configuration parameters of a passive part in $P_{\text{pas}}^{\text{mov}}$ depending on a probability distribution that considers its mobility coefficient δ and the difficulty for expanding q_{near}, which is estimated by the number of previous expansion failures n_{fail}. Therefore, a parameter is sampled if the following condition is satisfied:

$$NormalRand(\mu, \sigma^2) \geqslant 1 - \delta$$

Where `NormalRand` returns a random positive real number sampled from a normal distribution with mean $\mu = 0$ and variance $\sigma^2 = 0.1 \times n_{fail}$, and δ=(0,1] is the mobility coefficient of the part. Such a selection strategy maintains a low probability of moving parts with small mobility coefficient (e.g. big sticks) when

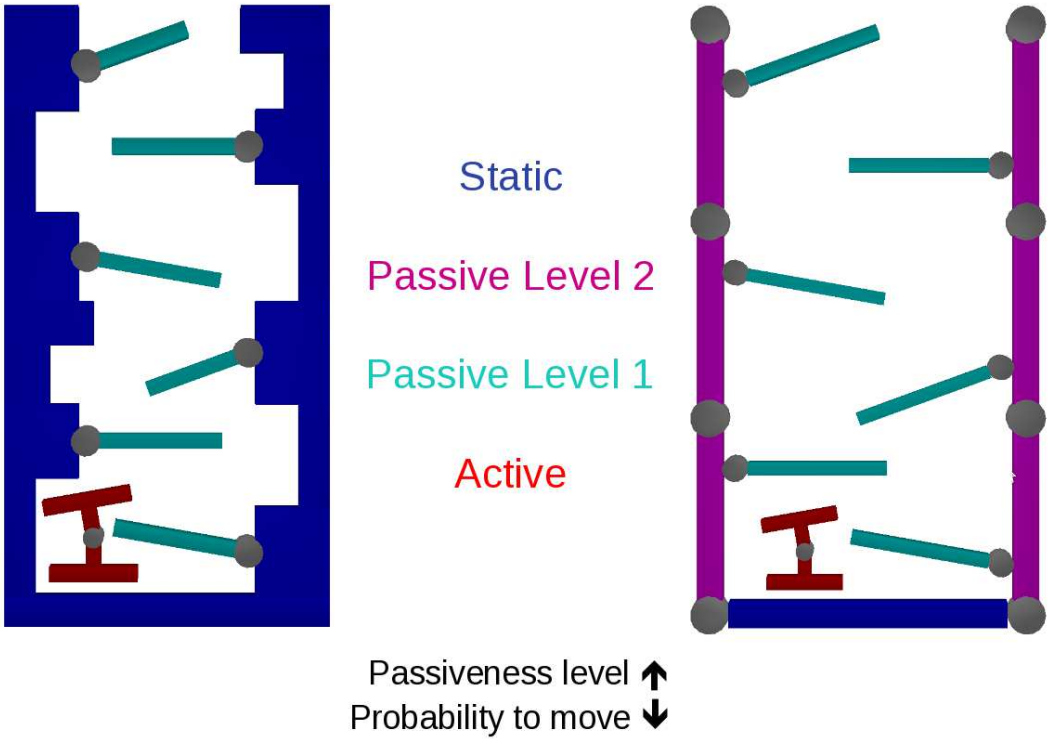


Figure 4.7: Illustration of two passive level problem in comparison with the basic problem. The bounding walls of the static object are now articulated. However, a lower probability to move with respect to the small sticks is assigned to them.

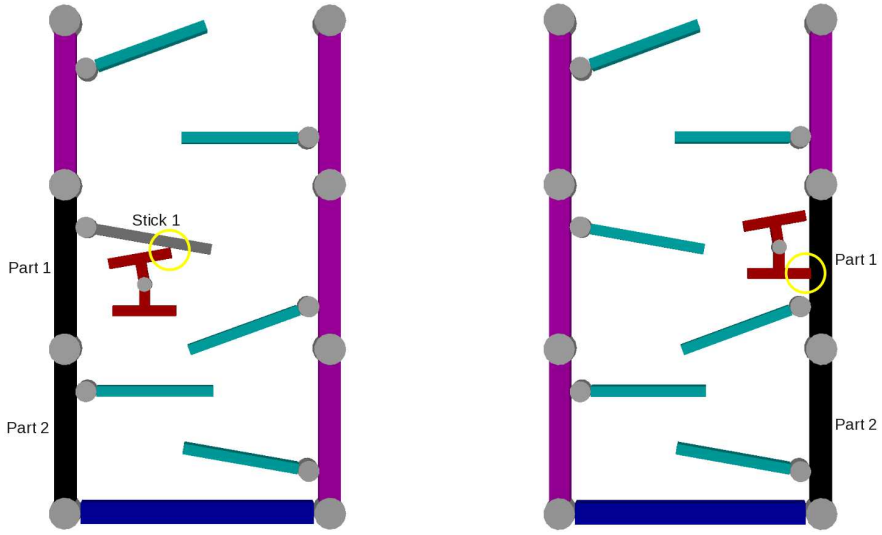


Figure 4.8: Determination of the list of passive parts to be moved $P_{\text{pas}}^{\text{mov}}$ based on the contacts with active parts and on the kinematic diagram of the protein model. Two typical cases are illustrated: (a) Case 1: $P_{\text{pas}}^{\text{mov}}$ = (Stick 1, Part 1, Part 2), (b) Case 2: $P_{\text{pas}}^{\text{mov}}$ = (Part 1, Part 2).

Table 4.2: Influence of passive priorities.

Mob.Coef(δ)	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Ave.Time(s)	4.39	3.42	4.55	5.0	6.1	8.67	10.05	12.13	11.94	13.72
Std.Dev.	4.67	2.57	4.36	4.83	8.2	13.3	10.89	10.1	13.58	12.77
N_{node}	470	438	502	544	552	763	860	964	896	1058
N_{samp}^{act}	1926	1774	2250	2301	2350	3152	3398	3840	3677	4203
N_{coll}	38148	34599	40238	42277	43017	55881	61475	66372	62817	70165

the diffusion tree easily grows, while the probability is increased when required to unblock the exploration.

4.6.1 Influence of passive parameter priorities

In order to analyze the influence of passive parameter priorities, we have performed 10 tests with the example *TwoPass* in Fig.4.7 in which the mobility coefficient of small sticks is always set at 1.0 (called the first level of passivity) while for the walls, it varies from 0.1 to 1.0 (called the second level of passivity). The highest mobility coefficient of the walls corresponds to the basic ML-RRT algorithm's treatment. Table 4.2 shows the computing times (in seconds) with their standard deviations, and the number of nodes, samples for active parameters, collision tests required by the planner. All values are averaged over 10 successful runs for each test. Note that, a distance constant was considered between the ends of the two articulated walls in order to increase the difficulty of the problem. The results show that when the mobility of the walls (i.e. the second level of passivity) increases, the planner gradually spends more time for solving the problem. The best result involves the second passive level mobility set at 0.2 while the worst result corresponds to the basic ML-RRT algorithm's performance (i.e 1.0). Because the limits in the movement of articulated walls allow the extended algorithm to treat less passive parameters at one iteration than the basic ML-RRT, therefore the problem can be solved more quickly.

4.6.2 Influence of number of passive levels

The analysis in this sub-section focuses on the influence of number of passive levels. The example illustrated in Figure 4.9 called *LongPass* displays a more difficult problem involving two five-joint walls which contain five rotational small sticks. Hence, this problem totally involves 24 d.o.f. Two bounded distance constraints (i.e. one between two extremities and the other between the middle points of the articulated walls) are also added to increase the problem's difficulty. The mobility coefficient for the small sticks is always set at 1.0, and 0.2 for the walls. Another example *ThreePass* illustrated in the Fig. 4.10 treats a problem with three levels of passiveness. 22 d.o.f are considered in this model. The smallest yellow sticks have the highest mobility coefficient (i.e. 1.0) and the mobilities for the cyan sticks and the pink walls are set at 0.5 and 1.0, respectively. The bounded distance constraint is again applied in this three-passive-level model.

Table 4.3 shows the performance comparison between basic and multi-passive ML-RRT algorithms. The results are also based on 10 successful runs for each examples. In the *LongPass* model, the computing time of the extended variant can be reduced by 2 in comparison to basic ML-RRT. In the other model (i.e *ThreePass*), the gain can be upto 7. We can conclude that the different priorities for passive parameter

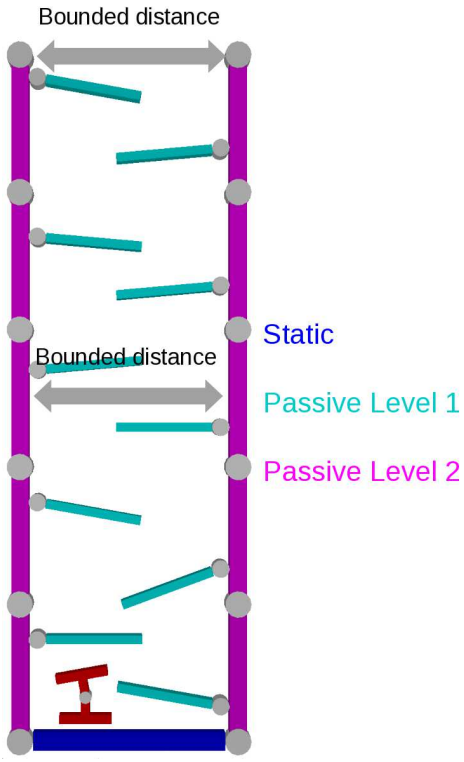


Figure 4.9: Illustration of larger number of passive parameters. Two different passiveness levels are considered for the small sticks and the walls.

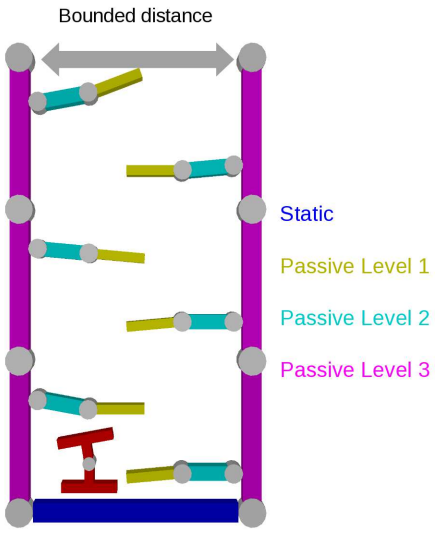
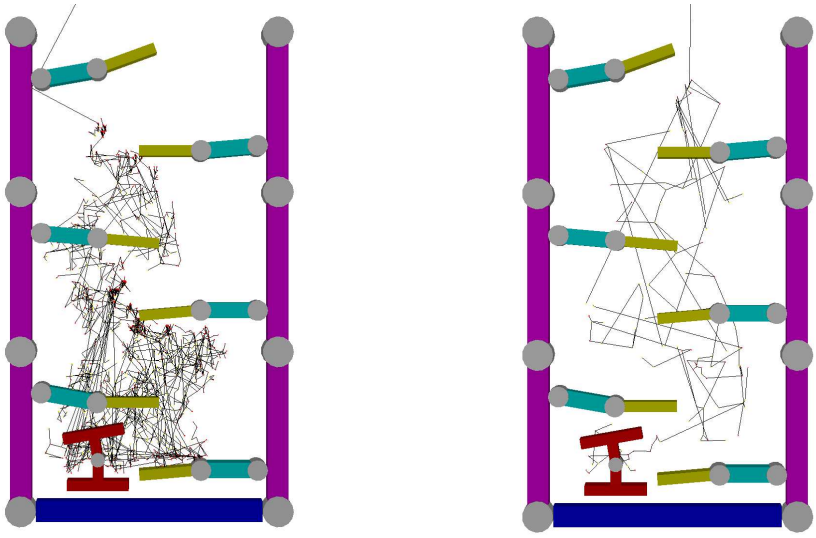


Figure 4.10: Illustration of three passive levels. Three different passiveness levels are considered for the yellow and cyan sticks, and the big walls.



(a) (b)

Figure 4.11: Development of search trees for ThreePass problem obtained with: (a) Basic ML-RRT, (b) Multi-Passive Variant.

Table 4.3: Influence of number of passive levels.

Example	**Performance**	**Basic ML-RRT**	**Multi-passive ML-RRT**
LongPass	Ave. Time(s)	1047.34±275.31	535.42±335.78
	N_{node}	9372	6769
	N_{samp}^{act}	55355	35271
	N_{col}	729434	481254
ThreePass	Ave. Time(s)	200.14±282.15	29.63±44.41
	N_{node}	3023	1141
	N_{samp}^{act}	25484	8149
	N_{col}	235148	84975

movement facilitates the exploration process of the planner, which allows finding the solution faster. Figure 4.11 shows a projection of the search trees on the coordinates of the center of mass of the mobile H-like object for *ThreePass* model. The tree computed with the basic ML-RRT algorithm contains about 3000 nodes but all are concentrated in a small region of the search-space around the initial configuration. On the contrary, the tree obtained with the Multi-Passive Level variant contains about 1000 nodes and yet better covers the search-space.

4.7 Pushing and Pulling Motions

The basic ML-RRT algorithm performs well when treating problems needed only "pushing" motion of passive parts by the mobile object. Another possible extension is to consider "pulling" motions which may be important in some classes of disassembly problems. Therefore, we introduce a new notion called *Passive*

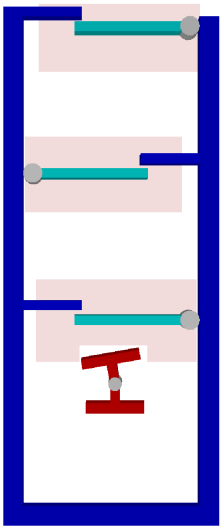


Figure 4.12: Illustration of Passive Zone of the bodies associated with passive parameters.

Zone (see Fig. 4.12). For each body associated with a passive parameter of the static object, a passive zone with predefined radius R is computed. Hence, in the diffusion process, if the mobile H-like object enters in a passive zone, the planner will recognize the corresponding passive part and then "push" it or "pull" it.

In order to detect "collision" between the mobile object and a passive zone for a given configuration, we compute all distances between the mobile object's bodies (i.e. the H-like object has two parts) and the other bodies associated with passive parameters (i.e. the small rotational sticks). If the difference between these distances and their corresponding radius are negative or zero (meaning that there is "collision"), a list of passive parameters is then returned (i.e. P_{col}^{pas}).

Algorithm 3 shows the pseudo-code of the "Pushing-Pulling" ML-RRT variant, `Construct_PushPull-ML-RRT`. Overall, this algorithm is an inversed ML-RRT (see Algorithm 2). It tries to expand several passive parameters before active parameters. For a given q_{near}, the function `PassZoneCol` looks for the passive parameters close to the mobile object and adds them to the list P_{col}^{pas}. A loop is therefore iterated to update the pose of several passive parameters involved in the "passive zone contact" list with the help of the function `PerturbConf` which generates a configuration q_{rand}^{pas} by randomly sampling the value of the passive parameters associated with P_{col}^{pas} in a ball around their configuration in q_{near} with a hope that these nearest passive parameters will not hinder the expansion of the active ones. Once a collision-free pose of passive parameters have been successfully sampled, it is ready for the planner to expand from the new q_{near} (q'$_{near}$ in Algorithm 3) toward $\{q_{rand}^{act}, q_{near}^{pas}\}$. The rest of the pseudo-code is similar to the basic ML-RRT.

4.7.1 Influence of passive zone size

In order to analyze the influence of the passive zone size, 10 tests in which the passive zone radius is varied were performed with the model illustrated in the Fig. 4.12. This radius can be defined by the formula:

Algorithm 3: Construct_PushPull-ML-RRT

input : the conformational space C;
the initial conformation q_{init};
the partition $\{P_{\text{act}}, P_{\text{pas}}\}$;

output : the tree τ;

begin

$\tau \leftarrow$ `InitTree`(q_{init});
while not `StopCondition`(τ) **do**
$q_{rand}^{act} \leftarrow$ `SampleConf`(C, P_{act});
$q_{near} \leftarrow$ `NearestNeighbor`(τ, q_{rand}^{act}, P_{act});
$P_{\text{pas}}^{\text{col}} \leftarrow$ `PassZoneCol`(q_{near});
$q'_{near} \leftarrow q_{near}$;
while $P_{pas}^{col} \neq \emptyset$ **do**
$P_{\text{pas}}^{\text{mov}} \leftarrow$ `PartsToMove`($P_{\text{pas}}^{\text{col}}$);
$q_{rand}^{pas} \leftarrow$ `PerturbConf`(C, q'_{near}, $P_{\text{pas}}^{\text{mov}}$, $q'_{near}.n_{\text{fail}}$);
$(q_{new}, P'^{\text{col}}_{\text{pas}}) \leftarrow$ `Expand`(q'_{near}, q_{rand}^{pas});
$P_{\text{pas}}^{\text{col}} \leftarrow P'^{\text{col}}_{\text{pas}} \setminus P_{\text{pas}}^{\text{col}}$;
$q'_{near} \leftarrow q_{new}$;
$(q_{new}, P_{\text{pas}}^{\text{col}}) \leftarrow$ `Expand`(q'_{near},q_{rand}^{act});
if not `TooSimilar`(q_{near}, q_{new}) **then**
`AddNewNode`(τ, q_{new});
`AddNewEdge`(τ, q_{near}, q_{new});
$q_{\text{near}}.n_{\text{fail}} \leftarrow 0$;
else $q_{\text{near}}.n_{\text{fail}} \leftarrow q_{\text{near}}.n_{\text{fail}} + 1$;

end

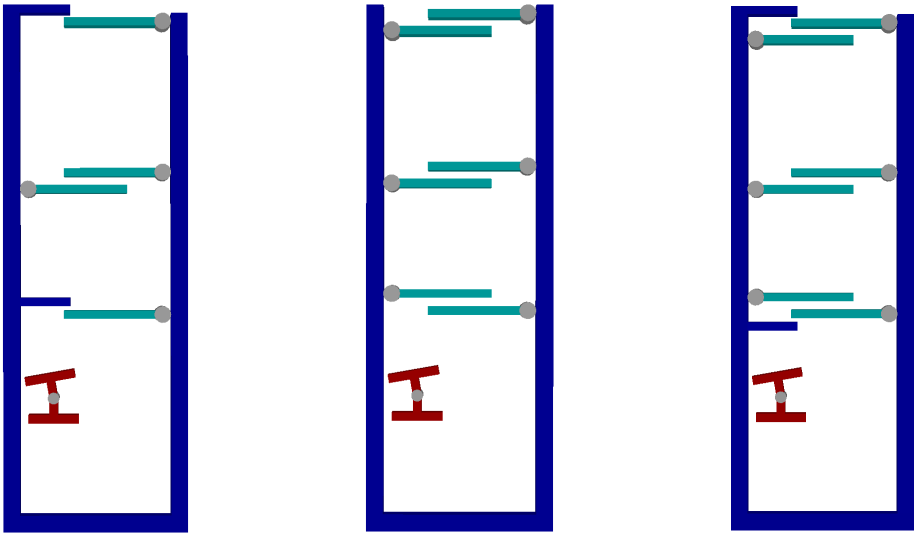
Table 4.4: Influence of passive zone radius.

K	10	20	30	40	50	60	70	80	90	100	MLRRT
Ave. Time(s)	0.41	0.49	0.53	0.62	0.58	0.69	0.67	0.70	0.71	0.83	0.61
Std. Deviation	0.28	0.41	0.42	0.54	0.35	0.57	0.52	0.53	0.51	0.92	0.56
N_{node}	336	366	383	454	415	464	467	473	485	495	296
N_{samp}^{act}	469	500	631	648	644	662	653	675	739	796	930
N_{coll}	5049	5619	5655	6315	6629	6652	6643	7515	7812	8089	5668

$$R = K \times \varepsilon \tag{4.1}$$

where K is a parameter (e.g., 10, 20, 30, etc) and ε is the smallest interpolation step used by the planner.

Table 4.4 shows the computing times (in seconds) with their standard deviations and the number of nodes, samples and collision tests required by the planner calculated from 20 successful runs. The value of K parameter varies from 10 to 100. The results shows that this parameter slightly affects the performance of the algorithm. The bigger the passive zone radius is, the more passive parameters are considered in "pulling" phase. Therefore, the planner has to treat more passive parts than in the basic ML-RRT algorithm while a big radius is set . Results show that, although the basic ML-RRT works with "pulling" problems, the new variant performs better with a reasonable value of passive-zone radius R (i.e. small K) (see column K=10



(a) (b) (c)

Figure 4.13: Illustration of "pushing and pulling" models with different number of passive parameters: (a) Pulling2, (b) Pulling3, (c) Pulling4.

and the last column corresponding to the basic ML-RRT).

4.7.2 Influence of passive parameter number

The goal of this section is to compare the performance difference between the basic ML-RRT and the "Pushing-Pulling" variant when the number of passive parameters is increased as well as the complexity of obstacles. Three new examples were designed from the one in the Fig. 4.12 where the number of passive parameters is gradually increased along with more complex static obstacles (see Fig. 4.13). Twenty successful runs are performed for each model where the K parameter is set at 10.

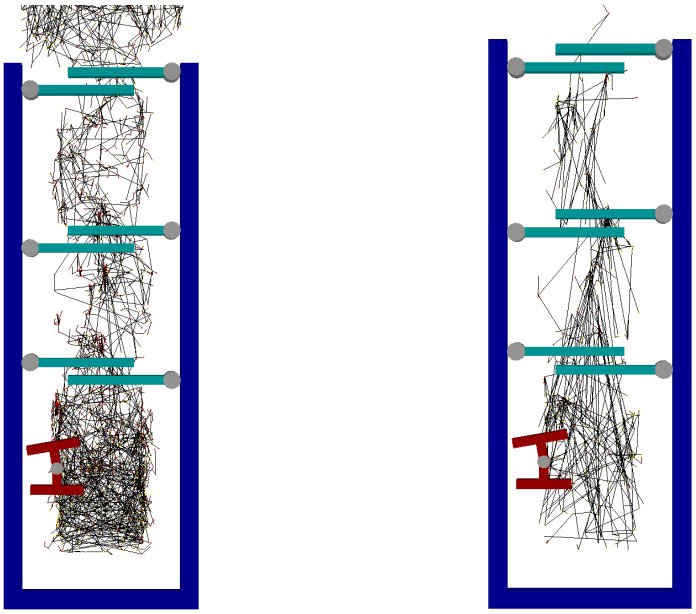
Table 4.5 shows that the performance of the basic ML-RRT is very similar to that of the "Pushing-Pulling" variant for the easier five-passive-joint model (i.e. *Pulling*2 model). However, the performance of the basic ML-RRT algorithm significantly degrades when the complexity of the problem increases. On the other hand, the new extended variant is able to solve the more difficult problems in reasonable computing time. Figure 4.14 shows the graphs constructed by the two methods for the example Pulling3. The basic ML-RRT's one is much more dense than the new variant's one. Therefore, both "pushing" and "pulling" motions are highly required to treat the most constrained problems. This combination of both motions allows the new extended planner to discover faster the key states of the configuration-space that lead to the exit of the mobile object.

4.8 Discussion

The basic ML-RRT algorithm described in this chapter is an efficient method for disassembly path planning of two objects with articulated parts. An interesting feature of the algorithm is its ability to treat problems with a high number of potentially mobile parts and to automatically identify the degrees of freedom that are

Table 4.5: Influence of number of passive parameters.

Example	Performance	Basic ML-RRT	Pushing-Pulling ML-RRT Variant
Pulling2	Ave. Time(s)	5.71±7.61	4.24±7.16
	N_{node}	911	1078
	N^{act}_{samp}	3179	2910
	N_{col}	20995	22299
Pulling3	Ave. Time(s)	51.97±83.2	22.11±43.68
	N_{node}	3390	2275
	N^{act}_{samp}	7907	5962
	N_{col}	119071	92286
Pulling4	Ave. Time(s)	$\rightarrow \infty$	454.6±343.13
	N_{node}	$\rightarrow \infty$	11856
	N^{act}_{samp}	$\rightarrow \infty$	36105
	N_{col}	$\rightarrow \infty$	273081



(a) (b)

Figure 4.14: Development of search trees for Pulling3 problem obtained with: (a) Basic ML-RRT, (b) "Pushing-Pulling" Variant.

important for the disassembly task. In this thesis, we show the interest of ML-RRT for solving problems in structural biology. Nevertheless, other application domains, such as CAD/PLM, could also benefit from the proposed method.

Two extensions of the ML-RRT algorithm have been also presented. The first extension addresses disassembly planning problems for articulated objects involving parts with different mobilities. Different mobility coefficients are assigned to passive parts, which allows the planner to better explore the

configuration-space. Therefore, the computational efficiency is significantly improved. This extension is applied for studying the conformational changes induced by the ligand exit when taking into account the different levels of flexibility of different protein's parts such as side chains, loops or domains, etc. This application will be detailed in the Chapter 5.

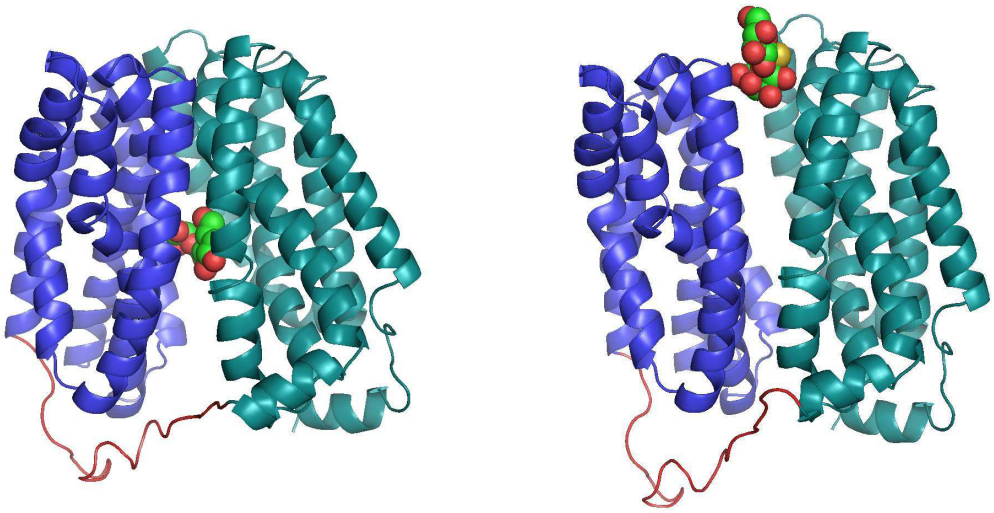
The second extended version of ML-RRT is devised for solving problems in which passive articulated parts are "pushed" and also "pulled" by the mobile object, which is important in some classes of disassembly problems. This extension will also help to solve more efficiently the problems addressed in next chapters.

5

Ligand-Protein Interaction Studies

Proteins are flexible macromolecules that fluctuate between nearly isoenergetic folded states [Frauenfelder et al. 1991]. In many cases, conformational changes are associated with their function, and they occur through the interaction with other molecules. For instance, conformational changes are of major importance for protein-ligand and protein-protein recognition [Carlson 2002; Lensink and Méndez 2008]. This chapter addresses protein conformational changes induced (or required) by the diffusion of a ligand (or substrate/product) molecule inside the protein. An illustrative example is the permeation of lactose through a membrane transport protein (LacY) [Kaback et al. 2001]. LacY fluctuates between a conformation where lactose is accessible from the cytoplasm, but the channel toward the periplasmic side is closed (Figure 5.1.a), and the opposite conformation where the channel is open toward the periplasm and closed in the cytoplasmic side (Figure 5.1.b). The transition between these two conformational states occurs during lactose diffusion inside the protein.

Despite impressive recent advances on the structural determination of protein motions [Katona et al. 2007; Schanda et al. 2007], currently available experimental methods are unable to provide an atomic-resolution structural description of protein conformational changes associated with ligand diffusion. Computational methods are therefore necessary to better understand such processes. However, the time-scale of the ligand diffusion process from a deep active site to the protein surface is out of range for standard molecular dynamics (MD) simulations. Variants of MD methods such as steered molecular dynamics (SMD) [Izrailev et al. 1998] and random acceleration molecular dynamics (RAMD) [Ludemann et al. 2000] have been proposed for accelerating the simulation of the ligand exit. Both methods introduce an artificial force in the molecular force field to enhance the ligand motion in a given direction. In SMD simulations, this direction is usually defined by the user through a haptic device. In RAMD simulations, the direction is randomly chosen and iteratively modified after a given number of simulation steps if the ligand gets



(a) (b)

Figure 5.1: Lactose permease (LacY) conformational transition. a) The crystal structure [Abramson 2003] (PDB ID 1PV7), where the substrate is accessible from the cytoplasm. b) Model of LacY after the conformational changes induced by the substrate diffusion toward the periplasm.

stuck. Although these methods have been shown to provide biologically relevant information, they remain computationally expensive. Besides, the artificial force introduced for accelerating the simulation may yield biased results about the induced conformational changes, so that the interest of simulating with an accurate molecular force field is partially lost.

This chapter presents an alternative method for simulating ligand diffusion motions, together with the possibly induced conformational changes of the protein. The method applies the ML-RRT algorithm presented in Chapter **4**. Given an initial structure with the ligand docked inside the protein, the proposed method computes a path (i.e. continuous sequence of conformations) simulating the ligand exit. Such a path search problem is formulated as a mechanical disassembly problem (Sec. 5.1), where the protein and the ligand are modeled as articulated mechanisms. The main feature of this method is its computational efficiency, enabling to compute large-amplitude conformation transition paths, such as the one illustrated in Figure 5.1, in the range of less than one hour of CPU time. Results illustrating the capacities of the method are presented on three biologically interesting systems involving ligand-induced conformational changes: amylosucarse, lactose permease (LacY), and the β_2-adrenergic receptor (Sec. 5.2). The last section 5.3 discusses on results and point out possible extensions of the method.

5.1 Methodology

5.1.1 Model and Parameters

Mechanistic molecular model:

A molecule contains a set of atoms A_i connected by bonds. Generally, the atom's position can be represented by a point and the bond connecting two atoms by a straight-line segment. Therefore, a sequence of bonded

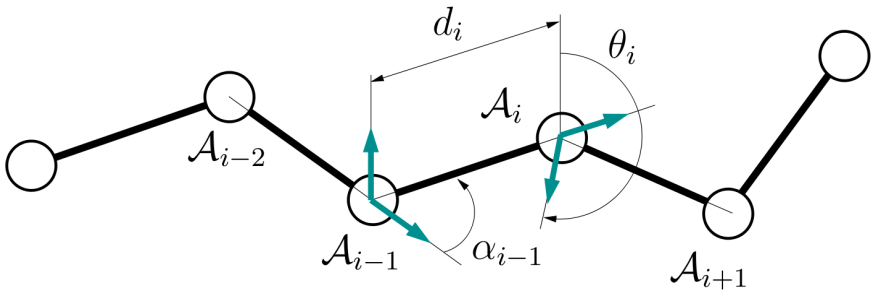


Figure 5.2: Molecular chain and the internal coordinates.

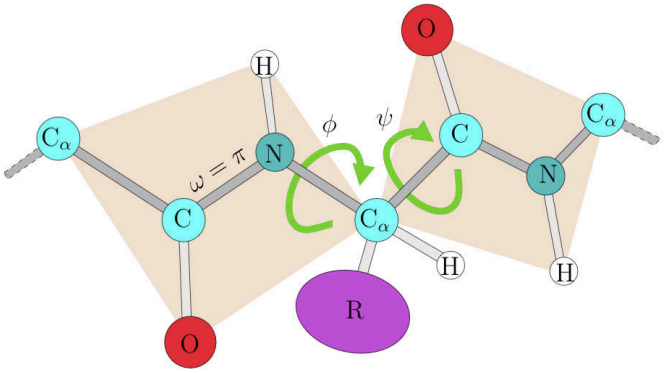


Figure 5.3: Model of two-peptide-unit segment.

atoms is called a *molecular chain*. In order to describe the relative position of consecutive atoms in this molecular chain, three parameters, called *internal coordinates*, are used: *bond lengths* d_i, *bond angles* α_i and *dihedral angles* θ_i. The internal coordinates are illustrated in Figure 5.2. According to the rigidity of bond lengths and bond angles, their values (d_i,α_i) are normally fixed. We have applied the modified-Denavit-Hartenberg (mDH) convention [Craig 1989] to model a molecular chain from the internal coordinates. Therefore, a molecular chain between atoms (A_i) can be then modeled as a kinematic chain in which joint variables correspond to dihedral angles. The conformation of the chain is represented by the array q of the θ_i.

This thesis, mainly deals with proteins, which involve one or several polypeptide chains. A polypeptide chain is a sequence of amino-acids (also called residues) which 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 Fig. 5.3) specific to each particular amino-acid. 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, called a *peptide bond*, has a double-bond character making it particularly rigid. 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). 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). Generally, in the fully flexible representation, the considered articulations are the rotations between rigid atom groups are all the ϕ, ψ, and ω for the

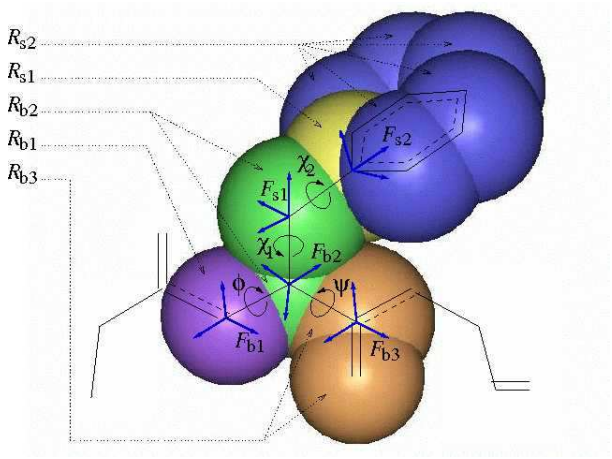


Figure 5.4: Mechanical model of an amino acid residue (phenylalanine) with all-atom fully flexible representation.

backbones and χ_i for the side-chains. Therefore, the conformation can be determined by an array:

$$q = \{q_{bkb}, q_{sch_1}, ..., q_{sch_n}\} \tag{5.1}$$

where q_{bkb} is the backbone conformation and q_{sch_i} is the conformation of each side-chain. The atom groups that correspond with the rigid bodies of the articulated molecular model are represented by spheres with a percentage of Val der Waal (VdW) radii. Figure 5.4 shows the mechanical model of phenylalanine residue which is composed of five rigid bodies divided into two classes:

- backbone rigids: $R_{b1} = \{N\}$, $R_{b2} = \{C_\alpha, C_\beta\}$, $R_{b3} = \{C, 0\}$
- side-chain rigids: $R_{s1} = \{C_\gamma\}$, $R_{s2} = \{C_{\delta 1}, C_{\delta 2}, C_{\varepsilon 1}, C_{\varepsilon 2}, C_\zeta\}$

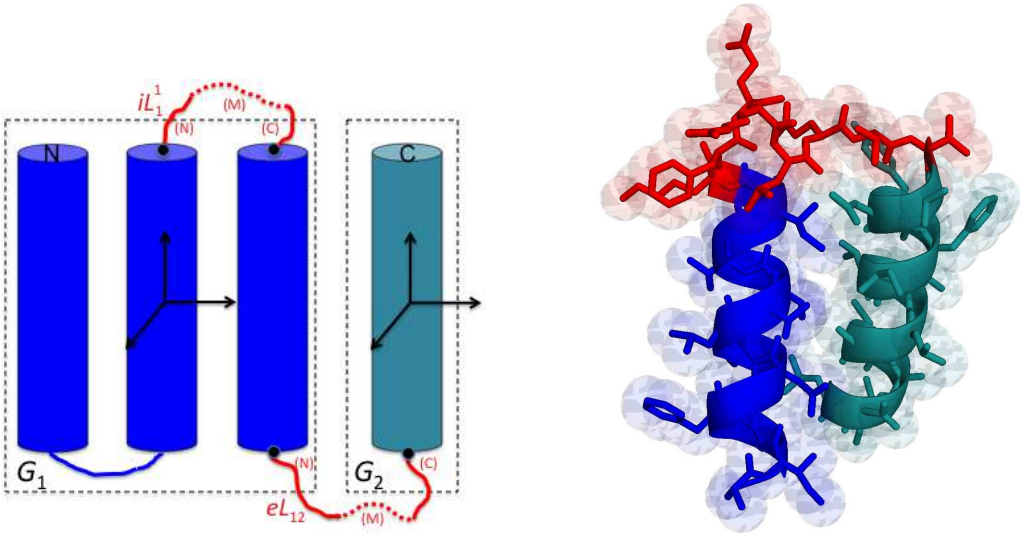
Note that, although in the biochemical description of proteins C_β is a side-chain atom, it belongs to a backbone rigid in our mechanistic model, since its position is determined by the values of the backbone joint torsions, but it does not depend on the side-chain conformation.

Definition of rigid and flexible segments:

The method described in this chapter deals with partially-flexible protein models to treat protein-ligand disassembly problems. Therefore, the size of the atom groups depends on the level of flexibility allowed to different parts of the molecule. Flexible and rigid regions can be assigned based on structural knowledge. In the present work, flexibility is defined by the user. Note however that the identification of rigid and flexible regions may be automated using computational methods such as FIRST [Wells et al. 2005].

Figure 5.5 illustrates the mechanistic model of a protein. The following notation is used:

- G_i group: set of rigid secondary structure elements (with flexible side-chains), possibly connected by flexible loops.
- iL_i^k intra-group loop: k^{th} flexible segment between two secondary structure elements of group G_i.
- $eL_{i,i+1}$ inter-group loop/linker: flexible segment between secondary structure elements in consecutive groups G_i and G_{i+1}.



(a) (b)

Figure 5.5: Schematic representation of a flexible protein model: (a) The three secondary structure elements grouped in G_1 are modeled as a rigid solid. The group G_2 involves only one secondary structure element. The loop/linker $eL_{1,2}$, between G_1 and G_2, is flexible. Loops connecting elements in a group can be flexible or not. Only the intra-domain loop iL_1^1 is flexible in this example. (b) Illustration of the mapping between the mechanistic model and hierarchical model. Two helices are considered as big rigid groups, while the flexible loop and all side-chains are composed of small atom-groups connected by rotatable bonds.

Each group G_i holds free rigid body mobility, independently from the other groups. Therefore, loop-closure constraints have to be imposed on flexible segments $eL_{i,i+1}$ and $eL_{i-1,i}$ connecting G_i to its neighboring groups, in order to maintain the molecular chain integrity. As indicated in Figure 5.5, several parts are differentiated inside inter- or intra-group loops: the N-terminal and C-terminal segments, and the middle part (M), which is composed by a tripeptide. Such a decomposition is required for the treatment of loop motions that will be explained below. Additionally, geometric (distance and orientation) constraints can be introduced between any pair of elements (rigid groups or loops) in order to model interactions such as hydrogen bonds or disulfide bonds. All these constraints must be satisfied during the conformational exploration. Side-chains (not represented in the figure) are generally modeled as flexible elements with freely rotatable bond torsions (i.e., [-π, π]) but they can also be defined as rigid parts together with the secondary structure elements. By default, the ligand is also fully flexible. Nevertheless, the user can arbitrarily define the flexibility of the ligand and the side-chains.

Conformational parameters:

The protein conformation is defined by the parameters determining the pose (position and orientation) of all the groups G_i, the values of the bond torsions in intra- and inter-group loops, and the bond torsions of the side-chains. If some flexibility is allowed to secondary structure elements, the associated variables are also part of the conformational parameters. The conformational parameters of the ligand have the six parameters defining the pose of its reference frame (associated with its center of mass) as well as the values of the allowed bond torsions.

Let q denote the array containing the values of all the conformational parameters of the protein and

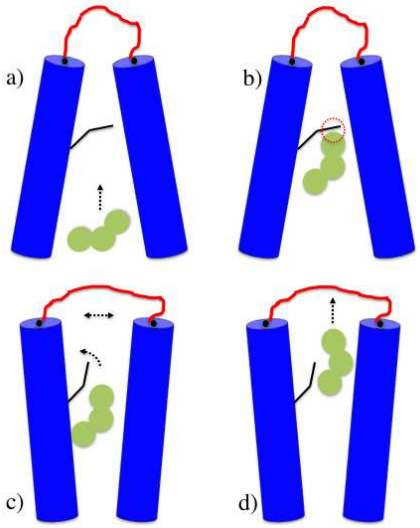


Figure 5.6: Illustration of the decoupled exploration of active and passive parameters within ML-RRT. a) Expansion of active parameters corresponding to the motion of the ligand. b) Identification of the passive parts hindering the ligand motion. c) The expansion of passive parameters yielding the opening motion of the protein. d) New iteration of the active parameters expansion.

the ligand. The ML-RRT algorithm explores the composite conformational space C, which is the set of all conformations q involving the parameters of the mechanistic models of the two molecules described in the previous subsection. As mentioned in Chapter 4, ML-RRT considers the conformational parameters at different levels of priority. Indeed, the parameters are partitioned into *active* and *passive* on the basis of their role in the disassembly problem. Active parameters are essential for carrying out the disassembly task, while passive parameters only need to move if they hinder the progress of the process.

Thus, the mobile parts of the molecular model are separated into two lists P_{act} and P_{pas} containing the active and the passive parts respectively. For a given partition, the conformational parameters are separated into two sets: $q = \{q^{act}, q^{pas}\}$, where q^{act} is the set of conformational parameters associated with the parts in P_{act} and q^{pas} is the set associated with P_{pas}. For the protein-ligand disassembly problems addressed in this chapter, q^{act} involves the ligand parameters, while q^{pas} concerns the protein flexibility.

Additionally, a mobility coefficient $\delta \in (0,1]$ is assigned to each passive parameter. This coefficient is used to differentiate passive parts that are allowed to move easily from those that should be moved only if the solution path cannot be found otherwise. By default, the mobility coefficient of all side-chains is set to 1, meaning that they will systematically move if they are identified during the exploration. Lower mobility coefficient is allowed to loops and secondary structure groups, with $\delta = 0.5$ and $\delta = 0.2$ respectively in the current implementation. This distribution of mobility coefficient is coherent with the fact that side-chains, small groups of atoms, are more flexible than loops and secondary structure groups.

5.1.2 Conformational exploration algorithm

The problem of computing the exit path of a ligand from a protein active site can be formulated as a mechanical disassembly problem in which molecules are represented as articulated mechanisms. The degrees of freedom (DOF) of the molecular models correspond to bond torsion (backbone or side-chains)

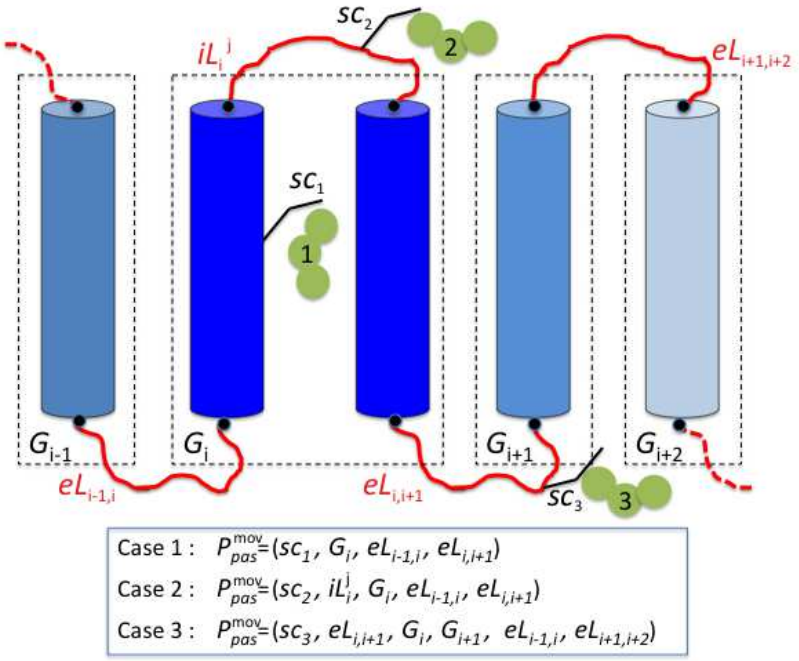


Figure 5.7: Determination of the the list of passive parts to be moved $P_{\text{pas}}^{\text{mov}}$ based on the contacts with active parts and on the kinematic diagram of the protein model. Three typical cases are illustrated.

and to rigid-body motion of atoms groups (rigid secondary structure elements). Starting from a given "assembled" (docked) position of the ligand inside the protein, the disassembly problem consists in finding the path (i.e. continuous sequence of conformations) leading to a "disassembled" state, where the ligand is located outside the protein. The disassembly path has to be searched in a composite conformational space involving the degrees of freedom of the protein and the ligand. The difficulty for solving such path search problem is due to the very high dimension of this search-space (e.g. more than a thousand of DOF). The ML-RRT algorithm, which is detailed in Chapter 4, computes motions of the ligand and the protein (associated with active and passive parameters, respectively) in a decoupled manner. Figure 5.6 provides a simple illustration of the process, which alternates expansion attempts of these parameter subsets. Active parameters are directly handled by the planner, while passive ones are treated only when required to further expand the tree. Indeed, passive parts only move if they hinder the motion of other mobile parts (active parts or other passive parts involved in the expansion). Figure 5.7 illustrates three typical situations. If the ligand motion is hindered by a side-chain in a secondary structure element (Case 1), then the list of passive parts to move involves this side-chain and the corresponding group G_i. When the colliding side-chain is on a flexible loop, then the list involves the side-chain, the loop backbone, and the group G_i for an intra-group loop iL_i (Case 2), or the groups G_i and G_{i+1} for an inter-group loop $eL_{i,i+1}$ (Case 3). In all the cases, when a group G_i is considered for subsequent motion, then the backbone of inter-group loops $eL_{i-1,i}$ and $eL_{i,i+1}$ (if any) is also considered into the list, since the conformation of these loops needs to be sampled together with the group pose in order to maintain the chain integrity.

5.1.3 Geometric constraints verification

During the conformational exploration, a set of geometric constraints has to be checked (e.g., collision avoidance, hydrogen/disulfide bond integrity) or reinforced (e.g., loop closure). These constraints are explained below.

Collision avoidance:

The main geometric constraint to be verified during the conformational exploration is the avoidance of atom overlaps. The atoms are represented by rigid spheres with a percentage of van der Waals radii. Considering a percentage of the van der Waals equilibrium distance ensures that only energetically infeasible conformations are rejected by the collision checker. The value of 80% is often used in techniques that geometrically check atom overlaps [DePristo et al. 2003]. Collisions are checked between the ligand and the protein, as well as internal collisions between mobile parts of each molecule. The collision test is done inside the function `Expand` (see Algorithm 1 and 2 in Chapter 4), which performs the local expansion motion. Our implementation builds on the efficient BioCD algorithm [Ruiz de Angulo et al. 2005], specially designed for articulated molecular models. BioCD uses hierarchical data structures to approximate the shape of the molecules at successive levels of detail, making the number of atom pairs tested for collision to be significantly reduced. BioCD is inspired by the dual $kd-tree$ traversal algorithms initially developed for n-point correlation problems in statistical learning. The algorithm maintains two levels of bounding volume hierarchies grouped according to spatial proximity. The first level organizes the rigid parts of the articulated model according to the selected while the second level organizes the atoms inside each rigid part of the first level. Such a data structure can be efficiently tested for collision and also updated at a moderate cost. Experimental tests performed with BioCD show its efficacy in processing thousands of collision tests per second on articulated protein chains with hundreds of DOF [Ruiz de Angulo et al. 2005].

Loop closure:

The functions `SampleConf` and `PerturbConf` (see Algorithm 1 and 2 in Chapter 4) perform a specific sampling procedure of loop conformations, taking into account loop closure constraints. Once the pose parameters of all groups G_i have been sampled, the Random Loop Generator (RLG) algorithm [Cortés et al. 2004] is applied to sample the backbone torsions of the N-terminal and C-terminal segments of each loop. This iterative algorithm, based on simple geometric operations, biases the sampling of these chain segments toward conformations with a high probability of satisfying the loop closure constraint. The constraint is reinforced within the function `Expand`, which applies an inverse kinematics method [Renaud 2000] to compute the bond torsions of the tripeptide in the middle loop part (M) for the conformations along the local expansion motion.

Hydrogen bonds and disulfide bonds:

These structural constraints can be considered within the mechanistic molecular model. Indeed, they are modeled as distance and angle constraints between the bonded atoms. For hydrogen bonds, the distance d between the donor and the acceptor atoms, and the bond angle θ, must remain within a given range. For instance, for O-H$\cdots$N bonds: $d_{\text{O-N}} \in [2.5\text{ Å}, 3.8\text{ Å}]$ and $\theta_{\text{O-H-N}} \in [110^\circ, 180^\circ]$. Disulfide bonds also imply bond length and bond angle constraints between the involved S and C atoms. Additionally, the S-S bond torsion γ is restricted around 90°. The ranges by default are $d_{\text{S-S}} \in [1.8\text{ Å}, 2.2\text{ Å}]$, $\theta_{\text{C-S-S}} \in [100^\circ, 130^\circ]$, and $\gamma_{\text{S-S}} \in [60^\circ, 120^\circ]$. All these constrains are checked within the function `Expand`.

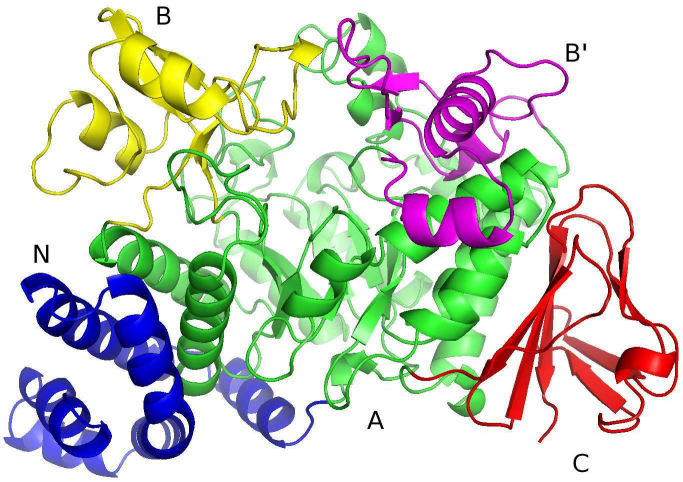


Figure 5.8: Illustration of five domains of the Amylosucrase structure (PDB ID 1G5A): N dark blue, A green, B yellow, B' magenta, C red.

5.2 Results

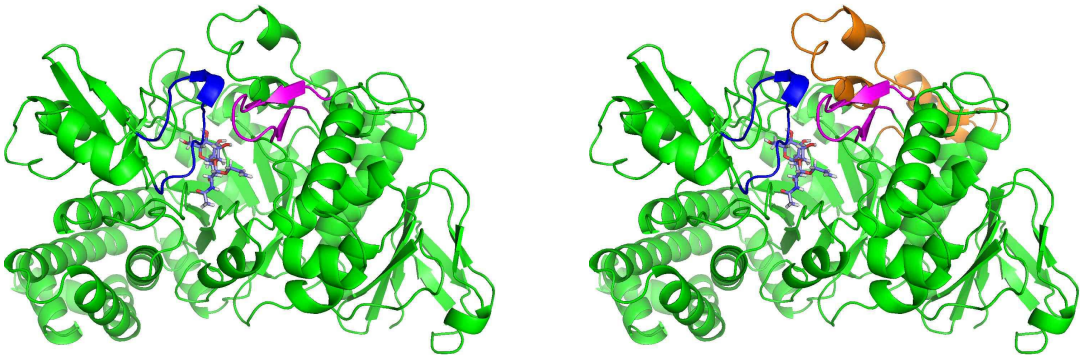
This section presents results obtained with the proposed method on three biologically interesting systems involving ligand-induced conformational changes while computing geometric-based exit pathways of the ligand from the active site. In the first system, the exit of the product from the binding site of amylosucrase, which requires the movements of several loops, is analyzed. In the second one, the mechanism of sugar permeation through LacY involves a large-amplitude relative motion of transmembrane domains. In the third system, the access/exit of a ligand to the active site of the β2-adrenergic receptor is related with side-chain motions, loop motions and transmembrane domain rearrangements. The presented results are not aimed to provide new insights into these biological systems, but to serve as a proof of concept and to show the interest of the proposed approach. The method was implemented within our software prototype BioMove3D. PyMOL [DeLano 2002] was used for viewing molecular models. The computing times reported below correspond to tests run on a single AMD Opteron 148 processor at 2.6 GHz.

5.2.1 Multi-Loop Motions: Amylosucrase

5.2.1.1 Structural Description

Amylosucrase (AS) from *Neisseria polysaccharea*, which catalyzes the synthesis of an amylose-like polymer from sucrose, belongs to the glycoside hydrolase family 13 (also called α-amylase family) [Sarcabal et al. 2000]. Since two decades ago, many studies have been performed to analyze the specificity of its binding sites for synthesis of glucan, oligosaccharea or glucoconjugates [Skov et al. 2001; Skov et al. 2002].

The AS structure contains five domains (Fig. 5.8) named N, A, B, B' and C. Three domains, A, B and C, are common to all α-amylases. Domain A (residues 98-184, 261-395, 461-550) made up by eight β-sheets and the same number of α-helices forms the catalytic core. Interestingly, the loop region that connects



(a) (b)

Figure 5.9: Models of AS with different loops (Loop 3: Blue, Loop 7: Magenta, Loop 8: Orange), and a product molecule in the active site: (a) Two-loop model, (b) Three-loop model.

Table 5.1: Flexibility of three considered models.

Model	**Number of DOF**				
	Domain/Helix	Side-chain	Loop's backbone	Product/Substrate/Ligand	Total
AS 2 Loops	0	1474	30	22	1526
AS 3 Loops	0	1474	76	22	1572
LacY	12	678	75	10	775
β_2-AR	42	490	159	12	703

strands to helices (loop1 and loop8) are much longer on average than those connecting helices to strands [Skov et al. 2002]. Domain B (residues 185-260) containing two short anti-parallel β-sheets can be found in many α-amylases. This domain has a very long loop 3 (Thr224-Gly236). Eight-stranded β-sandwich (residues 555-628) creates the last domain C. Domains N, which is specific to AS, containing only α-helical form comprises from 1 to 90 first residues. The interesting domain B' (residues 395-460) is also specific to this enzyme. It starts with two α-helices followed by short β-sheets and terminates with another short α-helix, and it contains a flexible loop 7 (Gly433-Gly449) [Albenne et al. 2007].

The active site of AS, which is often referred to as OB1, has a 15 $\AA$ deep and narrow pocket like architecture with seven well-defined subsites spanning from -1 to +6 according to the glycoside-hydrolase nomenclature [Champion et al. 2009]. Due to this type of active site topology, the analysis of the interactions between substrate/product molecules and AS along the access/exit pathway, as well as the deformations induced by these interactions, are very important for a better understanding of the specificity and the enzymatic mechanism of AS.

In this work, we have used a model of AS with a product molecule (Allyl-α-$_D$-Glcp-(1$\rightarrow$4)-α-$_D$-GlcNac) in the active site[1] in order to analyze the interactions and the protein deformations induced by the product exit. Two instances of the problem were considered. In the first one, two loops 3 and 7 were considered to be flexible, whereas in the second one the loop 8 was also involved. These two instances are

[1]The model of AS with the product molecule in the active site was generated using molecular modeling tools by our colleagues at the LISBP-INSA.

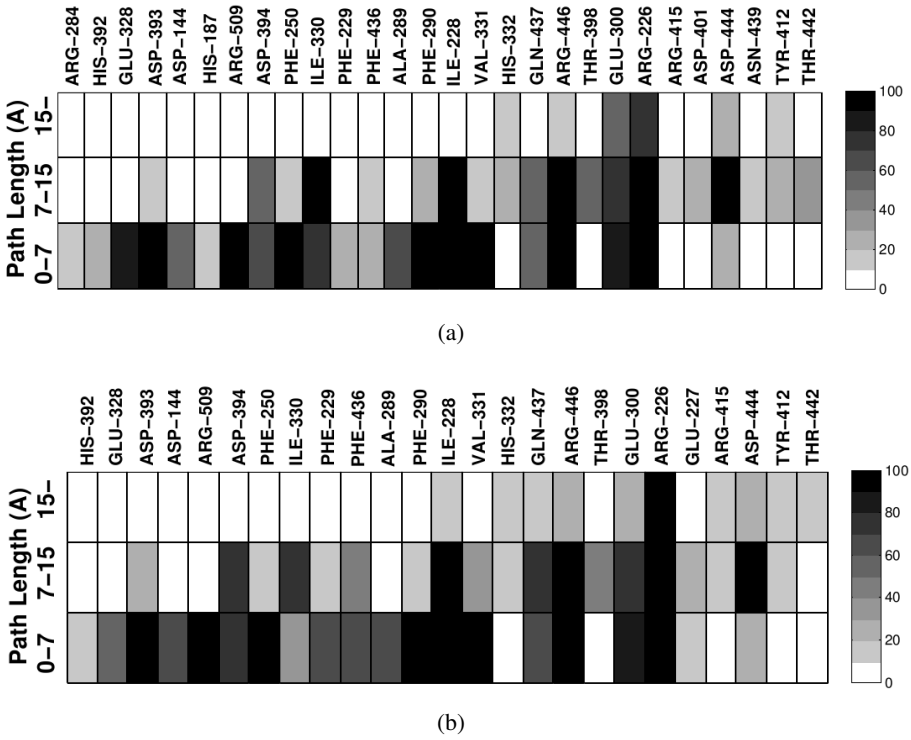


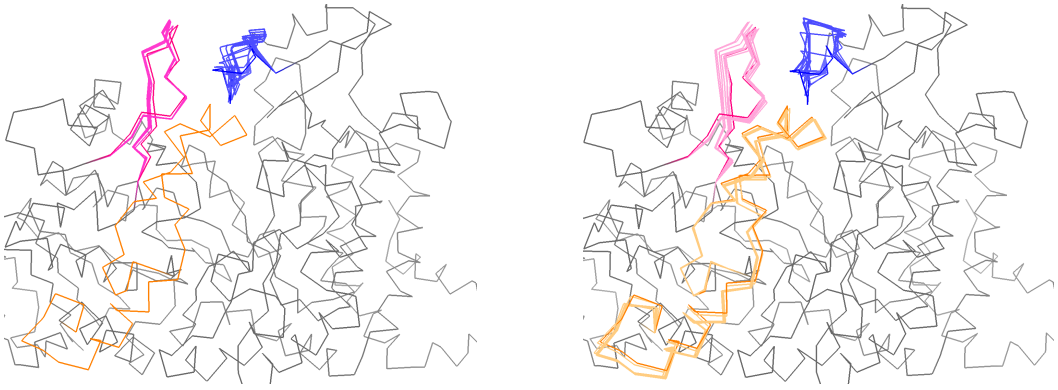
Figure 5.10: List of residues encountered by the product during its diffusion toward the exit of OB1 binding site of the AS. For facilitating interpretation, the pathway is divided into three segments. The grey-scale represents the percentage of times that the contact appears over the set of 20 paths. a) 2-Loop Model b) 3-Loop Model.

illustrated in Figure 5.9. Note that all the side chains in the protein were also considered to be flexible. The product molecule was modeled with full flexibility, and it could freely rotate and translate by 50 Å in any direction. Table 5.1 shows the number of degrees of freedom (DOF) for each model. Overall, the mechanistic models of enzyme-product complex contain 1526 DOF and 1572 DOF for the 2-loop and the 3-loop models respectively, in which 30 and 76 correspond to the backbone torsions of the two and three loops respectively, 1474 to the protein side chains, and 22 to the product molecule mobility and flexibility.

5.2.1.2 Access/Exit Pathway Analysis

The ML-RRT algorithm was applied to compute the exit pathway of the product from the binding site OB1 of AS, surrounded by two or three flexible loops. Table 5.2 shows the performance of the algorithm based on 20 successful runs for each model. An analysis of the contacts between the product and protein residues based on the 20 exit pathways was performed in order to identify key residues for product release. Note that residues involved in product release would also be important for the substrate entry.

The diagrams in the Figure 5.10 represent the residues encountered by the product molecule along its escape trajectory for the 2-loop and 3-loop models. A contact between the product and a protein's residue was recorded if the distance between the surface of van der Waals spheres modeling their atom was below 0.8 $\AA$. This value which is determined by experimentation permit to obtain most significant residue contacts.



(a) (b)

Figure 5.11: Superposition of the initial structure of Amylosucrase with considered loops (Loop 3: Blue, Loop 7: Magenta, Loop 8: Orange) and conformations induced by the product exit (corresponding light colors) following the 2-loop model (a), and the 3-loop model (b).

Table 5.2: Algorithm Performance.

Performance	**Two-Loop Model**	**Three-Loop Model**
Ave. Time(s)	2357.05±1024.97	3648.74±1174.12
N_{node}	7011	8297
N_{samp}^{act}	8301	10410
N_{coll}	58843	66757

These diagrams show the percentage of times that a contact appeared over the set of 20 paths. Recorded contacts were registered into three segments of the path: the beginning (0-7 $\AA$), the middle part (7-15 $\AA$), and the final part (above 15 $\AA$) where the product is close to the exit of the narrow pocket. Only residues with contact percentage greater than 10 % are displayed. Overall, the residues that are encountered with higher probability are the the same in both diagrams.

Remarkably, most of the residues highlighted by our results have been shown to be important residues. Asp286, Glu328 and Asp393, located at the bottom of the pocket, form the catalytic triad [Albenne et al. 2002; Champion et al. 2009]. Other identified residues such as His187, Phe250 and His392 are know to be important for substrate stabilization [Albenne et al. 2002; Albenne et al. 2004]. Residues Asp144 and Arg509, which form a salt-bridge highly conserved in α-amylases [Albenne et al. 2002], are also identified. Note the the product exit, as well as the substrate entry, require the breakage of this bridge. Residues Asp394 and Arg446, located in the first shell of subsite +1, are know to be important in the catalytic mechanism of AS [Albenne et al. 2002]. More importantly, site-directed mutagenesis experiments [Champion et al. 2009] have shown that mutation of residues Ile228 and Phe290, which are systematically found in the computed exit paths, lead to a significant catalytic efficiency improvement. Finally mention that residues Arg226 and Asp444, pointed out by our method since they highly hinder the diffusion of the product in the middle part of the pathway, will be the object of future site-directed experiments carried out by our colleagues at the LISBP-INSA. Figure 5.11 shows the conformational changes of the two models induced by the product exit.

Table 5.3: Distance variation between residue pairs in LacY.

Residue pair	Inward-open	Outward-open (experimental [Smirnova07])	Outward-open (simulation)
73-401	41 Å	27 Å	36.9(±1.0) Å
73-340	36 Å	21 Å	31.0(±1.3) Å
136-340	34 Å	17 Å	28.7(±1.4) Å
137-340	32 Å	16 Å	26.7(±1.4) Å
136-401	40 Å	24 Å	35.6(±1.3) Å
137-401	38 Å	22 Å	33.5(±1.4) Å
105-310	34 Å	41 Å	38.0(±1.6) Å
164-310	27 Å	43 Å	32.8(±1.4) Å
164-375	33 Å	49 Å	35.8(±1.2) Å

5.2.2 Domain Motions: Lactose Permease

5.2.2.1 Structural Description

Lactose permease (LacY) is a transport protein that transduces electrochemical proton gradients into sugar concentration gradients across the cell inner membrane [Kaback et al. 2001]. LacY is composed of two main domains [Abramson et al. 2003]: the N-domain involving helices I-VI, and the C-domain involving helices VII-XII. The two domains are connected by a long loop containing more than 20 residues. For carrying out its function, LacY is supposed to alternate between two conformational states: the inward-open state, where the substrate is accessible from the cytoplasm, and the outward-open state, where the access is possible from the periplasmic side. However, only the structure of the inward-open conformational state of LacY has been solved by X-ray crystallography.

Different approaches have been used to analyze the conformational transition pathway toward the outward-open state. In particular, experimental studies using double electron-electron resonance (DEER) [Smirnova et al. 2007] suggest that the conformational transition can be mainly described as a rigid-body rotation of the C-domain and the N-domain. Based on such structural knowledge, the mechanistic model of LacY was simplified by considering a rigid backbone for the C- and N- domains. Flexibility was allocated to the loop between helices VI and VII, and to all the protein side-chains. Thus, the mechanical model contains two main groups G_1 and G_2 associated with the C-domain and the N-domain respectively, and an inter-domain loop $eL_{1,2}$. Groups G_1 and G_2 were permitted to rotate $\pm 60^\circ$ around any axis.

The X-ray structure of LacY of *Escherichia coli* [Abramson et al. 2003] (PDB ID 1PV7), corresponding to the inward-open conformation, and used as starting point in this work, contains a bound substrate homologue TDG (see Figure 5.1.a). The substrate molecule was modeled with full flexibility, and it could freely rotate and translate by 50 Å in any direction except the direction to the cytoplasm (only 5 Å were permitted in this direction in order to force the exit toward the periplasmic side). Overall, the mechanistic model of LacY-TDG contains 775 degrees of freedom: 12 correspond to the rigid-body motion of the C- and N- domains, 678 to the protein side-chains, 75 to the backbone torsions of the inter-domain loop, and 10 to the substrate mobility and flexibility (see Table 5.1). The initial "assembled" conformation is represented in Figure 5.1.a.

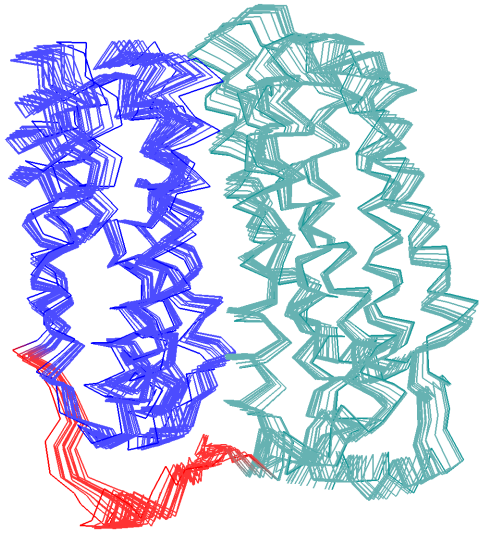


Figure 5.12: Superposition of the initial structure of LacY with two domains (Domain N: blue, Domain C: cyan) and a loop (red), and conformations induced by the ligand exit (corresponding light colors).

5.2.2.2 Analysis of Conformational Changes

The ML-RRT algorithm was applied to compute the exit pathway of TDG toward the periplasmic side, which involves the conformational transition of LacY. The computing time of a run was about 1 hour on a single processor AMD Opteron 148 processor at 2.0 GHz equipped with 2 Gb of memory. Such high computational performance is worth to be noted since it represents an important feature of the proposed approach compared to the very long computing times required by other simulation methods such as molecular dynamics. The algorithm was run 10 times in order to analyze a possible variability of results associated with the randomized exploration procedure. All the experiments yielded very similar results with regard to the protein conformational change. The obtained "disassembled" conformation, with the ligand outside the protein and LacY in outward-open state, is represented in Figure 5.1.b. As it has been pointed out by prior studies [Smirnova et al. 2007], the substrate exit requires the rotation of the two domains. In our results, the observed rotation between the domains is around 20°. Although this is smaller than the 60° suggested by DEER experiments, the overall motion is alike. The comparison of the variation of distances between some residue pairs in the inward- and outward- faces of LacY (see Table 5.3) shows an approximate overall ratio of 1/3 between the values measured by DEER and our results. The explanation to this quantitative difference is that our approach tends to produce the minimal conformational change required for the molecular disassembly, while larger motions may occur in reality. Interestingly, the distance between residues Ile40 and Asn245 in the outward-open conformation computed by ML-RRT is of approximately 15Å, which has been shown by cross-linking experiments [Zhou et al. 2008] to be the minimal distance between these residue positions for guaranteeing the activity of LacY. Figure 5.12 shows the conformational changes of LacY induced by the ligand exit.

In other recent studies [Jensen et al. 2007], steered molecular dynamics (SMD) simulations have been carried out to better understand the physical mechanisms of lactose permeation at the atomic level. SMD results provide detailed information about the interactions between lactose and LacY residues during

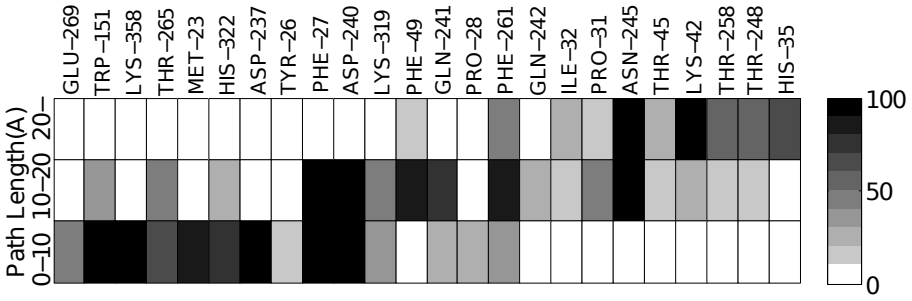


Figure 5.13: List of residues whose side-chain was encountered by the substrate during its diffusion toward the periplasm. For facilitating interpretation, the pathway is divided into three segments. The grey-scale represents the percentage of times that the contact appears over the set of 10 runs.

permeation. Such kind of information cannot be directly provided by our method, since it does not consider accurate energy functions. However, a straightforward geometric analysis of the paths obtained by ML-RRT can provide the list of residues that the ligand has encountered during its diffusion. The diagram in Figure 5.13 represents the residues encountered by the ligand along the path toward the periplasm.

A contact between the ligand and a residue side-chain was recorded if the distance between the surface of van der Waals spheres modeling their atoms was below 1 Å. The diagram shows the percentage of times that a contact appeared over the set of 10 paths. Contacts were recorded for three segments of the path: the beginning (0-10 Å), where the ligand is close to its location in the crystal structure, the middle part (10-20 Å), and the final part (above 20 Å), where TDG is near the periplasm. Remarkably, all the residues identified by SMD simulations [Jensen et al. 2007] as interacting residues (throught side-chain hydrogen bonds or hydrophobic interactions) appear in the diagram, with the exception of Asp36. Note however that this residue is on the periplasmic surface of the protein. On the other side, only one potentially interacting residue (Thr265) that is appearing in the contact diagram with a significant percentage is not reported in the referred work. Such an impressive consistency with results of SMD simulations confirms the validity and the potential interest of our approach.

5.2.3 Helix Motion: $\beta 2$-Adrenergic Receptor

5.2.3.1 Structural Description

The **β_2-Adrenergic Receptor** (β_2-AR) is a membrane protein belonging to the superfamily of the G-protein-coupled receptors (GPCRs) [Vauquelin and von Mentzer 2007], which activate signal transduction inside the cell in response to the binding of hormones and neurotransmitters in the extracellular region. GPCRs are important therapeutic targets for a large class of diseases. Therefore, numerous studies have been devoted to this family of proteins, aiming to better understand their activation/deactivation mechanism. However, many questions remain. In particular, little is known about the functional role of extracellular loops, and about their possible conformational coupling to ligand binding [Bokoch et al. 2010]. One major difficulty comes from the lack of structural information inherent to membrane proteins.

A high-resolution crystal structure of β_2-AR has been recently obtained [Cherezov et al. 2007]. The crystal structure also contains a molecule of carazolol, a partial inverse agonist, in the protein active site.

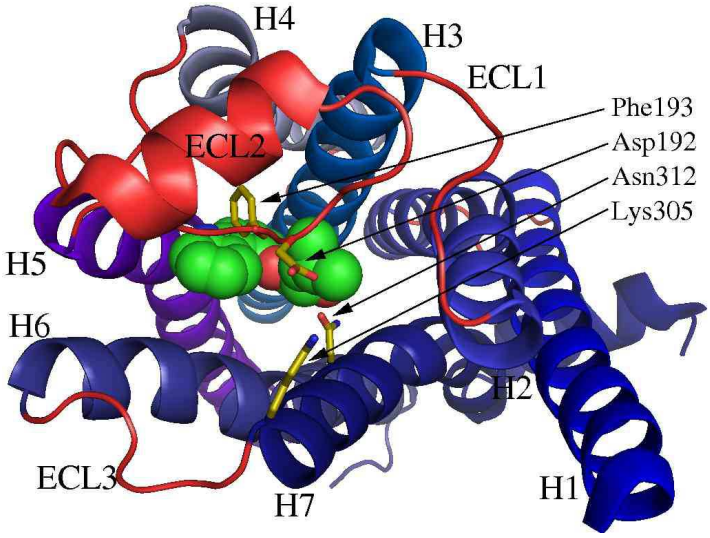
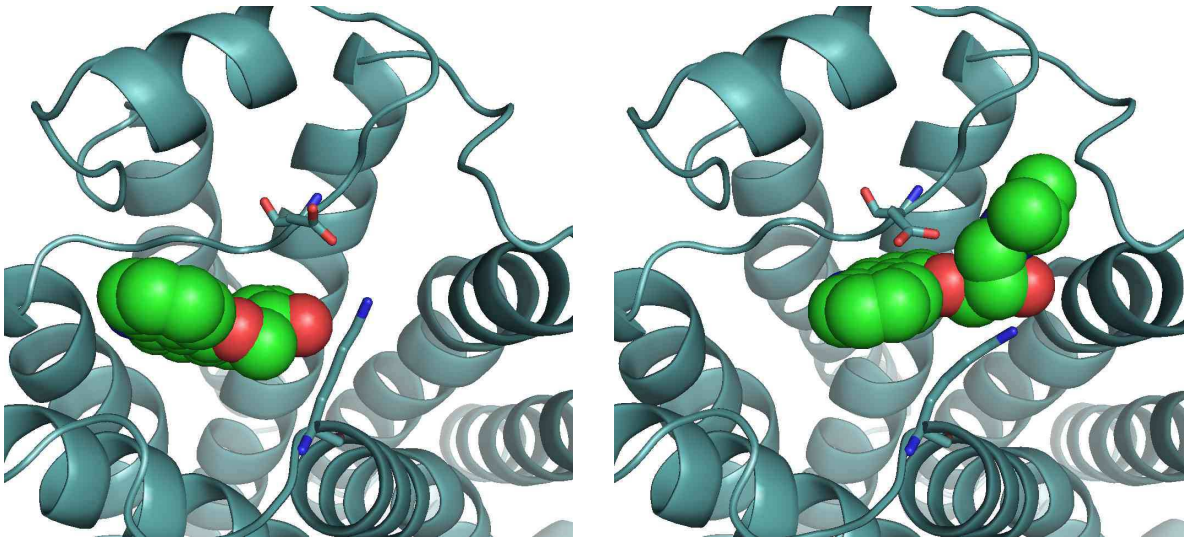


Figure 5.14: Structure of β_2-AR with carazolol bound in the protein active site viewed from the extracellular side. The secondary structure elements and important residues are displayed on the image.

This receptor-ligand structure is the starting point of the conformational analysis presented below. The structure is represented in Figure 5.14, using standard notation for the structural elements. Like all GPCRs, β_2-AR contains seven transmembrane helices, which were modeled as rigid groups G_i. The intracellular and extracellular loops were modeled as flexible elements $eL_{i,i+1}$. Note that, in the crystal structure, the third intracellular loop was replaced by T4 lysozyme (T4L) to facilitate the growth of diffraction-quality crystals [Cherezov et al. 2007]. Since T4L is *a priori* not important for the exit/access of the ligand, it was removed for constructing the mechanistic model considered in this study. All the side-chains are fully flexible. Groups G_i can move ± 2 Å in any direction and can rotate $\pm 30^\circ$ around any axis. The ligand is also modeled with a fully flexibility, and it can freely translate and rotate in a cube of edge length 50 Å centered at the initial docking position. The number of degrees of freedom of the whole model is 703: 42 of them correspond to the rigid-body motion of the seven transmembrane helices, 490 to the protein side-chains, 159 to the backbone torsions of the five loops, and 12 to the ligand mobility and flexibility (see Table 5.1).

5.2.3.2 Analysis of Conformational Changes

The exit pathway of carazolol from the active site of β_2-AR was computed by using the ML-RRT. A first set of 10 runs revealed some variability on the trajectories followed by the ligand. Thus, the algorithm was run 60 times in order to do a more accurate statistical analysis of results while comparing with prior works [Wang and Duan 2009]. 60 paths were obtained in 2 hours of computing time (each run takes an average of only 2 minutes on a single processor AMD Opteron 148 processor at 2.0 GHz equipped with 2 Gb of memory). These exit paths can be divided into two main clusters. In one class of paths, which we refer to as "left-hand" paths, carazolol exits between transmembrane helices H5, H6 and H7. In the other class, called "right-hand" paths, the ligand exits between H2, H3 and H7. The two clusters can be separated by an axis traced between residues Asp192 and Lys305, which form a salt bridge in the crystal structure. Figure 5.15 shows snapshots of the ligand exit path for each path class.



(a) (b)

Figure 5.15: Snapshots of the ligand exit from β_2-AR following the left-hand pathway (a), and the right-hand pathway (b).

Interestingly, these two classes of exit paths have also been observed in prior studies [Wang and Duan 2009] based on random acceleration molecular dynamics (RAMD) simulations. A quantitative comparison can be done between results obtained with ML-RRT and RAMD. The most significant comparable result is that both approaches suggest that left-hand and right-hand exit paths are approximately equiprobable. Indeed, 31/60 of the ML-RRT solutions correspond to left-hand, and 29/60 to right-hand paths. Another result from RAMD simulations described by [Wang and Duan 2009] concerns the recurrent breakage of the salt bridge Asp192-Lys305 during ligand exit. Paths computed with ML-RRT show a significant motion of the side-chains of these two residues, which lead to the salt bridge breakage for most of the 60 paths. However, in some of the left-hand paths, the ligand exits with only a slight perturbation in the conformation of Asp192 and Lys305. The interpretation is that it is geometrically possible for the ligand to exit between helices H5, H6 and H7 without breaking the salt bridge.

A further comparison between left-hand and right-hand paths obtained with ML-RRT displays other interesting differences. The first one concerns the orientation of the ligand. In most of the left-hand paths, the ring head of carazolol reaches first the protein surface (see Figure 5.15.a). Contrarily, the ring and the alkylamine-alcohol tail exit almost simultaneously in most of the right-hand paths (Figure 5.15.b). A possible interpretation may be that one of the pathways could be preferred for the exit of the ligand, while the other could be more suited to the access. A more accurate analysis of the paths computed by ML-RRT would be required to reinforce such a suggestion. Note however that RAMD simulations from a putative ligand-free model of β_2-AR [Wang and Duan 2009] suggest that carazolol enters the receptor between helices H2, H3 and H7, with its ring head diving first.

Another interesting difference between the two classes of exit paths concerns the conformational changes of the extracellular loop ECL2 induced by the ligand exit. As shown in Figure 5.16, right-hand paths imply, in average, a more significant motion of ECL2 than left-hand paths. Note that although the

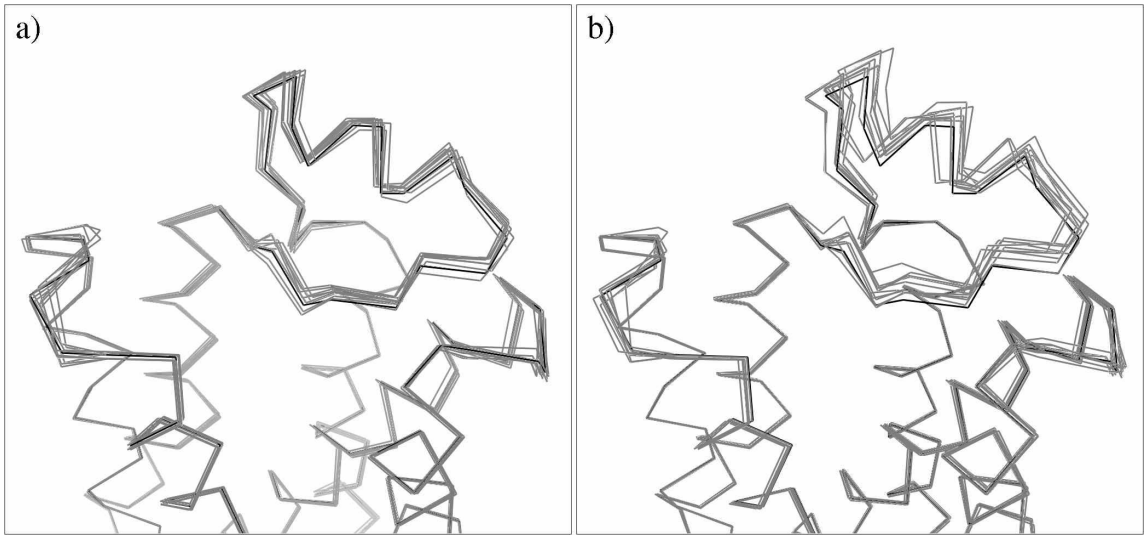
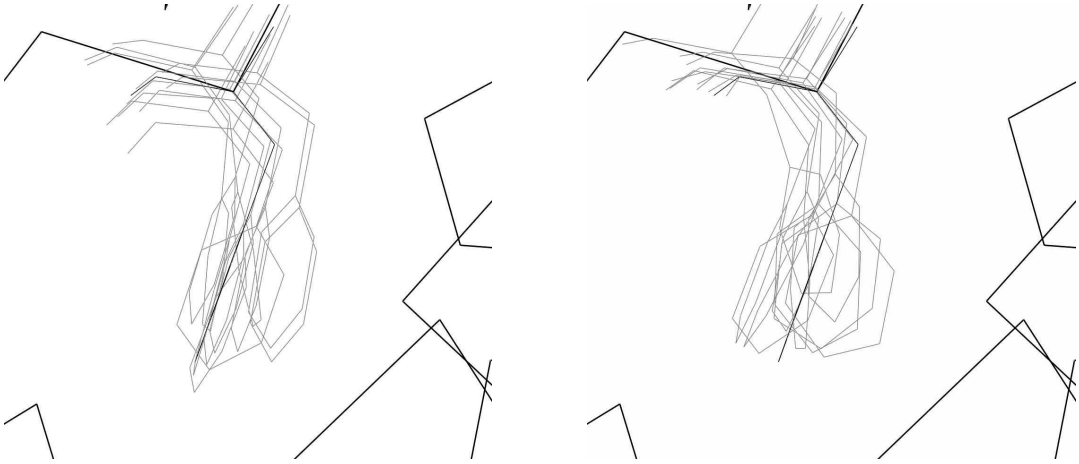


Figure 5.16: Superposition of the initial structure of β_2-AR (black) and conformations induced by the ligand exit (grey) following the left-hand pathway (a), and the right-hand pathway (b).

loop ECL2 of β_2-AR is very long, its conformation is constrained by two disulfide bonds, one between residues in the loop (Cys184-Cys190), and one between the loop and H3 (Cys106-Cys191). Thus, in any case, this loop cannot undergo large conformational changes. The observed relationship between right-hand paths and ECL2 flexibility has been confirmed by tests performed on a model of β_2-AR only considering side-chain flexibility. Using this rigid-backbone model, the ligand exited through the left-hand pathway in 90% of the ML-RRT runs. These results suggest that right-hand access/exit paths, between transmembrane helices H2, H3 and H7, involve a more important interaction between the ligand and ECL2 than left-hand paths. Note that recent studies on GPCRs show important roles of ECL2. Indeed, it can be required for ligand binding [Avlani et al. 2007], and its motion can be involved in the activation mechanism [Ahuja et al. 2009]. In the crystal structure, the access/exit channel to the active site is partially covered by the second extracellular loop (ECL2). The ligand exit requires the opening of this loop.

The analysis of contacts between carazolol and β_2-AR residues along the set of 60 exit pathways computed with ML-RRT was performed using the technique described above for the study of LacY. Figure 5.18 shows the list of residues whose side-chain was encountered by the ligand. For clarity reasons, the figure only reports contacts that appeared in more than 30% of the paths. Four residues are clearly highlighted in the diagram: Asp192, Phe193, Lys305, and Asn312. The positions of these residues are indicated in Figure 5.14. Two of them, Asp192 and Lys305, form the aforementioned salt bridge, which is broken during the ligand diffusion. Phe193, which is located on ECL2, has also been identified as an important residue in related works. As shown in Figure 5.17, right-hand paths imply, in average, a more significant motion of this key residue than left-hand paths do. Results of RAMD simulations [Wang and Duan 2009] suggest that this aromatic residue may participate in the ligand entry and stabilization in the active site of β_2-AR. Recent NMR experiments [Bokoch et al. 2010] have shown that inverse agonists induce a conformational change of this residue. Finally, Asn312 is an important residue for the stabilization of carazolol in the active site through a polar interaction with its alkylamine-alcohol tail. Overall, the presented results show that structural information on the access/exit of carazolol to the active site of β_2-AR



(a) (b)

Figure 5.17: Superposition of the initial structure of the residue Phe193 (black) and conformations induced by the ligand exit (grey) following the left-hand pathway (a), and the right-hand pathway (b).

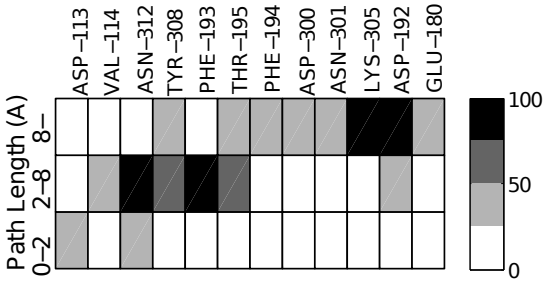


Figure 5.18: List of residues whose side-chain was encountered by the ligand along the exit pathway. For facilitating interpretation, the pathway is divided into three segments. The grey-scale represents the percentage of times that the contact appears over the set of 60 runs. Only contacts that appeared in more than 30% of the paths are displayed.

provided by Pushing-Pulling-ML-RRT is in agreement with results of other experimental and computational studies.

5.3 Discussion

The results in this chapter show that a mechanistic approach to molecular simulations may lead to the development of efficient computational methods, able to provide relevant information on the interaction of biological molecules. The proposed algorithm, ML-RRT, is a novel and fast conformational search method for simulating ligand diffusion inside flexible models of proteins including more than thousands of degrees of freedom. Indeed, this algorithm generates long (20-30 Å) diffusion paths within tens of minutes of computing time on a single processor, which is remarkably short compared to the time required by MD-based methods. Such a high computational performance is achieved thanks to the efficiency of the conformational search method that operates on geometric models of molecules. Geometrically feasible paths are a reasonably good approximation that provides itself very useful information. Furthermore, as

shown in prior work [Cortés et al. 2005], the approximate solution path can also be efficiently refined with standard molecular modeling tools (e.g. energy minimization) in order to perform a more accurate energetic analysis.

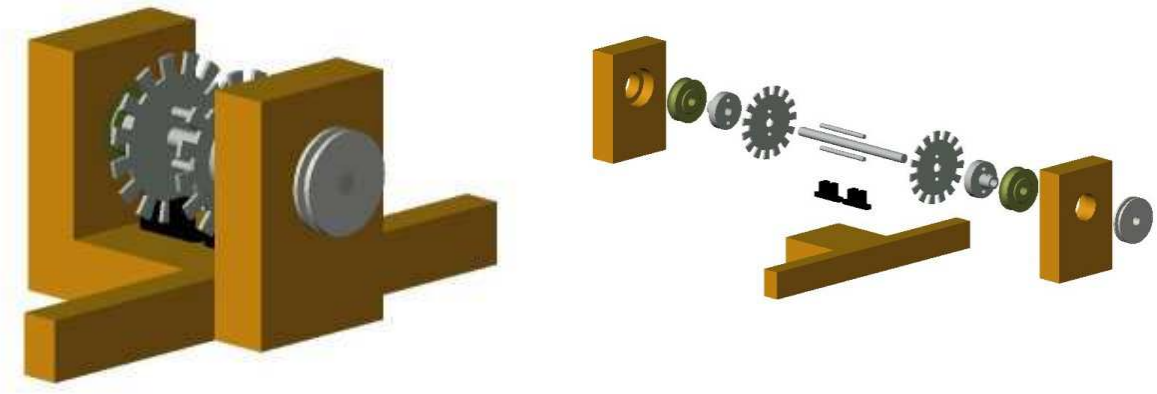
As future work, we intend to further improve the method to better deal with full molecular flexibility during protein-ligand interactions. We also expect to extend the method for its application to the modeling of protein-protein interactions. One way to do it could be to combine ideas of ML-RRT and T-RRT algorithms [Jaillet et al. 2008]. Basically, T-RRT method extends RRT-like algorithm to bias the tree exploration toward low-energy region of the conformational-energy landscape. Therefore, the integration of T-RRT inside the "Pushing-Pulling" variant of ML-RRT should be useful to generate energy-minimized pathways in ligand-protein or protein-protein interaction studies. An implementation of this combination is introduced in Chapter 7.

6

Path Planning Approach to Disassembly Sequencing

Disassembly planning is a very active research field with a number of direct applications such as end-of-life product processing, maintenance operations, and product repair [Lambert and Gupta 2005]. Moreover, integrated within CAD/CAM systems [Arai and Iwata 1993], disassembly planning helps to design products that are easier to manufacture, to maintain and to recycle. Since a bijection between assembly and disassembly sequences usually exists [Halperin et al. 2000], the assembly-by-disassembly strategy has been a common approach to assembly planning. Thus, although a distinction between both problems can be made [Lambert 2003], it is usual to talk indistinctly about assembly and disassembly planning. This chapter directly addresses the latter problem. Nevertheless, like most of related works, the proposed method can be used to infer the assembly sequence from the model of the assembled object. Figure 6.1 illustrates the problem of disassembly sequencing: what is the order for sequentially disassembling an engine into its elementary parts?

In this chapter, we present an algorithm that extends prior works to disassembly sequencing problem of complex assemblies. This algorithm will be applied in Chapter 7 to simulate sequential dissociation pathways of protein complexes. We first provide a short state of the art on disassembly planning in Section 6.1. Then, we propose a general formulation for simultaneously disassembly sequencing and path planning (Section 6.2). Based on this formulation, the method builds on sampling-based path planning algorithms, which are able to solve problems in high dimensions (Section 6.3). Results and an empirical performance analysis (Section 6.4) show the good performance and the generality of the method. One of the reasons for such a generality is that the method only requires simple computational geometry tools such as collision detection. However, in some cases, further geometric operations will be integrated in the algorithm in order to enhance its performance (Section 6.5).



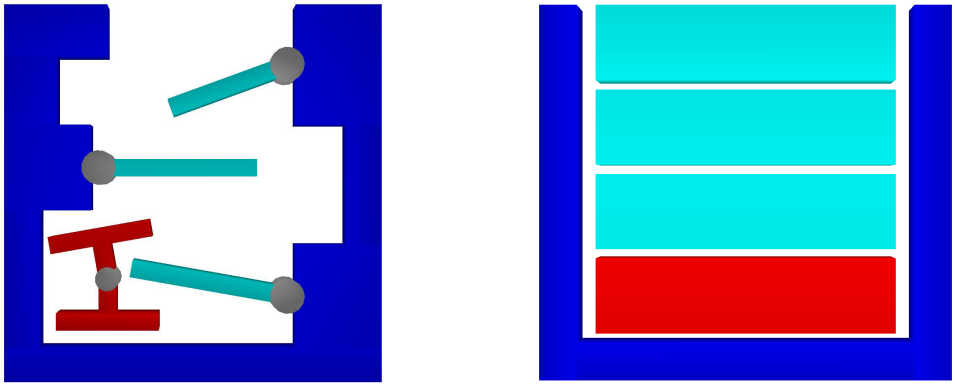
(a) (b)

Figure 6.1: Illustration of 15-component encoder assembly [Duggan et al. 2000] : (a) Assembled State, (b) Disassembled State.

6.1 Related Works and Overview of the Proposed Method

Disassembly planning can be tackled at different levels of detail [Lambert 2003]. The highest level concerns *disassembly sequencing*, which is the problem of listing subsequent disassembly actions that can separate individual parts of an assembly. This problem is usually formulated as a discrete search and optimization problem, and is solved using AI methods such as AND/OR graphs [Lambert 2000] or genetic algorithm [Kongar and Gupta 2001]. Geometric reasoning approaches (e.g. [Wilson et al. 1995; Halperin et al. 2000; Fogel and Halperin 2008]) can be applied at this level in order to reduce the combinatorial complexity of the disassembly sequencing problem. *Disassembly path planning*, which addresses the parts motion considering physical and manipulability constraints, rises at a more detailed level of the disassembly planning problem. Due to the high computational complexity of treating all part motions simultaneously, the disassembly path planning problem has usually been formulated for a single part. This simpler instance is also called the *assembly maintainability study* [Chang and Li 1995]. Efficient path planning methods (e.g. [Ferré and Laumond 2004; Aguinaga et al. 2008]), based on the popular RRT algorithm [LaValle and Kuffner 2001b], have been proposed to solve very constrained single-part disassembly problems on complex CAD models. Based on this type of method, the ML-RRT algorithm, presented in Chapter 4, addresses disassembly path planning for objects with articulated parts.

However, disassembly sequencing and path planning are parts of a whole problem, and ideally, they have to be treated simultaneously. The relationship between both sub-problems is more obvious for non-monotonic disassembling (see Figure 6.5 for an example), in which parts have to be moved to intermediate locations for permitting the disassembly of other parts. Despite their potential interest, few methods have been proposed for simultaneously disassembly sequencing and path planning in a general framework, like the one presented in this chapter. Probably the most closely related method was proposed by Sundaram *et al.* [Sundaram et al. 2001]. Based on randomized path planning algorithms, this technique was able to compute disassembly sequences considering all the parts disassembly paths. The method samples motion



(a) (b)

Figure 6.2: Similarity between articulated disassembly and disassembly sequencing with the same idea of [Active–Passive] degrees of freedom decomposition: (a) Articulated Disassembly, (b) Disassembly Sequencing.

directions of one or several parts using geometric information (i.e. the normal direction to faces in contact). Although general, this method strongly depends on geometric operations, and thus, its performance for solving problems involving parts with complex shapes is questionable. Besides, its ability to treat non-monotonic disassembly problems was neither experimentally proved nor discussed.

This chapter introduces a general formulation for simultaneously disassembly sequencing and path planning, and proposes an algorithmic solution based upon it. The proposed method extends the ML-RRT algorithm, described in Chapter 4, which was originally proposed for disassembly path planning of two objects with articulated parts (see example in Figure 6.2.a). The idea developed in this chapter consists in iterating the ML-RRT algorithm for extracting all the parts of a general assembly. A simple example of disassembly sequencing and path planning problem is illustrated in Figure 6.2.b. At each iteration, a different active-passive partition of the system configuration parameters will be considered. This partition can be based on information about assembly structure. Nevertheless, if such information is difficult to obtain, the algorithm still provides good results when using a random selection. At the end of the iterative process, the algorithm has computed a sequence of elementary part motions yielding the system disassembly. Although optimality criteria are not explicitly considered, the method tends to minimize the number of elementary motions.

6.2 Problem Formulation

This section presents a unified formulation for disassembly path planning and disassembly sequencing. The formulation is a generalization of the one presented in Chapter 4 for the disassembly path planning problem. A configuration q is a minimal set of parameters defining the location of the mobile system in the world, and the configuration-space C is the set of all the configurations. Given the initial assembled configuration q_{init}, the problem consists in finding a feasible path in C form q_{init} to a disassembled configuration q_{dis}. Note that q_{dis} may not be specified by a precise goal configuration, like in the standard path planning problem, but it can be implicitly defined by a condition based on distances between parts. Path feasibility in this context

Algorithm 4: `Iterative-ML-RRT`

```
input  : the model M; the search-space C; the root q_init;
output : the disassembly pathway sequence S;
begin
    n_iter ← 0;
    repeat
        n_iter ← n_iter + 1;
        m_i ← SelectPart(M);
        (P_act, P_pas) ← SetPartition(M, C, m_i);
        τ ← InitTree(q_init);
        while not StopCondition(τ, MaxTreeSize) do
            if Construct_ML-RRT(M, C, τ, P_act, P_pas) then
                if PartDisassembled(P_act, q_new) then
                    break;
        q_end ← SelectEndConf(τ);
        if not TooSimilar(q_init, q_end) then
            s_i ← ExtractSubPath(τ, q_init, q_end);
            S ← MergeSubPaths(s_i);
            q_init ← q_end;
    until AllDisassembled(q_end) or n_iter ≥ MaxIter;
    if AllDisassembled(q_end) then return S;
    else return FAILURE;
end
```

mainly involves collision avoidance. Nevertheless, constraints on the the number of hands, the possible motion directions, and optimality criteria can also be considered.

In the case of a system M involving n mobile objects m_i (i.e. the assembled parts), C is the Cartesian product of the configuration-spaces of all the objects: $C = \prod C_{m_i}$, $i = 1 \ldots n$. Motions of a single object m_a, which we will call *active part*, take place in a sub-manifold $C_j^a = C_{m_a} \times q_j^p$, where q_j^p is a point in a lower-dimensional manifold $\overline{C_{m_a}} = \prod C_{m_i}$, $\forall i \neq a$. This point q_j^p represents a fixed location of all the other objects, referred to as *passive parts*. Note that for each value of q_j^p, C_j^a corresponds to a foliation leaf of C. Such a foliation F_a of C can be made for each object $m_a \in M$ being selected as an active part.

Starting from a configuration of the whole system q_j (initially, q_j is the assembled configuration q_{init}), one part m_a can be disassembled without moving other parts if it exists a feasible path in C_j^a between $q_j = (q_j^a, q_j^p)$ and a configuration $q_{sub,a} = (q_{dis}^a, q_j^p)$, which represents a subassembly with m_a extracted from the assembly. Otherwise, disassembling m_a will require motions of other parts that can be find by exploring a continuous sequence of adjacent foliation leaves in F_a starting from C_j^a.

Considering that parts are moved and disassembled one by one (what is called a *two-handed* (dis)assembly sequence in related literature), an assembly admits a monotonic disassembly sequence if the path for disassembling each part m_a can be found in only one leaf C_j^a of its corresponding foliation (i.e. F_a) being $q_j = q_{init}$ for the first part or $q_j = q_{sub,a-1}$ for the others parts. Here, $q_{sub,a-1}$ represents a subassembly where all the previous parts in the sequence have been already extracted. Therefore, the minimal number of leaves C_j^a needed to be explored is equal to the number of parts minus one (the last part does not need to move). For non-monotonic disassembly problems, motions of parts to intermediate configurations that do not correspond with a subassembly are required. Thus, the number of leaves to be explored increases,

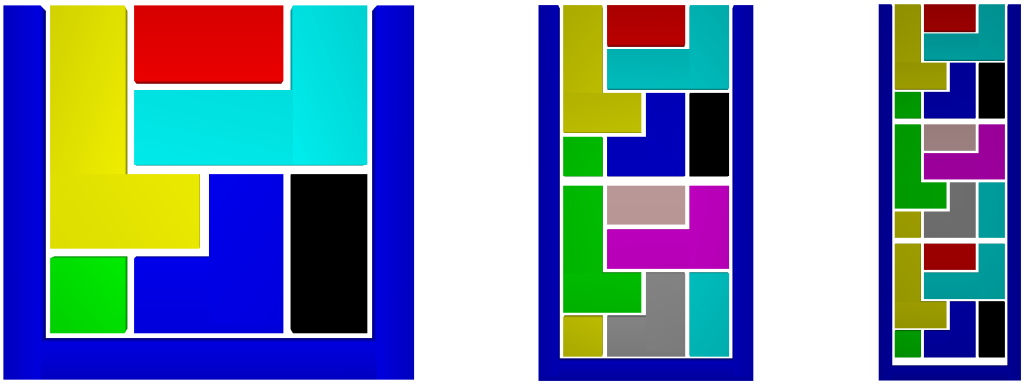
and it is necessary to find paths in C connecting these different leaves. Disassembly sequencing within this formulation can be expressed as the problem of finding the order to select active parts m_a that minimizes the number of leaves C_j^q to be explored for finding all the disassembly sub-paths.

6.3 Disassembly Sequencing Algorithm

The basic ML-RRT is able to rapidly compute the path for extracting a part from an assembly, also producing motions of other parts if necessary. The idea developed in this chapter consists in iterating the ML-RRT algorithm a number of times for extracting all the parts. The Iterative-ML-RRT (I-ML-RRT) algorithm is sketched in Algorithm 4. At each iteration, one part is tried to be disassembled. The part is selected by the function `SelectPart`. Part selection may follow a predefined disassembly order to be tested, or maybe determined by a heuristic based on the assembly structure. If such information is not available, the part is simply selected at random. The function `SetPartition` sets the pose parameters of the selected part as active parameters P_{act}, while those of all the other parts are passive P_{pas}. The variance of the Gaussian distribution associated with the mobility coefficient of passive parts is initialized with a high value, making these parts have low tendency to move in the first iterations. This coefficient is used by the function `PerturbConf` in the Algorithm 3 (`Construct_ML-RRT`) in Chapter 4. Once the parameter partition is set, the basic ML-RRT is applied to compute the disassembly path. The ML-RRT expansion process is stopped in two cases: if the currently treated part reaches a disassembled configuration (detected by the function `PartDisassembled` when the distance to all the other parts is greater than a given threshold), or if a `StopCondition` determines that the part extraction is not possible. In our implementation, the latter stop condition acts when the number of nodes in the search tree reaches a maximum user-set value *MaxTreeSize*.

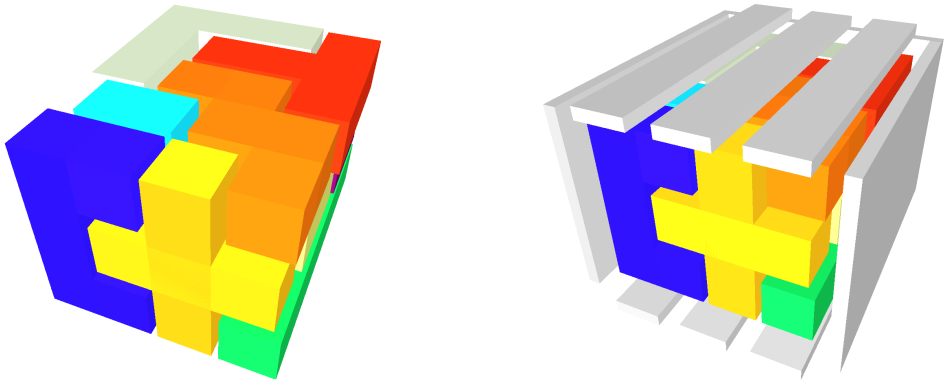
The information encoded in the constructed search tree is then treated. If the current active part has been disassembled, the function `SelectEndConf` returns the last computed configuration. Otherwise, it returns the configuration that maximizes the distance to the other parts. Except if this end configuration is too similar to the initial one (i.e. parts have only slightly moved), the path connecting both configurations is computed from the search tree, and it is merged to the sub-paths obtained in previous iterations. In order to avoid including unnecessary part motions in the solution disassembly path, the threshold used in function `TooSimilar` is initialized with a high value. This value is then correlatively decreased if parts cannot be disassembled after a consecutive number of I-ML-RRT iterations. Finally, the initial configuration for the next iteration is updated to the end configuration of the current one. The whole process is iterated until all parts are disassembled, or the number of iterations reaches a maximum value, *MaxIter*. In the latter case, the algorithm returns failure.

The output of the I-ML-RRT algorithm is a path S consisting of a sequence of elementary part motions yielding the system disassembly. This path results from the concatenation of the sub-paths s_i obtained at each iteration. Each sub-path s_i involves motions of one part (the active part in the corresponding iteration), and, in some cases, slight motions of other parts that hinder its disassembly. In easy disassembly problems, most of the computed sub-paths yield the entire disassembly of one part. Nevertheless, in more difficult problems, some of the computed sub-paths may only produce partial disassembly motions that increase the clearance of the active part. The number of such intermediate sub-paths tends to increase with difficulty of the problem.



(a) (b) (c)

Figure 6.3: Planar Puzzles: (a) Simple, (b) Double, (c) Triple.



(a) (b)

Figure 6.4: The Pentomino Puzzles: (a) without box, (b) with box.

Note that, although optimality criteria are not directly considered in this work (this is a possible extension mentioned in Section 6.5), the I-ML-RRT algorithm tends to minimize the the number of elementary motions (i.e. the number of explored leaves C_j^a defined in Section 6.2) required for the disassembly. Furthermore, in general, the obtained sequence is sub-optimal in the number of necessary elementary motions required for the disassembly. However, experimental results presented in next section show that even in the worst case, the "brute" solution contains a limited number of motions. Some unnecessary motions can be removed by simple post-processing of the brute path sequence.

6.4 Results

The I-ML-RRT algorithm has been also implemented within the motion planning software Move3D [Siméon et al. 2001], and tested with a set of benchmark examples. The results reported in the next section aim to illustrate both the generality and the good overall performance of the method. We start with the description of the puzzle models used as examples of both monotonic and non-monotonic disassembly problems.

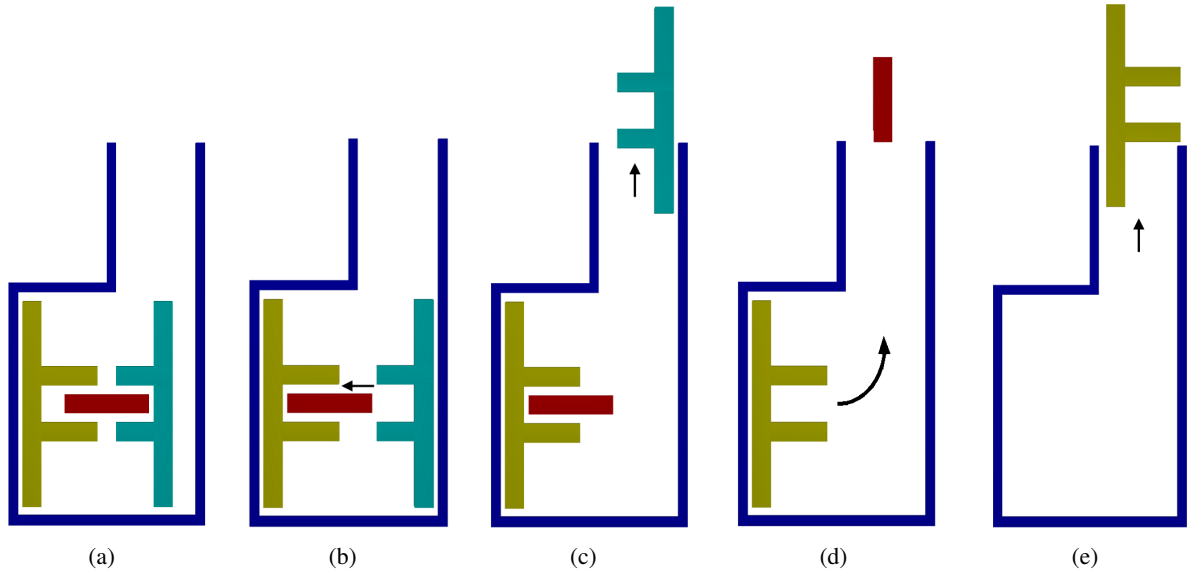


Figure 6.5: 2D Narrow Puzzles: (a) Assembled configuration, (b)(c)(d)(e) Four disassembly sub-paths.

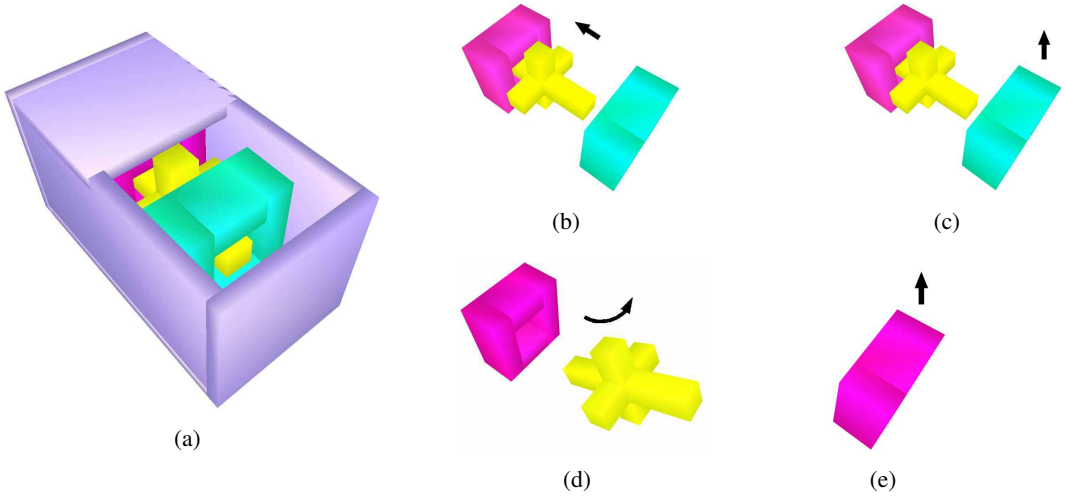


Figure 6.6: 3D Narrow Puzzles: (a) Assembled configuration, (b)(c)(d)(e) Four disassembly sub-paths (the box is not displayed for clarity reasons).

The intrinsic performance of the algorithm is then analyzed and compared with other existing techniques [Sundaram et al. 2001] and [Fogel and Halperin 2008]. All reported results correspond to averaged values based on 50 runs. Note that, this algorithm was run with the *MaxIter* parameter set to a sufficiently large value (10.000), so that each run ended up when a solution was found. All experiments were performed on a computer equipped with an AMD Opteron 148 2.0 GHz processor and 2 Gb of memory.

6.4.1 Benchmark Models

Monotonic Cases:

The two first models, *Planar* and *Pentomino* (see Fig. 6.3 and Fig. 6.4), correspond to the simpler class of

Table 6.1: Algorithm Performance.

Example	2D Simple *Transaltion*	2D Simple *Rotation*	Pentomino *Transaltion*	Pentomino *Rotation*	2D Narrow	3D Narrow
Nb.Parts	6	6	12	12	3	3
Nb.DOF	12	18	36	72	9	18
Ave.Time(s)	0.1	0.16	3.79	6.55	1.65	13.83
Std.Dev.	0.05	0.14	1.05	1.14	1.29	9.94

monotonic disassemblies in which each part can be extracted using a single continuous path (i.e. the length of the optimal disassembly sequence is equal to the number of parts, minus -the static- one). Both models are inspired from Sundaram's benchmarks [Sundaram et al. 2001] (Fig. 6.3-a and Fig. 6.4-a) from which we created more complicate variants for the aim of our tests. First, the number of parts of the *Planar* benchmark was increased by duplicating the original *Simple* model into the *Double* and *Triple* variants (Fig. 6.3-b,c), respectively involving 12 and 18 parts. We also considered a more difficult version or the three-dimensional *Pentomino* puzzle in which the disassembly motions of the twelve parts are constrained by the presence of a bounding box (Fig. 6.4-b).

Non-Monotonic Cases:
The two other examples, *2D_Narrow* and the three-dimensional variant *3D_Narrow* (respectively shown in Fig. 6.5 and 6.6), correspond to more complex problems, although less parts are involved. First, the disassembly sequence of both problems is non-monotonic, since one part has to be moved first to some intermediate position for "unlocking" the disassembly (see Fig. 6.5-b and 6.6-b). Moreover, due to spatial constraints, both disassembly sequences require rotational motions for extracting the second part from the narrow corridor (see Figure 6.5-d) or from the assembly box (see Figure 6.6-d).

6.4.2 Performance Analysis

Algorithm performance:
Table 6.1 shows the computing times required by I-ML-RRT to solve the test examples. The results show the good performance of the method that is able to solve all disassembly problems in times ranging from 0.1 sec. to 13 sec. Note that even if the *Planar* and *Pentomino* puzzles can be both solved with only translational motions, the table also reports computing times obtained when rotations are allowed. These results indicate that the overhead of considering rotational motions is rather limited (factor less than two). Further results reported in Table 6.2 for the three *Planar* puzzle variants also indicate that the computational efficiency is linearly influenced by the number of parts.

Effect of the disassembly order strategy:
As explained in Section 6.3, the current implementation of the *SelectPart* function relies on a simple random strategy for selecting at each iteration the active part tried to be disassembled. Table 6.2 further investigates the efficiency of this random choice in comparison to best and worst case selections. The

Table 6.2: Effect of the Sequence Initialization.

Model		Nb. Elemental Motions	Aver. Nb. Iterations	Aver. Nb. Subpaths	Average Time(s)	Standard Deviation
2D Simp Tran	Good-Bad	6	6-180.95	6-6.95	0.08-0.14	0.04-0.12
	Random	6	70.6	6.59	0.1	0.05
2D Doub Tran	Good-Bad	12	12-212.55	12-12.56	0.66-0.88	0.06-0.37
	Random	12	180.1	12.51	0.72	0.2
2D Trip Tran	Good-Bad	18	18-318.89	18-21.61	3.45-4.52	0.38-0.34
	Random	18	217.26	18.72	3.84	0.71
3D Pentomino	Good-Bad	12	12-68.7	12-15.96	4.52-7.73	0.92-1.36
	Random	12	58.7	15.73	6.55	1.14
2D Narrow	Good-Bad	4	4-370.67	4-158.61	0.71-1.84	0.29-1.29
	Random	4	36.62	10.95	1.65	1.29
3D Narrows	Good-Bad	4	4-455.54	4-90.47	10.1-21.96	4.51-4.56
	Random	4	177.19	19.16	13.83	9.94

best case corresponds to the correct sequence order given as input to the algorithm, while the worst case corresponds to the inverse sequence. The reported results indicate that the best/worst case selections only influence the computational efficiency by a factor less than 2.5 and that the random selection strategy stands in the middle of the time range. The table also reports the averaged number of iterations together with the number of extracted sub-paths (i.e. the length of the computed disassembly sequence). As one can see from the results, the computed solutions are close to the optimal ones for the monotonic disassemblies, but remain suboptimal for the two last non-monotonic examples. While some post-processing of the solution paths could certainly filter unnecessary motions in such cases, improving the optimality of the disassembly sequence clearly remains an issue to be further investigated.

Influence of Parameters:

In the I-ML-RRT algorithm, the *MaxTreeSize* parameter (see Section 6.3) is used to control the maximal RRT's size allowed when searching for a given part disassembly motion. When this limit is exceeded, a new part is selected and tried in turn for extraction. The maximal size of the search trees may therefore be seen as an important parameter, possibly influencing the overall efficiency of the method. The analysis on the influence of this parameter, whose results are summarized in Figure 6.7, indicates that it actually has a very limited impact, and thus, it can be easily tuned. The curves of Figure 6.7 display for each test example the evolution of the computing time as a function of *MaxTreeSize* for values ranging from 10 to 1000. As one can see, the performance is only affected for very small values (in the range [10,100]) resulting in search trees that are insufficiently rich for finding the whole feasible motion of a given part in a single iteration. The performance rapidly increases once the size becomes sufficient (more than 100 nodes), and then remains constant within most of the range. Thus, the *MaxTreeSize* parameter has to be preferably set with a large value and does not need any specific tuning since it does not really impact the overall performance. All experiments reported in the next section were performed with *MaxTreeSize* set to 200.

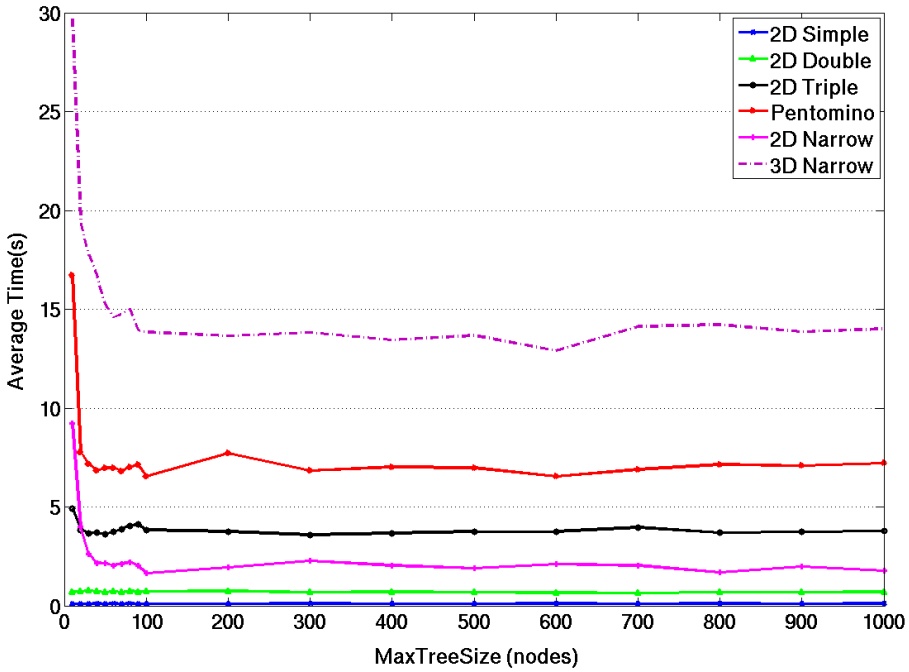


Figure 6.7: Impact of *MaxTreeSize* in the global performance of I-ML-RRT.

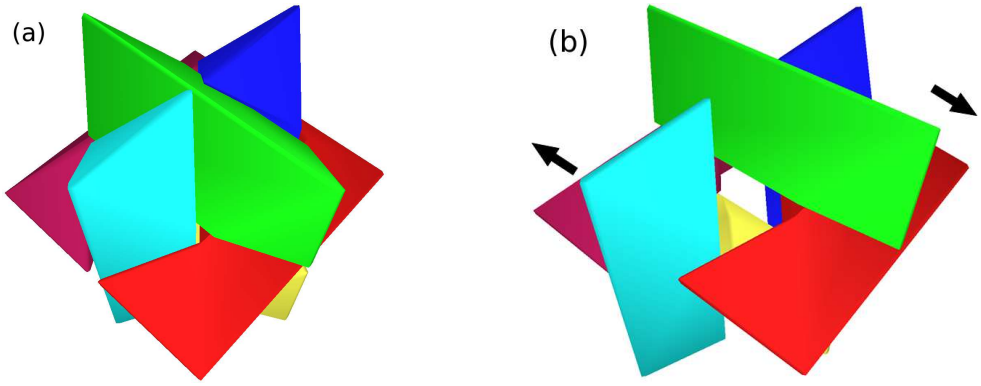
Table 6.3: Performance Comparison with Sundaram's Approach.

Model	Average Time(s)	
	Iterative-MLRRT	Sundaram *et al.*
2D Simple Tran	0.1	**474**
2D Simple Rot	0.16	
Pentomino Tran	3.79	**21794**
Pentomino Rot	6.55	

Performance Comparison:

In this section, we compare the performance of our algorithm with two other techniques [Sundaram et al. 2001],[Fogel and Halperin 2008] previously proposed for disassembly planning. The method of Sundaram *et al.* [Sundaram et al. 2001] is the most closely related to ours, since it also builds on a sampling-based path planning approach. Table 6.3 relates computing times reported in [Sundaram et al. 2001] for the $Planar(Simple)$ and *Pentomino* benchmarks (see Fig. 6.3-a and 6.4) to the averaged times obtained with the I-ML-RRT algorithm. The reported times indicate a huge gain factor (more than 4000) in favor to I-ML-RRT. Such direct time comparison is of course biased by the higher speed of our AMD Opteron 2 Ghz processor compared to the older processor used in [Sundaram et al. 2001]. Considering a CPU speed factor of about 10, the algorithmic performance of I-ML-RRT remains superior by at least two orders of magnitude.

We also ran the algorithm on the difficult puzzle benchmark used in a recent publication by Fogel *et al.* [Fogel and Halperin 2008]. This *DiagonalStar* puzzle (see Fig. 6.8) consists of six identical parts that create together a highly constrained assembly. The difficulty of this benchmark is that it requires coordinated motions between groups of parts that need to gradually move together for disassembling the



(a) (b)

Figure 6.8: The Diagonal Star Puzzle: (a) Assembled configuration, (b) Coupled disassembly motions.

Table 6.4: Performance Comparison with Fogel's Approach.

Model	**Iterative ML-RRT**			**Fogel *et al.***		
Diagonal Star (6 parts)	Clearance (size reduction)			Nb convex sub-parts per part		
	2%	5%	10%	3	5	8
Aver.Time(s)	49.25	36.83	13.03	4.89	13.62	41.59

puzzle. Table 6.4 provides some comparative results between I-ML-RRT and the exact algorithm proposed in [Fogel and Halperin 2008] for the specific class of disassemblies solvable with infinite translational motions. As explained in Fogel's paper, the algorithm requires each part to be decomposed into convex sub-parts and the computing time strongly increases with the total number of sub-parts. On the other hand, the I-ML-RRT algorithm does not rely on exact computations of contact motions and thus requires some "clearance" between the parts. The clearance corresponds to the reduced object size percentages which allows a collision-free assembled complex. Table 6.4 compares the "clearance dependent" computing times of I-ML-RRT with the "part complexity dependent" times reported in [Fogel and Halperin 2008]. Interestingly, although I-ML-RRT was not specifically developed for problems in which subassemblies of several parts have to slide together to disassemble each other, it is able to solve the *DiagonalStar* puzzle almost as efficiently as the exact algorithm designed for this class of problems. A group of three parts must be sliced together as a unit for disassembling this 6-part puzzle. Furthermore, this result is totally consistent with the demonstration of Natarajan [Natarajan 1988] that any sets of three-dimensional disjoint convex objects where no single convex object could move without disturbing other objects, but half of the objects could be moved together as a unit.

Another example created by Snoeyink *et al.* [Snoeyink and Stolfi 1994] contains 6 identical convex sticks that can not be taken apart with two hands using only translational motions (see Fig. 6.9). We also applied our method on this difficult puzzle benchmark with a little reduction of the sticks' size (0.9 %) in order to avoid the contact (remind that contact motions are not allowed in our implemented method). Although this modification changes the nature of the problem, this example shows that our method is able

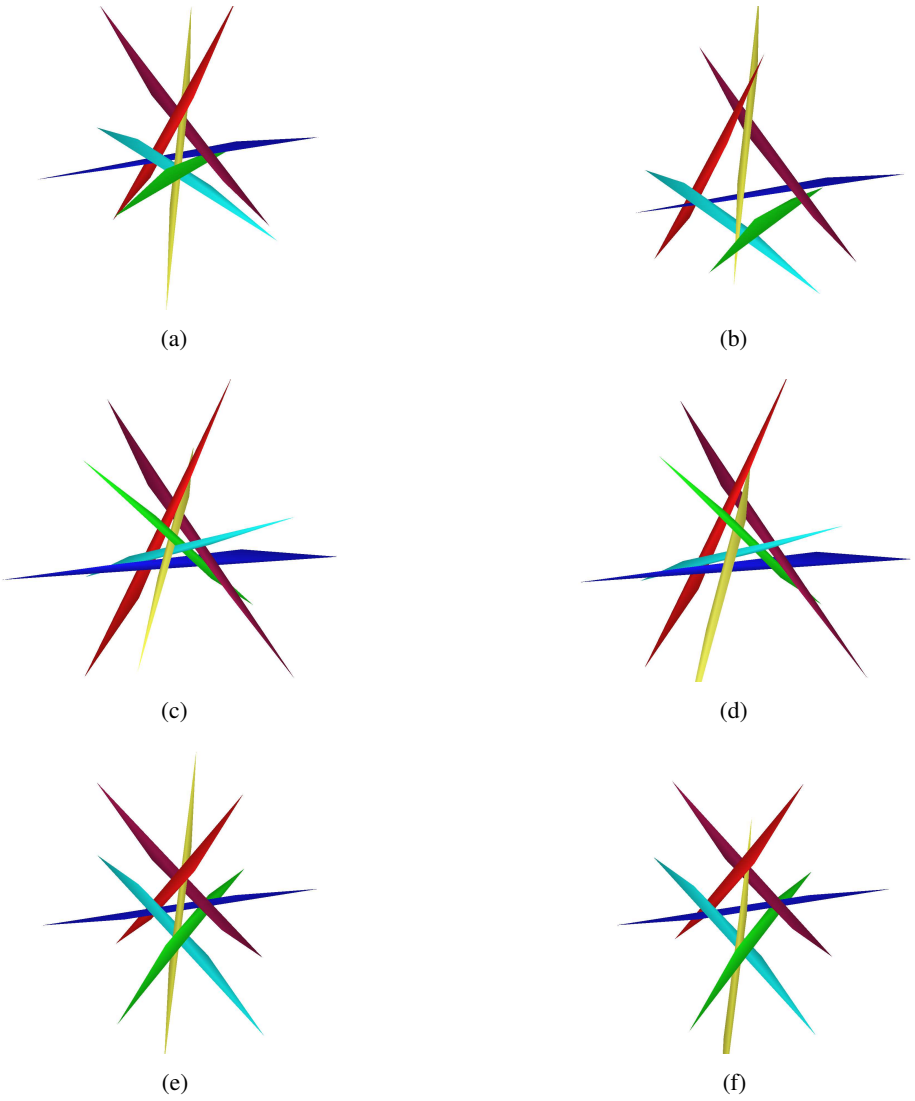
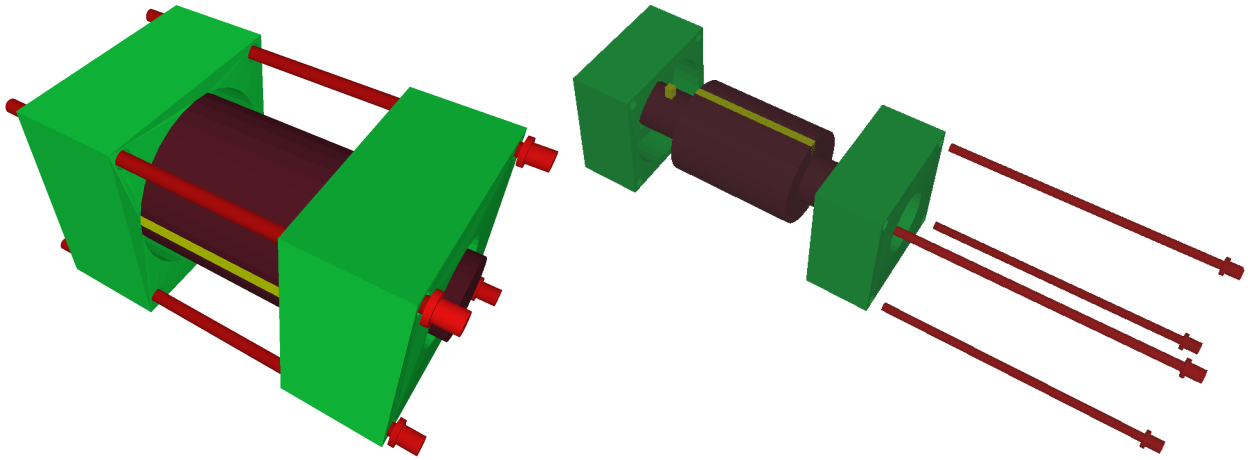


Figure 6.9: The Twisted Tetrahedron [Snoeyink and Stolfi 1994] with three different views. In this 6 identical part model, no single convex part could move without disturbing other parts. (a)(c)(e) Three views of the initial configuration, (b)(d)(f) Three corresponding views of the configuration before the disassembly of the complex. Coordinated motions of 6 sticks extended the volume of the tetrahedron in order to attempt the disassembled configuration.

Table 6.5: Performance on the Twisted Tetrahedron.

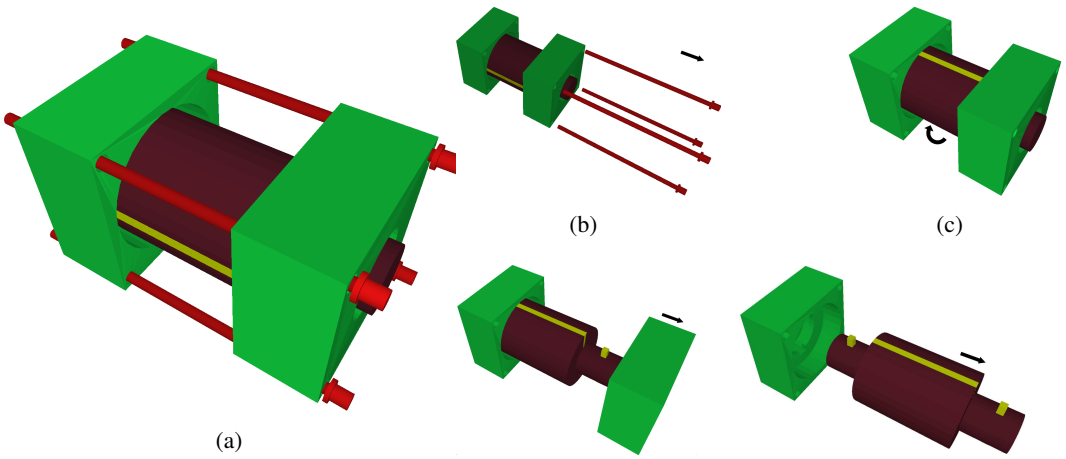
Example	Nb. Parts	Nb. DOF	Ave. Nb. Sub-paths	Ave. Time(s)	Std. Dev.
Twisted Tetrahedron	6	18	47.7	10.42	13.71

to solve such a very constrained disassembly benchmark. Table 6.5 shows the efficiency of the algorithm that requires a few seconds of computing time. Interestingly, the obtained paths show that the 6 sticks must be moved together to extend the volume of the tetrahedron, and then can be taken apart one by one.



(a) (b)

Figure 6.10: Illustration of 7-part engine assembly: (a) Assembled State, (b) Disassembled State.



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

Figure 6.11: Disassembly Sequence of the 7-part Simplified Engine: (a) Assembled configuration, (b)(c)(d)(e) Four disassembly sub-paths. Sub-path (c) makes the disassembly sequence be non-monotonic.

6.4.3 Realistic Examples

This section presents results on the application of the I-ML-RRT to compute the disassembly sequence and paths for two engine models: 7-part engine and 15-part engine with two different initial configurations that lead to monotonic and non-monotonic disassembly sequences.

Model Description:

The first simplified engine model contains 7 parts. Figure 6.10 shows the assembled and disassembled states of this model. Both translation and rotation motions are required. Figure 6.11 shows the disassembly sequence of the engine, which consists of 7 elementary motions: the translations of four screws and rectangular part, the rotation of the cylinder. Note that the necessary rotation of the cylinder before the extraction of the rectangular part makes this sequence be non-monotonic. The second model contains 15

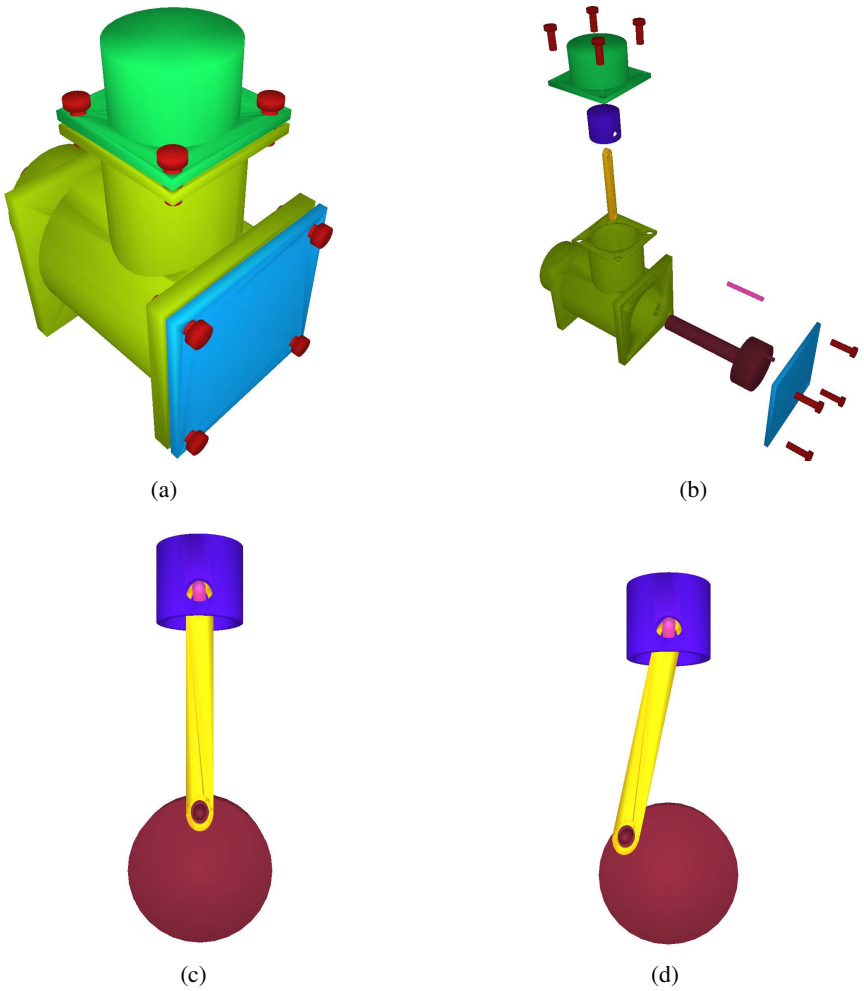


Figure 6.12: Illustration of 15-part engine assembly with two 4-inside-part configurations: (a) Assembled State, (b) Disassembled State, (c) The first configuration of the monotonic sequence, (d) The first configuration of the non-monotonic sequence.

parts. Figure 6.12 shows the assembled and disassembled states of this model with two different initial configurations of 4 internal parts. The initial configuration represented in Figure 6.12.c permits a monotonic disassembly sequence of the engine. Therefore, the disassembly sequence from this configuration contains optimally 14 elementary motions illustrated by the sequence of images a-b-d-e-f-g in Figure 6.13. However, the initial configuration in Figure 6.12.d leads to a non-monotonic sequence. Indeed, the disassembly of the 4 internal parts, which from a kinematic chain, requires a coordinated motion (illustrated in Figure 6.13.c) that enables the extraction of the wrist pin that links the piston and the rod.

Algorithm Performance:

Table 6.6 summarizes the computational performance of the algorithm averaged over 10 runs. The results show a good performance of the method that is able to solve all these difficult problems requiring both translation and rotation motions in a reasonable computing time (23.67, 170.06, and 391.7 seconds). The average number of generated sub-paths for the 7-part engine, whose disassembly sequence is non-

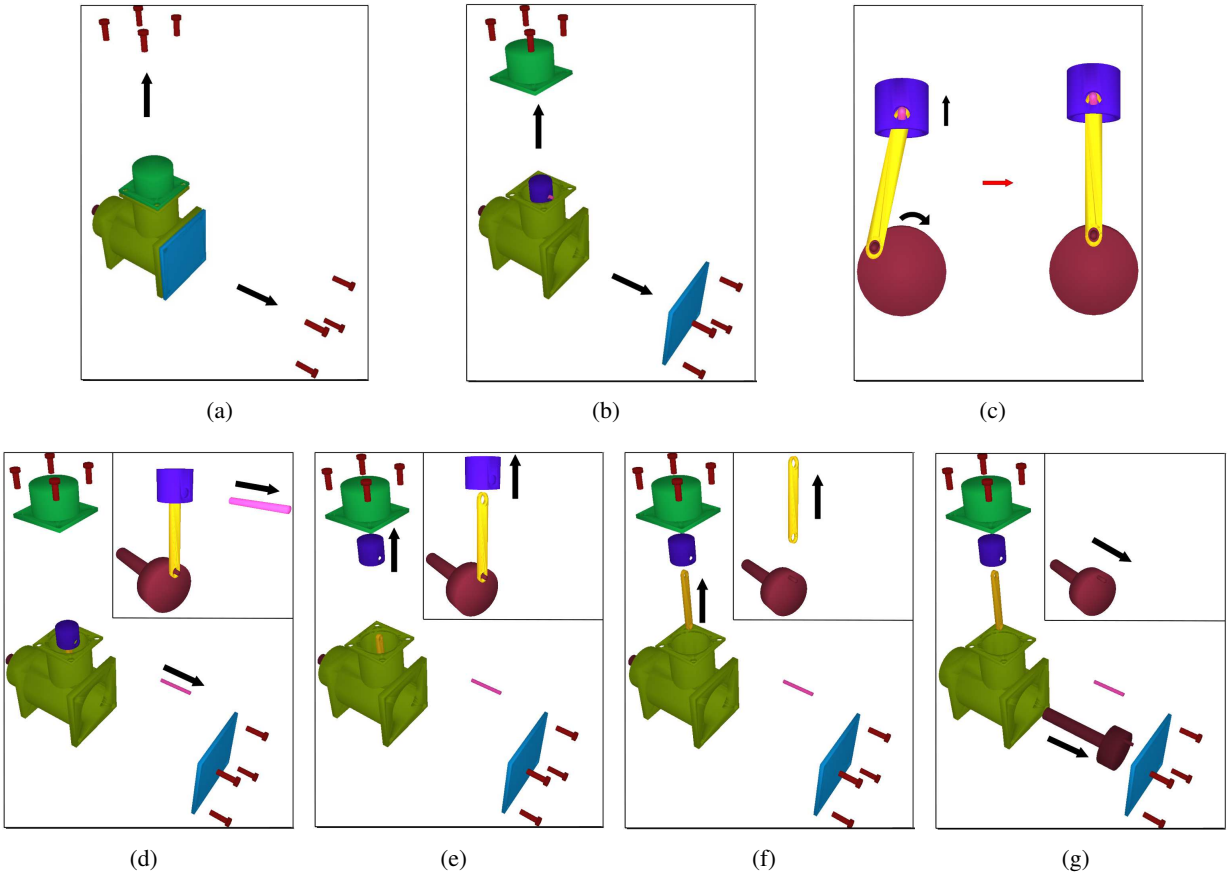


Figure 6.13: Disassembly Sequences of the 15-part Simplified Engine. Monotonic sequence: The disassembly of each part does not requires any intermediate motion of other parts. The monotonic sequence is composed of the motions in (a)(b)-(d)(e)(f)(g). Non-monotonic sequence: In order to disassemble the four internal parts, some coordinated motions are required to transform the first configuration of the non-monotonic sequence to the first configuration of the monotonic sequence, where the wrist pin (the pink part) can be extracted (c). The non-monotonic sequence is composed of the motions in (a)(b)(c)(d)(e)(f)(g).

Table 6.6: Performance Analysis for Simplified Engine Problems.

Example	Nb. Parts	Nb. DOF	Ave. Time(s)	Ave. Nb. Sub-paths
7-part Engine	7	7	23.67	7.83
15-part Mono Engine	15	22	170.06	15.2
15-part Non-Mono Engine	15	22	391.68	290.7

monotonic, is very close to the optimum. This is also the case for the monotonic disassembly sequence of the 15-part engine. However, the number of computed sub-paths for the non-monotonic sequence of the 15-part engine is far from the optimum (15). This degradation can be explained by the need of a coordinated motion of the 4 internal parts that is decomposed by our method in a long sequence of small motions.

6.5 Discussion

We have presented a general approach to disassembly planning, which treats sequencing and path planning problems simultaneously. The algorithmic solution we have developed is based on an extension of RRT, which is a simple and efficient path planning algorithm. Nevertheless, the proposed formulation can be seen as a general framework that should enable the development of other classes of algorithms. The method has been presented considering that parts move one by one. However, active and passive parameter subsets in I-ML-RRT may involve an arbitrary number of parts. Thus, the algorithm could be directly applied for treating problems involving any number of parts simultaneously moving in different directions (i.e. disassembly problems with any number of hands).Furthermore, it will be easy to modify the method for treating more efficiently problems requiring ensemble motions of groups of parts.

The presented algorithm does not need geometric operations such as computing normals to faces for determining blocking directions. It only requires a collision checker. Consequently, I-ML-RRT can solve problems involving complex part models with any shape (convex or concave), which may be the bottleneck for other methods. However, in particular cases (e.g. polyhedral objects), there is place for improvement by integrating some geometric operations that will provide good heuristics for selecting parts to move and suitable motion directions. Indeed, a possible trend for future work would be to combine I-ML-RRT with automated geometric reasoning methods for assembly partitioning and disassembly sequencing (e.g. [Wilson et al. 1995; Halperin et al. 2000; Fogel and Halperin 2008]).

Another future extension would be to consider optimality criteria, in particular, for minimizing the number of elementary motions. Although this criterion is already indirectly considered by the algorithmic design of I-ML-RRT (which does not keep part motions below a given distance threshold), it reminds an important issue that merits a more direct treatment. Besides, some costs associated with the disassembly could be considered within the path planning process. One way to do it could be to combine ideas of I-ML-RRT and T-RRT [Jaillet et al. 2008]. One particular instance of this combination of both algorithms is implemented in the next chapter to treat protein complex dissociation.

7

Protein Complex Disassembly

Specific interactions between macromolecules, that self-assemble into complex structures, play an important role within cells or organisms. However, little is known about how the subunits are temporally and spatially coordinated to form functional complexes [Stenberg et al. 2007]. This chapter addresses the inverse problem, that is, the study of protein complex dissociation. Given a protein complex, the problem consists in investigating the disassembling process into the elementary monomers (see Figure 7.1). Determining the disassembly order and identifying the most important events in the mechanism of denaturation are the two main questions to be solved. Also, following the assembly-by-disassembly paradigm like in manufacturing engineering, we can expect to gain some insights into protein assembly from the analysis of the disassembly process.

As we mentioned in Chapter 3, very few computational approaches have been proposed to analyze the dissociation of protein complexes. This chapter presents a method to compute the sequential disassembly pathways of protein complexes. The proposed algorithm, called Iterative-ML-Transition-RRT (I-ML-T-RRT), introduces ideas from T-RRT [Jaillet et al. 2010] (see the short description of this algorithm in Section 3.2.3) within the I-ML-RRT algorithm described in Chapter 6. The I-ML-RRT algorithm has been shown to be a general method to disassembly planning, which treats sequencing and path planning problems simultaneously, and the T-RRT algorithm, which is inspired by Monte Carlo search techniques, was introduced to compute low-cost paths in high-dimensional search-spaces. Combining these two algorithms, I-ML-T-RRT computes the sequence of paths leading to the complex disassembly while trying to minimize the variation of the interaction energy between subunits of the considered complex.

Next section presents the proposed methodology for computing protein complex disassembly sequences and paths. Then preliminary results described in Section 7.2 illustrate the capacities of the method on three systems for which experimental results on the association or dissociation process are available: Cytochrome

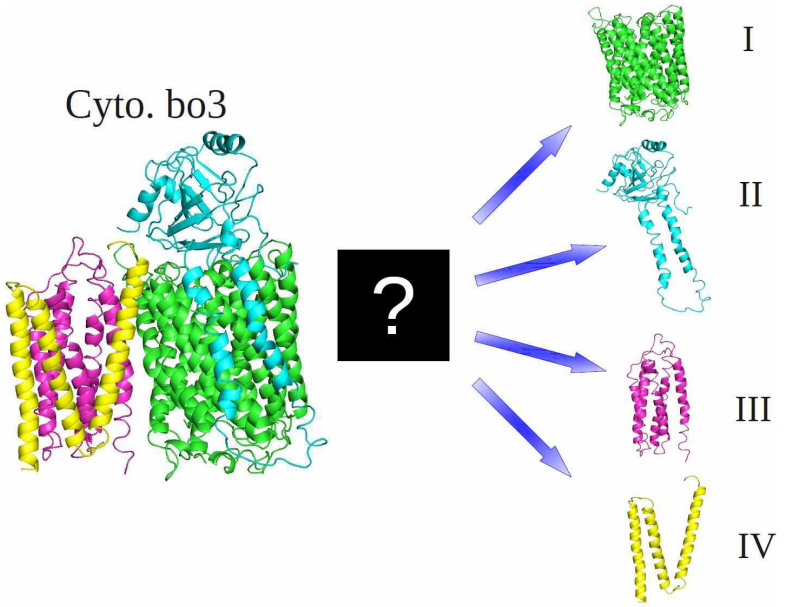


Figure 7.1: Illustration of Cytochrome bo_3 Oxidase structure (PDB ID 1FFT) and its four sub-units. What is the disassembly process ?

cbb_3 Oxidase, Cytochrome bo_3 Ubiquinol Oxidase, and Homotetrameric Protein Transthyretin Finally, the last section 7.3 will discuss the current limitations and some possible improvements of the method.

7.1 Methodology

The methodological contribution of this chapter is an extension of the I-ML-RRT algorithm (Chapter 6) for treating protein complex disassembly problems. Like in the methodology for computing protein-ligand access/exit paths presented in Chapter 5, a mechanistic model of proteins is considered. However, interaction energies are also considered within the exploration algorithm. The description of the model and the proposed method are detailed next.

7.1.1 Model and Parameters

Mechanistic molecular model:
The combined method called I-ML-T-RRT also deals with partially-flexible protein models represented as articulated mechanisms detailed in Chapter 5. Figure 7.2 illustrates the mechanistic model of a protein complex. Each sub-unit is modeled as a rigid group G_i which holds free rigid body mobility, independently from the other groups. Side-chains (not represented in the figure) are generally modeled as flexible elements with rotatable bond torsions, but they can also be defined as rigid parts by the user. Therefore, this model does not allow to treat folding or unfolding problems during the (dis)assembly process.

Conformational parameters:
The protein complex conformation is defined by the parameters determining the pose (position and

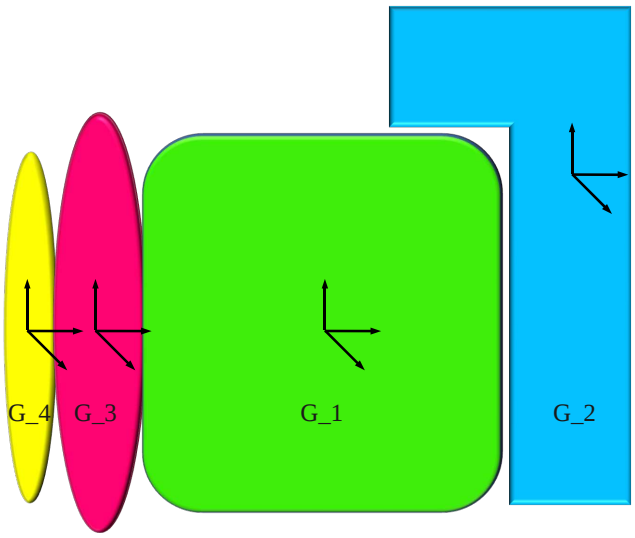


Figure 7.2: Schematic representation of a protein complex model. Four rigid proteins with their corresponding Cartesian coordinate frames are denoted from G_1 to G_4. Flexible side chains are not shown.

orientation) of all the groups G_i and the bond torsions of all flexible side-chains. Following the formulation detailed in Chapter 6, a different active/passive partition of the conformational parameters is considered at each iteration of the disassembly sequencing algorithm. At each iteration, the pose parameters of some groups G_i (ensemble motion of several sub-units can be considered) are active, while those of the others are passive, together with the parameters associated with all the side-chains. In addition, the present extension also needs to deal with different mobility coefficients for passive parameters (see explanations on passive part mobility in Chapter 4). By default, the mobility coefficient of all side-chain torsions is set to 1.0, meaning that they will systematically move if they are identified during the exploration. Lower mobility coefficient is allowed to sub-unit poses, with $\delta = 0.2$ in the current implementation.

Energy Evaluation:

For energy evaluation within our method, we propose to use a coarse-grained force field in order to overcome the computational expensive cost of the all-atom representation when applied for studying interactions of large systems such as protein complexes. In particular, we have adopted the model proposed by Zacharias [Zacharias 2003; Zacharias 2005], which has proved good results in protein-protein docking studies. This model was first applied for treating rigid protein-protein docking problem, and then the approach was extended to deal with some flexible portions of proteins [Bastard et al. 2006]. Another work also adopted this coarse-grained representation to investigate mechanical properties of proteins [Lavery and Sacquin-Mora 2007].

Each residue is represented by up to four pseudo atoms (i.e. two for the backbone and up to 2 for the side chain). For a realistic description of the polar character of the protein backbone, two pseudo atoms located at the position of the N and O atoms of the backbone's carbonyl group are also used. Small amino acid side chains such as Ala, Ser, Thr, Val, Leu, Ile, Asn, Asp, and Pro are represented by one pseudo atom located at the geometrical center of all side chain heavy atoms. In case of longer amino acids such as Arg, Lys, Glu,

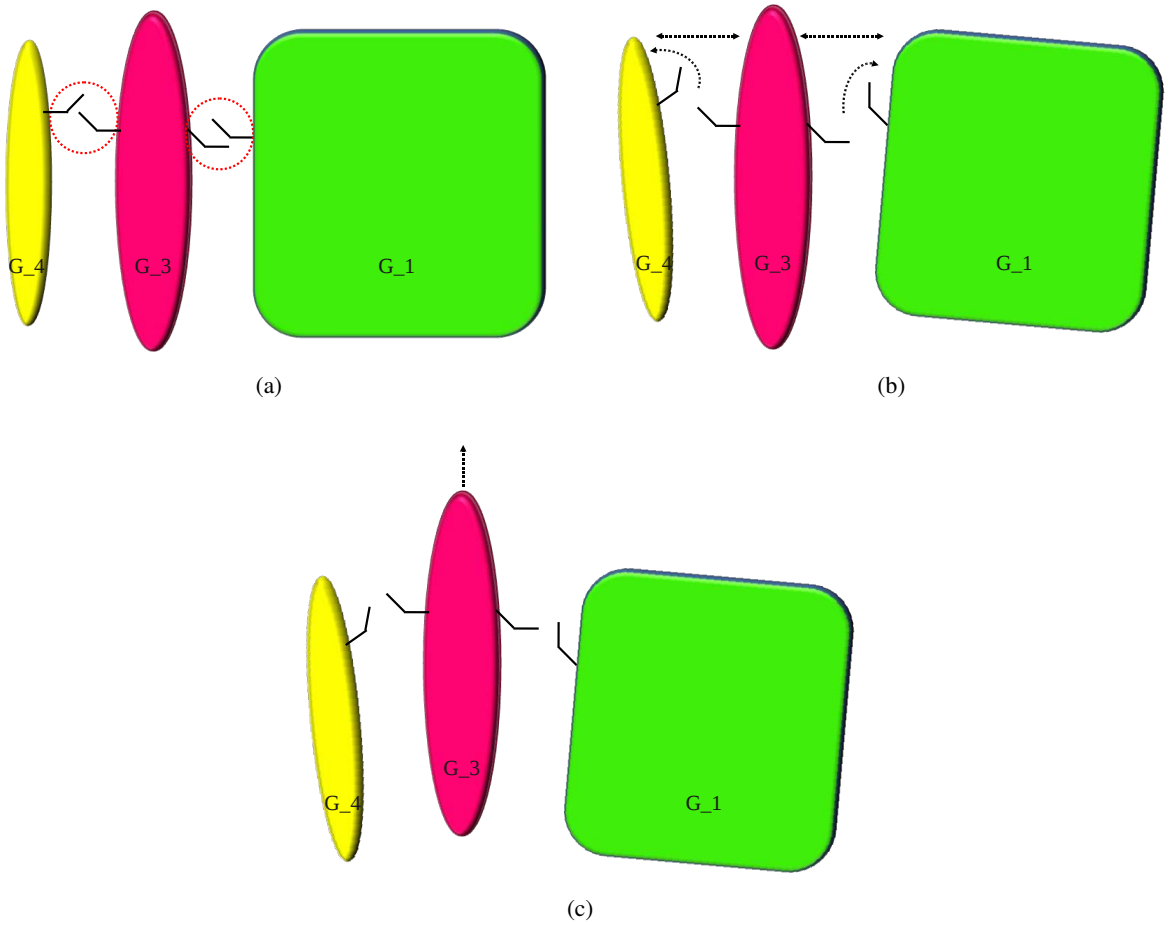


Figure 7.3: Illustration of the decoupled exploration of active and passive parameters within the I-ML-T-RRT algorithm. (a) Identification of the passive parts hindering the active subunit motion or "near-collision" passive parameters. (b) Expansion of the involved passive parts. c) Expansion of active parameters corresponding to the motion of active subunit.

Gln, His, Met, Phe, Tyr, Trp, it is necessary to use two pseudo atoms in order to maintain good agreement between energy-minimized complexes and geometrical constraints. The first pseudo atom is located in the middle between C_β and C_γ of each side chain, and the other one can be then calculated as the geometrical center of all other side chain atoms.

The total energy of the system is the sum of all pseudo-atom pair energies:

$$E_{sys} = \sum_{ij} (E_{LJ}(r_{ij}) + E_{elec}(r_{ij})) \tag{7.1}$$

The energy function between pseudo-atom pair i and j, belonging to different proteins, at the distance r_{ij} consists of the pairwise soft Lennard-Jones type potential (E_{LJ}) and an electrostatic interaction term (E_{elec}) with a distance dependent dielectric constant ($\varepsilon(r) = 15r$) for the interaction between charged residues.

These formulas below are used to calculate the two energy terms:

$$E_{LJ}(r_{ij}) = \frac{B_{ij}}{(r_{ij})^8} - \frac{C_{ij}}{(r_{ij})^6} \; and \; E_{elec}(r_{ij}) = \frac{q_i q_j}{\varepsilon r_{ij}} \tag{7.2}$$

where B_{ij} and C_{ij} correspond respectively to repulsive and attractive terms of Lennard-Jones potential which are computed as follow:

$$\begin{aligned} B_{ij} &= A_i A j (R_i + R_j)^8 \\ C_{ij} &= A_i A j (R_i + R_j)^6 \end{aligned} \tag{7.3}$$

where R and A represent the approximated size and the physio-chemical of side chains respectively. The values of R and A are detailed in [Zacharias 2003]. q_i and q_j are charged values of pseudo atom i, j respectively.

7.1.2 I-ML-T-RRT algorithm

The I-ML-T-RRT extends the algorithm I-ML-RRT by integrating a state-transition test that rejects new conformation based on the energy variation. For this, the construction algorithm is basically the same than the one detailed in Chapter 6 (Algorithm 4). The only modification concerns the main function `Construct_ML-RRT`, which is called at each iteration to compute the disassembly path of one part (a sub-unit in the present case). Within I-ML-T-RRT, this function is replaced by the function `Construct_ML-T-RRT`, which integrates the state-transition test inspired by T-RRT [Jaillet et al. 2010]. Algorithm 5 shows the pseudo-code of this new function. The overall of this function is based on the basic ML-RRT algorithm (see Algorithm 3) in Chapter 4. Figure 7.3 provides a simple illustration of the expansion process performed by this function, which alternates expansion attempts of the active and passive parameter subsets. The movement of the chosen active subunit is facilitated by the conformational changes of the side-chains and the poses of other passive subunits. The main difference with the basic ML-RRT algorithm concerns the introduction of two functions, `TransitionTest` and `MinExpandControl`, that control the evolution of the search-tree.

The `TransitionTest` function rejects some of the generated states if they do not correspond to energetically acceptable moves. Similarly to MC methods, the acceptance rule of a local move is defined by comparing the energy E_j of the new state with the energy E_i of the previous state (i.e. the parent node in the tree). This test is based on the Metropolis criterion, with a transition probability p_{ij} defined as follows:

$$p_{ij} = \begin{cases} exp(-\frac{\Delta E_{ij}}{kT}), & \text{if } \Delta E_{ij} > 0 \\ 1, & \text{otherwise} \end{cases} \tag{7.4}$$

where $\Delta E_{ij} = E_j - E_i$ is the energy variation between the two states (computed with the coarse-garined model), k is the Boltzmann constant, and T is the temperature. Note that the term temperature is employed in analogy with methods in statistical physics, but in our case it does not have any physical meaning. Indeed T is a parameter of the algorithm used to control the difficulty level of transition tests. The value of T is automatically tuned according to the information acquired during the exploration. When a maximum number of consecutive rejections is reached, T increases by a factor λ. Contrarily, each time an uphill

Algorithm 5: Construct_ML-T-RRT

input : the conformational space C;
the initial conformation q_{init};
the partition $\{P_{\text{act}}, P_{\text{pas}}\}$;
output : the tree τ;
begin
  $\tau \leftarrow$ `InitTree`(q_{init});
  while not `StopCondition`(τ) **do**
    $q_{rand}^{act} \leftarrow$ `SampleConf`(C, P_{act});
    $q_{near} \leftarrow$ `NearestNeighbor`(τ, q_{rand}^{act}, P_{act});
    if `MinExpandControl`(q_{near}, q_{rand}) **then**
      $P_{\text{pas}}^{\text{col}} \leftarrow$ `PassZoneCol`(q_{near});
      $q'_{near} \leftarrow q_{near}$;
      while $P_{pas}^{col} \neq \emptyset$ **do**
        $P_{\text{pas}}^{\text{mov}} \leftarrow$ `PartsToMove`($P_{\text{pas}}^{\text{col}}$);
        $q_{rand}^{pas} \leftarrow$ `PerturbConf`(C, q'_{near}, $P_{\text{pas}}^{\text{mov}}$, $q'_{near}.n_{\text{fail}}$);
        $(q_{new}, P'^{\text{col}}_{\text{pas}}) \leftarrow$ `Expand`(q'_{near}, q_{rand}^{pas});
        $P_{\text{pas}}^{\text{col}} \leftarrow P'^{\text{col}}_{\text{pas}} \setminus P_{\text{pas}}^{\text{col}}$;
        $q'_{near} \leftarrow q_{new}$;
      $(q_{new}, P_{\text{pas}}^{\text{col}}) \leftarrow$ `Expand`(q'_{near},q_{rand}^{act});
      if $q_{new} \neq NULL$
      and `TransitionTest`($c(q_{near})$,$c(q_{new})$,$d_{near-new}$)
      and not `TooSimilar`(q_{near},q_{new}) **then**
        `AddNewNode`(τ, q_{new});
        `AddNewEdge`(τ, q_{near}, q_{new});
        $q_{\text{near}}.n_{\text{fail}} \leftarrow 0$;
      else $q_{\text{near}}.n_{\text{fail}} \leftarrow q_{\text{near}}.n_{\text{fail}} + 1$;
end

transition test succeeds, T decreases by the same factor λ (see [Jaillet et al. 2010] for details).

The adaptive temperature tuning may however lead to bottleneck situations. The temperature T may be stabilized by the insertion of new states very similar to the ones already contained in the tree, whereas the expansion toward new regions of the space would require an increment of T. The insertion of such states only contributes to the refinement of the exploration in regions already reached by the tree. To overcome this drawback, the function `MinExpandControl` discards node expansion if the distance between the selected state q_{near} and the random state q_{rand} is smaller than the expansion step-size δ. Such a simple filtering avoids an excessive refinement of low-energy regions, therefore facilitating the tree expansion toward new regions of the space.

7.2 Preliminary Results

This section presents preliminary results obtained with the proposed method on three biologically interesting systems. The first one is a four-subunit complex called Cytochrome cbb_3 Oxidase, which is the second most abundant oxidases of heme oxidase family is analyzed. The second one is a very similar system: Cytochrome *bo*3 Ubiquinol Oxidase. The third complex, Homotetrameric Protein Transthyretin, contains four identical subunits. The method was implemented within our software prototype named BioMove3D. The program

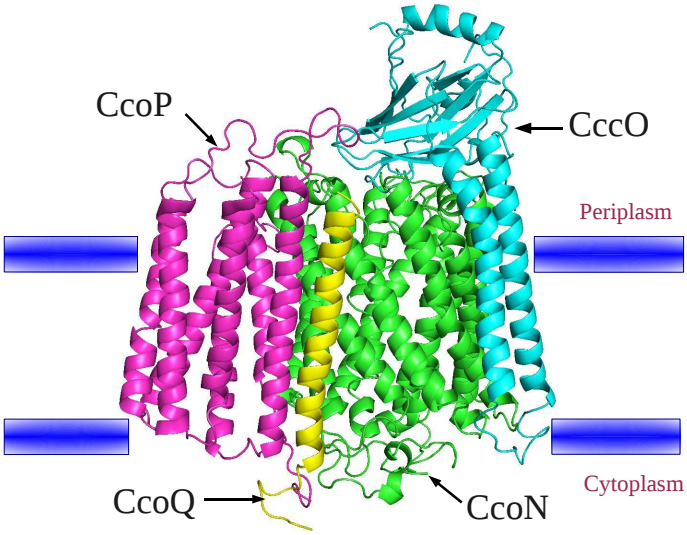


Figure 7.4: Structure of Cytochrome cbb_3 Oxidase: (PDB ID 1QLE): Subunit CcoN - Green, Subunit CcoO - Cyan, Subunit CcoP - Magenta, Subunit CcoQ - Yellow.

PyMOL [DeLano 2002] was used for viewing molecular models. The computing times reported below correspond to tests run on a computer equipped with an single AMD Opteron 2222 processor at 3.0 GHz and 8Gb of memory.

7.2.1 Cytochrome cbb_3-type Oxidase

7.2.1.1 Structural Description

The cbb3-type cytochrome oxidases (cbb_3-Cox), which are the second most abundant oxidases of the heme-copper oxidase superfamily, can be found in various proteobacteria under microaerobic conditions [Sharma et al. 2006; Peters et al. 2008]. This enzyme contains four subunits whose assembly information and interactions are still limited (see Fig. 7.4 extracted from PDB ID 1QLE [Sharma et al. 2010]). The biggest catalytic 61 kDa subunit CcoN has up to 14 putative transmembrane helices and binds heme B, where oxygen reduction takes place. Thus, the oxygen affinity of cbb_3-Cox is about five times higher than that of the other member of its family called aa_3-type cytochrome oxidase [Pitcher et al. 2002]. The two other membrane-bound c-type cytochromes (cyt *c*) (i.e., CcoO and CcoP) whose masses are 28 and 32 kDa respectively have been suggested to be part of the electron transfer chain from the donor cyt *c* to the catalytic binuclear center located inside the CcoN [Sharma et al. 2006]. The forth subunit CcoQ is comprised of a single transmembrane helix. This subunit is suggested to be necessary for protecting cbb_3-Cox against high oxygen concentrations under aerobic conditions in *Rhodobacter sphaeroides* [Oh and Kaplan 2002]. Two subunit CcoN and CcoO are common for all heme-cooper oxidases, but the two other ones (i.e., CcoP and CcoQ) are unique for cbb_3-type cytochrome [Sharma et al. 2006].

Table 7.1: I-ML-T-RRT's Performance on Cytochrome cbb_3 Oxidase.

Model	Av.Time(s)	Std Dev	N_{node}	N^{act}_{samp}
Cytochrome cbb_3	46804	19901.47	218	14786

7.2.1.2 Related Works

Several experimental methods have been used to investigate the assembly process of this system. In the work of Zufferey *et al.* [Zufferey et al. 1996], using Western blot analysis, CcoN and CcoO were shown to be inserted firstly into the membrane to form an apparently stable CcoNO core complex in *Bradyrhizobium japonicum*. The next step involved the assembly of the CcoNO core with the CcoP protein. The smallest subunit CcoQ was not required for the complex formation and therefore, did not play a key role for the complex association [Zufferey et al. 1996]. Another method has been presented to analyze the assembly process of cbb_3-Cox in *Rhodobacter capsulatus* by using blue-native polyacrylamide gel electrophoresis (BN-PAGE) along with activity staining of wild-type and mutant membranes [Kulajta et al. 2006]. Their results revealed an intermediate 210 kDa subcomplex (i.e., CcoNOQ) before recruiting subunit CcoP to form the final 230kDa active complex (CcoNOQP). The work of Peters *et al.* [Peters et al. 2008] has recently revealed the important role of subunit CcoQ in the stabilization of the interaction between subunit CcoP and the preassembled CcoNO core complex by combining activity measurements, BN-PAGE, and chemical cross-linking. Their results suggested that CcoQ can obviously bind to CcoP before CcoP associates with CcoNO core complex to create the stable form. Therefore, a possible assembly order of cbb_3-Cox is the formation of the core complex CcoNO containing two subunits CcoN and CcoO and the formation of the subassembly CcoP-CcoQ, following by the association of these two subassemblies.

7.2.1.3 Sequential Disassembly Pathways

In order to elucidate the disassembly process of cbb_3-Cox, 20 tests were performed with I-ML-T-RRT. These results show that the subassembly CcoPQ containing two subunits CcoP and CcoQ is preferred to be the first disassembled part from the whole complex with a high percentage of 80% (16/20 runs), following by the dissociation of two subassemblies (i.e., CcoPQ and CcoNO). Four other pathways display the first dissociation of the groups CcoQ-CcoO, CcoN-CcoP from the complex. We also computed the average interaction energy (using the aforementioned coarse-grained force-field model) for extracting the first subassembly from the complex: 197.91 kJ/mol for CcoPQ dissociation and 224.2 kJ/mol for the other cases (i.e., CcoQ-CcoO, CcoN-CcoP), respectively. Note that the work of Peters *et al.* [Peters et al. 2008] indicates that the two subunits CcoP and CcoQ are in close contact, and due to the lack of CcoQ, the CcoP's ability to assemble with the core complex CcoNO is drastically reduced. The preferred disassembly order of cbb_3-Cox calculated by our method is thus the reverse of the assembly order proposed by Peters *et al.* [Peters et al. 2008]. Note that the work of Sedlak et al [Sedlak and Robinson 2009] also suggests that the order of denaturant-induced subunit dissociation mimics the reverse of Cytochrome c Oxidase assembly sequence [Nijtmans et al. 1998; Taanman and Williams 2001]. Table 7.1 shows the computational performance of our method. The computational time for generating the sequencing pathways is about 13 hours on a single

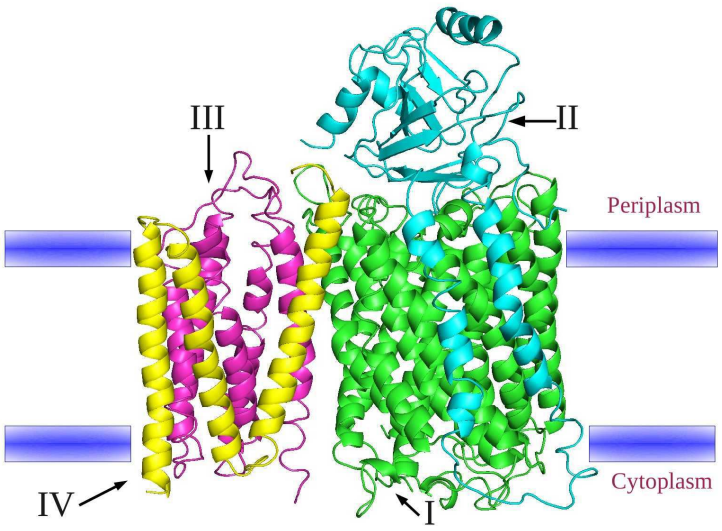


Figure 7.5: Illustration of Cytochrome bo3 ubiquinol oxidase (PDB ID 1FFT): Subunit I - Green, Subunit II - Cyan, Subunit III - Magenta, Subunit IV - Yellow.

Table 7.2: Subunit Masses of the Cytochrome bo_3 Ubiquinol Oxidase [Stenberg et al. 2007].

Subunit	**I**	**II**	**III**	**IV**
Mass (kDa)	75	33	22	12

processor.

7.2.2 Cytochrome bo_3 Ubiquinol Oxidase

7.2.2.1 Structural Description

Cytochrome bo_3 Ubiquinol Oxidase (CUO), which is the terminal oxidase in the aerobic respiratory chain of the *Escherichia coli*, belongs to the heme-copper oxidase superfamily. Consisting of four subunits (I, II, III and IV), this enzyme catalyzes the two-electron oxidation of ubiquinol-8 (Q_8H_2) at the periplasmic side and the four-electron reduction of oxygen to water at the other side of the membrane (i.e. cytoplasmic side) (see Fig. 7.5). In addition, CUO also plays a functional role as a proton pump by translocating protons across the cytoplasmic membrane in order to establish an electrochemical proton gradient. The generated transmembrane proton and voltage gradient is then converted to more useful energy forms through energy conserving systems such as the ATP synthase [Abramson et al. 2000].

7.2.2.2 Related Works

For investigating the complex formation of the CUO complex, Stenberg and Daley [Stenberg et al. 2007; Daley 2008] have recently proposed a novel pulse-labelling method using the rifampicin blocking technique and blue native (BN)-/SDS-PAGE. Figure 7.6 shows the sequential assembly order of the four-subunit

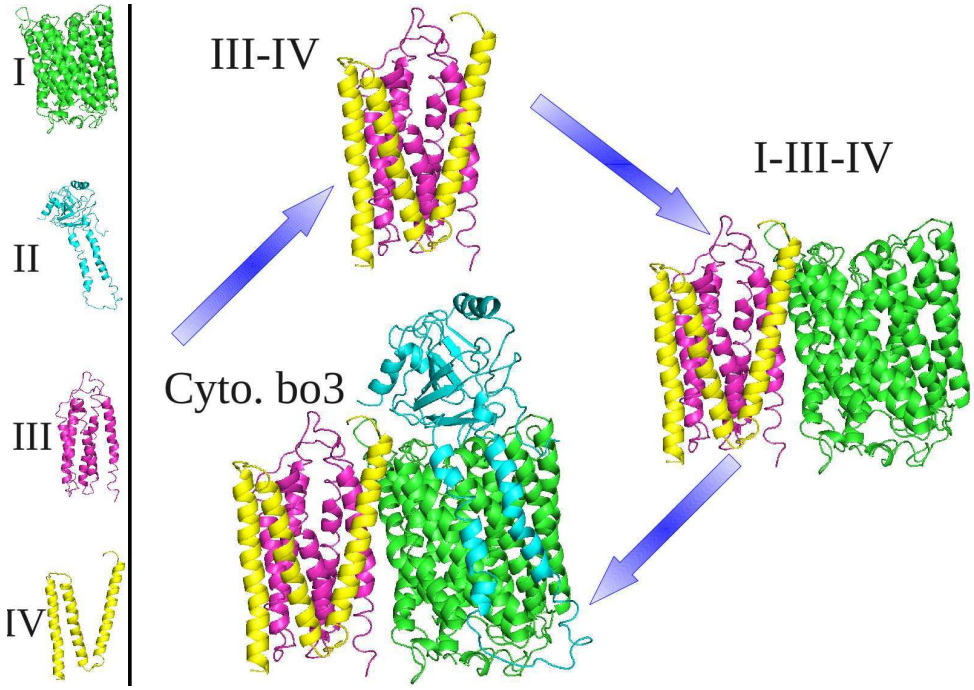


Figure 7.6: Assembly process of Cytochrome bo_3 Ubiquinol Oxidase: Subunit III and IV assemble first, followed by subunit I and finally subunit II [Stenberg et al. 2007; Daley 2008].

Table 7.3: I-ML-T-RRT's Performance on Cytochrome bo_3 Ubiquinol Oxidase Model.

Model	**Av.Time(s)**	**Std Dev**	$\mathbf{N}_{node}$	$\mathbf{N}^{act}_{samp}$
Cytochrome bo_3	27800.6	11261.2	321	9619

membrane complex obtained with this method, where the subunits III and IV firstly assemble to form an temporary subassembly, followed by subunit I and finally subunit II.

7.2.2.3 Sequential Disassembly Pathways

In order to investigate the disassembly pathway of this complex, 20 runs were performed with the I-ML-T-RRT algorithm starting from the assembled structure (PDB ID **1FFT**). The results show that the two subunits III and IV are alternatively preferred to firstly disassembled from the CUO complex with high percentage 90% (18/20 runs), followed by the dissociation of the left subassembly I-II. Only two runs in which the subunit I or subunit II were firstly extracted have been observed. The average interactions energies for extracting the two first subunits in the two cases are also computed: 160.98 kJ/mol for the dissociation of two subunits III and IV, and 192.46 kJ/mol for the dissociation of two subunits I and II. In terms of energy, this difference proved that the two small sub-units (III and IV) can be disassembled more easily than the two bigger sub-units (I and II). Thus, the preferred disassembly order calculated by our method is not the reverse of the assembly order proposed by [Stenberg et al. 2007]. This phenomena can be answered by the structural topology of the considered complex where the two subunits III and IV are situated around the big

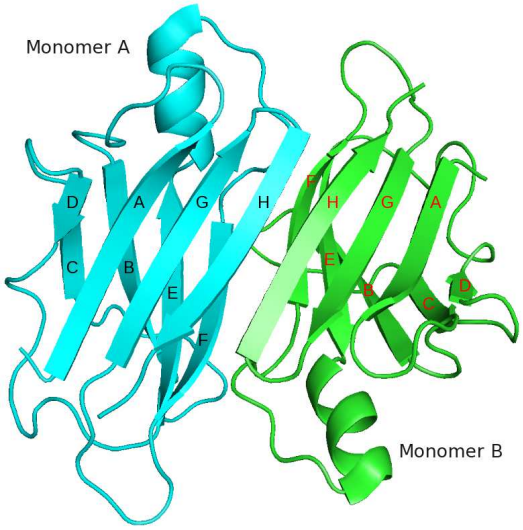


Figure 7.7: Structure of Dimer of the Transthyretin: (PDB ID 2PAB): Subunit A - Green, Subunit B - Cyan.

subassembly I-II that forms a stable core. Moreover, due to the mass of CUO subunits (see Tab. 7.2), subunits III and IV are significantly smaller than the other ones. Therefore, the two small subunits have more chance to be disassembled first with lower energy interactions. Table 7.3 shows the computational performance of our method. Overall, it took about 8 hours to compute all the sequential disassembly pathways of the CUO complex.

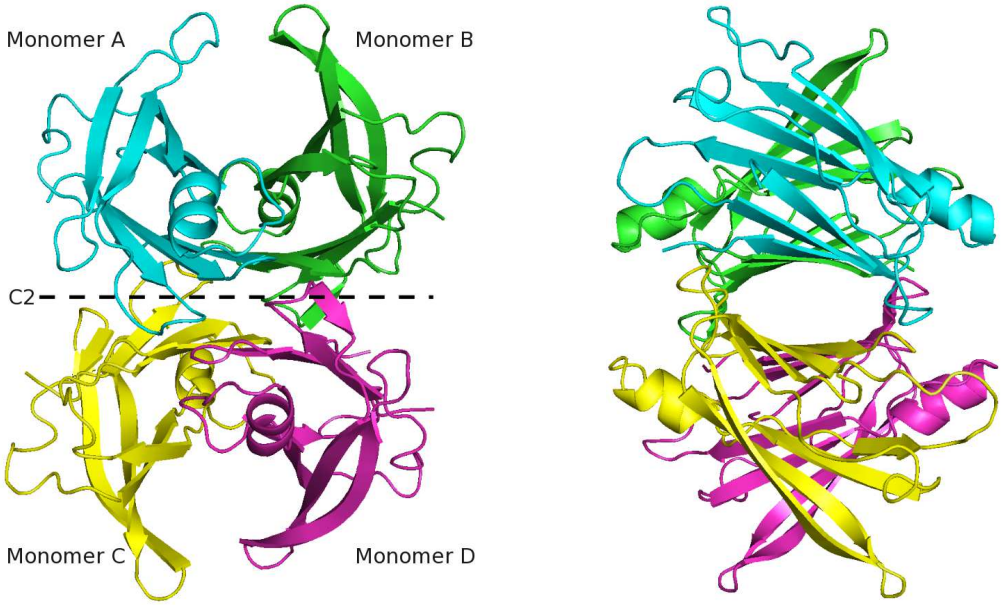
7.2.3 Homotetrameric Protein Transthyretin

7.2.3.1 Structural Description

Transthyretin (TTR), which belongs to the homotetrameric human plasma protein family, is composed of four 127 amino-acid monomers (14 kDa) [Foss et al. 2005]. Each monomer contains eight anti-parallel β-sheets, denoted from A to H, and a small α-helix. There are two sheets in the monomer that are organized in the β-barrel topology. The dimerization of two monomers is performed through an intermolecular main-chain interaction between H-strands from each monomer in order to form an eight-stranded β-sandwich [Hornberg et al. 2000]. Due to hydrophobic contacts between residues situated in the AB and GH-loops, two dimers are associated to form the tetramer that contains thyroxine binding sites in the large hydrophobic channel formed between the two dimers [Hornberg et al. 2000]. Figure 7.8 shows the structure of TTR in which the the two dimer AB and CD is symmetric over the crystallography axis C_2 represented by a dotted line.

7.2.3.2 Related Works

Foss *et al.* [Foss et al. 2005] proposed a study on the disassociation pathways of the homotetrameric protein transthyretin (TTR) which must undergo rate-limiting dissociation to its constituent monomers in order to enable partial denaturation that allows the process of amyloidogenesis associated with human pathology



(a) (b)

Figure 7.8: Structure of Tetramer Transthyretin (PDB ID 1QAB): Subunit A - Green, Subunit B - Cyan, Subunit C - Yellow, Subunit D - Magenta. (a) Front view, (b) Side view.

Table 7.4: I-ML-T-RRT's Performance on Transthyretin Model.

Model	**Av.Time(s)**	**Std Dev**	$\mathbf{N}_{node}$	$\mathbf{N}^{act}_{samp}$
Transthyretin	20701.4	12794.41	393	14568

to ensue. They suggested that three possible pathways can be performed. Two mechanisms will split the tetramer along the dimer interface to create two different intermediate dimer types (AB/CD or AC/BD) which can disassociate latter to monomers. The last mechanism can be performed in which one monomer is left from the complex to yield a trimer that will disassociate into three other monomers (see Fig. 7.9). Their results show that the mechanism 2, initial dimer scission into AB/CD dimers, is more consistent than the two others.

7.2.3.3 Sequential Disassembly Pathways

20 runs were performed with I-ML-T-RRT for the TTR model. The obtained results show that the mechanism 2, where the TTR is split along its horizontal dimer interface (AB/CD), is observed with the highest percentage 70% (i.e., 14/20 runs). Only 6 other pathways (30%) followed the mechanism 1 (i.e., vertical dimer interface split - AC/BD) and there is not any pathway which illustrates the third mechanism. The average differences of interaction energy between the assembled complex and the state where the dimers are split were computed over the two dimer-dimer dissociations : 80.1 kJ/mol for mechanism 1 and 77.7 kJ/mol for mechanism 2. This difference proved that the mechanism 2 is preferred in term of energy. Moreover, in term of structure, the stabilization of the AB/CD dimer interface mediated by hydrophobic

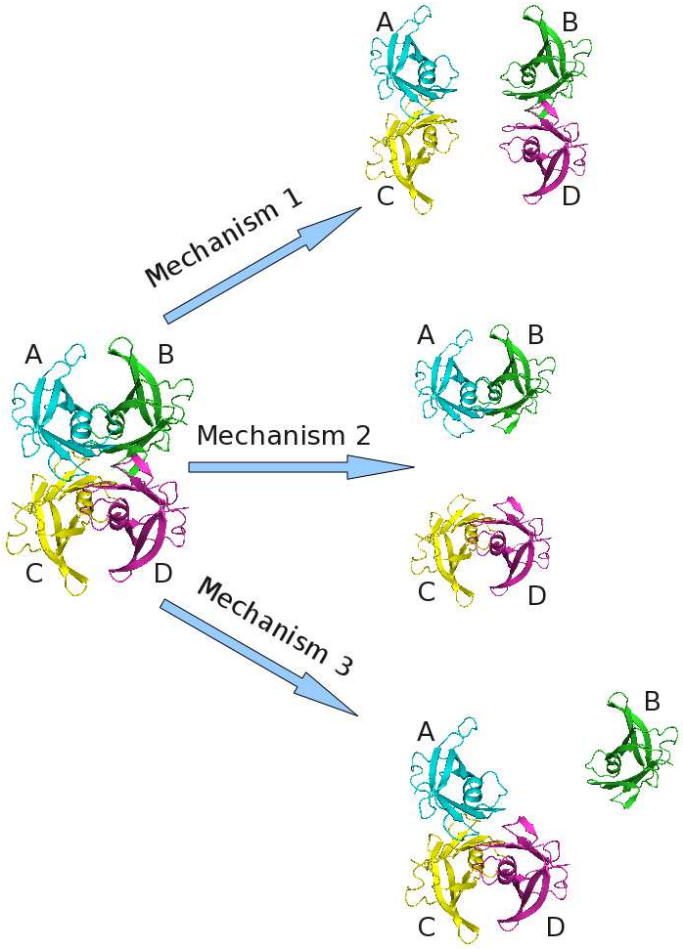


Figure 7.9: Three mechanisms of dissociation of the Transthyretin [Foss et al. 2005]: In mechanism 1 and 2, the tetramer split along vertical and horizontal dimer interfaces, respectively. The dimers then dissociate to monomers. In mechanism 3, a monomer is dissociated from the TTR to yield a trimer which then degrades to monomers.

interactions between AB and GH loops has little effect on quaternary structural stability [Foss et al. 2005; Hornberg et al. 2000]. On the contrary, the monomers of AB and CD dimers are held together by an intermolecular main-chain interaction involving the H-strands (residues Ser115-Thr123) which forms an eight-stranded β-sandwich between two monomers, and hydrogen bonding between two F-strands (residues His90-Val94) (see Fig. 7.7) [Hornberg et al. 2000]. These significant interactions inside a dimer do not facilitate the mechanisms 1 and 3. Furthermore, other works [Foss et al. 2005; Wiseman et al. 2005] also proved the stabilization of the quaternary structure of TTR can be maintained by ligand binding in on of the two binding-sites between A-C and B-D sub-units. Generally, our results are coherent with the experimented results [Foss et al. 2005]. Table 7.4 shows the computational performance of our method based on 20 experiments. The computational time for computing the sequencing pathways is still reasonable in the range of 6 hours.

7.3 Discussion

We have presented a mechanistic approach to molecular simulations which may lead to the development of efficient computational methods, able to provide relevant information on the protein complex dissociation. The proposed algorithm, I-ML-T-RRT, is a novel and fast method for simulating protein complex dissociation. Indeed, this algorithm generates the sequence of disassembly paths within a few hours of computing time on a single processor. Such a high computational performance is achieved thanks to the efficiency of the conformational search method combined with a simplified coarse-grained model to evaluate protein interaction forces. Preliminary results show the potential interest of the method. Nevertheless, the method requires improvements in order to deal with more complex meta-structures with higher accuracy. First, the conformational exploration method requires improvements to better deal with protein flexibility (i.e. loops and domain motions). Besides, a more accurate energy evaluation, taking into account the solvent and an entropic term, will probably be required in order to better model the effects of protein dissociation.

8 Conclusions

In this thesis we have investigated new computational methods for simulating large-amplitude molecular motions that represent important and challenging problems for structural biology, in particular for the understanding of the structure-function relationship in proteins. Our algorithms build on sampling-based planning methods recently developed in robotics. They exploit the efficiency of the Manhattan-like RRT method which was developed to circumvent the limitation of the basic RRT algorithm for dealing with disassembly problems involving complex articulated objects and high-dimensional search-spaces. The algorithmic contribution of the thesis is twofold.

First, in order to handle the high complexity of molecular models, we have presented promising extensions of the algorithm by introducing the notion of multi-level passiveness and of pushing-pulling motions. The first extension addresses disassembly planning problems for articulated objects involving parts with different mobilities. Different mobility coefficients are assigned to passive parts, which allows the planner to better explore the configuration-space. Therefore, the computational efficiency is significantly improved. The second extension is devised for solving problems in which passive articulated parts are "pushed" and also "pulled" by the mobile object, which is important in some classes of disassembly problems. The resulting algorithm was applied for studying the conformational changes induced by the ligand exit when taking into account the different levels of flexibility of different protein's parts such as side chains, loops or domains. Our results show the efficiency of our mechanistic approach to simulate large-amplitude diffusion paths within flexible protein models including more than one thousand degrees of freedom. Indeed, this algorithm generates long (20-30 $\AA$) diffusion paths within tens of minutes of computing time on a single processor, which is remarkably short compared to the time required by MD-based methods. Such a high computational performance is achieved thanks to the efficiency of the conformational search method that operates on geometric models of molecules. Geometrically feasible

paths are a reasonably good approximation that provides itself very useful information. Furthermore, as shown in our prior work, the approximate solution path can also be efficiently refined if needed with standard molecular modeling tools (e.g. energy minimization) in order to perform a more accurate energetic analysis.

Our second contribution is a novel method for simultaneously (dis)assembly sequencing and path planning. The proposed Iterative ML-RRT algorithm is able to solve general disassembly planning problems involving objects with arbitrary shapes and possibly requiring non-monotonic disassembly sequences, which may be a bottleneck for other methods. The method was applied to several CAD models, in particular to a complex 15-part engine model for which the disassembly sequence and motions were generated after only few minutes of computation. The method was also extended for investigating protein complex disassembly problems. The extension concerns the integration of energy computations to guide the conformational search. The new algorithm called I-ML-T-RRT was promisingly applied to disassemble several protein complexes. The sequential dissociation orders obtained within a few hours of computing time on a single processor are consistent with experimented results.

Future Research

The obtained results of this thesis open several directions for possible extensions. As future work, we intend to further improve the computational approach to better deal with full molecular flexibility during protein-ligand interactions while considering a simplified energy minimization during diffusion process. These energies could allow to consider important constraints that can not be treated by a geometric method when studying conformational changes.

We also expect to extend the method for its application to the modeling of protein-protein interactions or DNA-protein binding. Simulating the docking between flexible macromolecules yet remains a computational challenge and the efficiency of our conformational search techniques may help to further introduce molecular flexibility in docking methods.

Another interesting extension would be to consider optimal criteria, in particular, for minimizing the number of elementary motions generated by the Iterative-ML-RRT algorithm. Moreover, the current version which only treats one-by-one part disassembly can not properly afford other classes of problems (e.g. the Twisted Tetrahedron problem show in Chapter 6) where the grouped-part motion as a unit are highly required.

The latter extension may also be needed for the application to protein complex disassembly problems, in which groups of several subunits usually undergo ensemble motions. Also note that for this application, the preliminary results presented in the last chapter serve as a proof of concept but the method certainly has to be further improved to treat more accurately the physicochemical characteristics of protein complex dissociation.

References

ABAGYAN, R., TOTROV, M., AND KUZNETZOV, D. 1994. Icm – a new method for protein modeling and design: applications to docking and structure prediction from the distorted native conformation. *Journal Comput Chem, Vol 15, pp 488-506*. 15

ABRAMSON, J., RIISTAMA, S., LARSSON, G., JASAITIS, A., SVENSSON-EK, M., LAAKKONEN, L., PUUSTINEN, A., IWATA, S., AND WIKSTROM, M. 2000. The structure of the ubiquinol oxidase from *Escherichia coli* and its ubiquinone binding site. *Nature Struct. Biol. 7*, 910–917. 93

ABRAMSON, J., SMIRNOVA, I., KASHO, V., VERNER, G., KABACK, H. R., AND IWATA, S. 2003. Structure and mechanism of the lactose permease of Escherichia coli. *Science 301*, 610–615. 61

AGUINAGA, I., BORRO, D., AND MATEY, L. 2008. Parallel RRT-based path planning for selective disassembly planning. *Int. J. Adv. Manuf. Techno. 36*, 1221–1233. 70

AHMED, A., KAZEMI, S., AND GOHLKE, H. 2007. Protein flexibility and mobility in structure-based drug design. *Frontiers in Drug Design Discovery 3*, 455–476. 13, 14

AHUACTZIN, J. M. AND GUPTA, K. 1995. On manipulation planning. *Conference on Intelligent Robots and Systems - IROS*. 20

AHUJA, S., HORNAK, V., YAN, E. C., SYRETT, N., GONCALVES, J. A., HIRSHFELD, A., ZILIOX, M., SAKMAR, T. P., SHEVES, M., REEVES, P. J., SMITH, S. O., AND EILERS, M. 2009. Helix movement is coupled to displacement of the second extracellular loop in rhodopsin activation. *Nat. Struct. Mol. Biol. 16*, 168–175. 66

AKINC, M., BEKRIS, K. E., CHEN, B. Y., LADD, A. M., PLAKU, E., AND KAVRAKI, L. E. 2003. Probabilistic roadmaps of trees for parallel computation of multiple query roadmaps. *The Eleventh International Symposium*, 80–89. 20

ALAMI, R., LAUMOND, J., AND SIMÉON, T. 1994. Two manipulation planning algorithms. *The First Worshop on the Algorithmic Foundations of Robotics*. 20

ALBENNE, C., DE MONTALK, G. P., MONSAN, P., SKOV, L., MIRZA, O., GAJHEDE, M., AND REMAUD-SIMÉON, M. 2002. Site-directed mutagenesis of key amino acids in the active site of amylosucrase from neisseria polysaccharea. *Biologia, Vol 57(11)*, 119–128. 60

ALBENNE, C., SKOV, L., MIRZA, O., GAJHEDE, M., FELLER, G., D'AMICO, S., ANDRÉ, G., POTOCKI-VÉRONÈSE, G., MONSAN, P., AND REMAUD-SIMÉON, M. 2004. Molecular basis of the amylose-like polymer formation catalyzed by neisseria polysaccharea amylosucrase. *Journal Biol. Chem., Vol 279(1)*, 726–734. 60

ALBENNE, C., SKOV, L., TRAN, V., GAJHEDE, M., MONSAN, P., REMAUD-SIMÉON, M., AND ANDRÉ-LEROUX, G. 2007. Towards the molecular understanding of glycogen elongation by amylosucrase. *Proteins, Vol 66*, 118–126. 58

ALEXANDROV, V., LEHNERT, U., ECHOLS, N., MILBURN, D., ENGELMAN, D., AND GERSTEIN, M. 2005. Normal modes for predicting protein motions: a comprehensive database assessment and associated web tool. *Protein Sci. 14*, 633–643. 14

ALLEN, M. P. 2004. Introduction to monte carlo methods. In *Computational Soft Matter: From Synthetic Polymers to Proteins*, N. Attig, K. Binder, H. Grubmuller, and K. Kremer, Eds. Vol. 23. John von Neumann Institute for Computing, 1–28. 13

ALTEROVITZ, R., BRANICKY, M., AND GOLDBERG, K. 2008. Motion planning under uncertainty for image-guided medical needle steering. *Int. J. Robotics Research*, 1361–1374. 20

AMATO, N. M., B.BAYAZIT, O., DALE, L., JONES, C., AND VALLEJO, D. 1998. Obprm: An obstacle-based prm for 3d workspaces. *Proc of the WAFR, pp 155-168*. 21

AMATO, N. M. AND WU, Y. 1996. A randomized roadmap method for path and manipulation planning. *Proc. IEEE Int. Conf. Robot. Autom. (ICRA) 12*, 113–120. 20

ANDRUSIER, N., MASHIACH, E., NUSSINOV, R., AND WOLFSON, H. J. 2008. Principle of flexible protein-protein docking. *Proteins, Volume 73, Issue 2, Pages 271-289*. 17

APAYDIN, M. S., BRUTLAG, D. L., GUESTRIN, C., HSU, D., LATOMBE, J.-C., AND VARMA, C. 2003. Stochastic roadmap simulation: An efficient representation and algorithm for analyzing molecular motion. *Journal of Computational Biology, Vol 10(3/4), pp 257-281*. 26

APAYDIN, M. S., GUESTRIN, C., VARMA, C., BRUTLAG, D. L., AND LATOMBE, J.-C. 2002. Stochastic roadmap simulation for the study of ligand-protein interactions. *ECCB :18-26*. 25, 26

ARAI, E. AND IWATA, K. 1993. CAD system with product assembly/disassembly planning function. *Robotics and Computer-Integrated Manufacturing 10*, 41–48. 69

AVLANI, V. A., GREGORY, K. J., MORTON, C. J., PARKER, M. W., SEXTON, P. M., AND CHRISTOPOULOS, A. 2007. Critical role for the second extracellular loop in the binding of both orthosteric and allosteric g protein-coupled receptor ligands. *J. Biol. Chem. 282*, 25677–25686. 66

B-RAO, C., SUBRAMANIAN, J., AND SHARMA, S. D. 2009. Managing protein flexibility in docking and its applications. *Drug Discovery Today 14(7-8)*. 14, 15

BAKOWIES, D. AND THIEL, W. 1996. Hybrid models for combined quantum mechanical and molecular mechanical approaches. *J. Phys. Chem. 100*, 10580–10594. 12

BARRAQUAND, J. AND LATOMBE, J. C. 1991. Robot motion planning: A distributed representation approach. *Int. J. Robot. Res., vol. 10, no. 6*, 628–649. 22

BASTARD, K., PRÉVOST, C., AND ZACHARIAS, M. 2006. Accounting for loop flexibility during protein-protein docking. *Proteins 62*, 956–969. 17, 87

BESSIÈRE, P., AHUACTZIN, J. M., TALBI, E.-G., AND MAZER, E. 1993. The ariadnes clew algorithm: Global planning with local methods. *IEEE Int. Conf. on Intelligent Robots and Systems*. 22

BOHLIN, R. AND KAVRAKI, L. 2000. Path planning using lazy prm. *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 521–528. 20, 21

BOKOCH, M. P., ZOU, Y., RASMUSSEN, S. G. F., LIU, C. W., NYGAARD, R., ROSENBAUM, D. M., FUNG, J. J., CHOI, H.-J., THIAN, F. S., KOBILKA, T. S., PUGLISI, J. D., WEIS, W. I., PARDO, L., PROSSER, R. S., MUELLER, L., AND KOBILKA, B. K. 2010. Ligand-specific regulation of the extracellular surface of a G-protein-coupled receptor. *Nature 463*, 108–112. 63, 66

BOOR, V., OVERMARS, M. H., AND VAN DER STAPPEN, A. F. 1999. The gaussian sampling strategy for probabilistic roadmap planners. *Proc. IEEE Int. Conf. Robot. Autom. ICRA*, 1018–1023. 20, 21

BOURJAULT, A. Contribution à une approche méthodologique de l'assemblage automatisé : Elaboration automatique des séquences opératoires. Thèse d'Etat, Université de Franche-Comté, 1984. 29

BROOKS, B., BROOKS, C., MACKERELL, A. J., NILSSON, L., PETRELLA, R., ROUX, B., WON, Y., ARCHONTIS, G., BARTELS, C., BORESCH, S., CAFLISCH, A., CAVES, L., CUI, Q., DINNER, A., FEIG, M., FISCHER, S., GAO, J., HODOSCEK, M., IM, W., KUCZERA, K., LAZARIDIS, T., MA, J., OVCHINNIKOV, V., PACI, E., PASTOR, R., POST, C., PU, J., SCHAEFER, M., TIDOR, B., RM, V., HL, W., X, W., W, Y., DM, Y., AND KARPLUS, M. 2009. Charmm: the biomolecular simulation program. *J. Comput. Chem. 30(10)*, 1545–1614. 13

BROOKS, B. AND KARPLUS, M. 1985. Normal modes for specific motions of macromolecules: application to the hinge-bending mode of lysozyme. *Proc Natl Acad Sci USA 82*, 4995–4999. 14

BROOKS, B. R., BRUCCOLERI, R. E., OLAFSON, B. D., STATES, D. J., SWAMINATHAN, S., AND KARPLUS, M. 1983. Charmm: A program for macromolecular energy, minimization, and dynamics calculations. *J. Comput. Chem. 4(2)*, 187–217. 13

BURENDAHL, S., DANCIULESCU, C., AND NILSSON, L. 2009. Ligand unbinding from the estrogen receptor: A computational study of pathways and ligand specificity. *Proteins, Vol 77, pp 842-856*. 16

BURKERT, U. AND ALLINGER, N. 1982. *Molecular Mechanics*. American Chemical Society Publication. 12

BURNS, B. AND BROCK, O. 2005. Sampling-based motion planning using predictive models. *ICRA*, 3120–3125. 20, 21

BURNS, B. AND BROCK, O. 2007. Single-query motion planning with utility-guided random trees. *Proc. IEEE Int. Conf. Robot. Automat.*, 3307–3312. 32

CAMACHO, C. J. AND VAJDA, S. 2001. Protein docking along smooth association pathways. *Proceeding of the National Academy of Sciences of the United States of America*. 25

CARLSON, H. A. 2002. Protein flexibility is an important component of structure-based drug discovery. *Curr. Pharm. Des. 8*, 1571–1578. 49

CARPIO, C. A. D., QIANG, P., JCHIISHI, E., TSUBOI, H., KOYAMA, M., HATAKEYAMA, N., ENDOU, A., TAKABA, H., KUBO, M., AND MIYAMOTO, A. 2006. Robotic path planning and protein complex modeling considering low frequency intra-molecular loop and domain motions. *Genome Informatics 17(2): 270-278*. 24

CARPIO, C. A. D., SHAIKH, A. R., JCHIISHI, E., KOYAMA, M., KUBO, M., NISHIJIMA, K., AND MIYAMOTO, A. 2005. A graph theoretical approach for analysis of protein flexibility change at protein complex formation. *Genome Informatics 16(2): 148-160*. 24

CAZALS, F. AND DREYFUS, T. 2009. Assessing the stability of protein complexes within large assemblies. *ISMB Satellite Meeting on Structural Bioinformatics and Computational Biophysics*. 18

CAZALS, F., PROUST, F., BAHADUR, R. P., AND JANIN, J. 2006. Revisiting the voronoi description of protein-protein interfaces. *Protein Science 15,* 9, 2082–2092. 17

CHAMPION, E., ANDRÉ, I., BOUTET, J., DESCROIX, K., MOREL, S., MONSAN, P., MULARD, L., AND REMAUD-SIMÉON, M. 2009. Towards the molecular understanding of glycogen elongation by amylosucrase. *Journal Am. Chem. Soc., Vol 131(21)*, 7379–7389. 58, 60

CHANDLER, D. 2005. Interfaces and the driving force of hydrophobic assembly. *Nature 437*, 640–647. 7

CHANG, H. AND LI, T.-Y. 1995. Assembly maintainability study with motion planning. *Proc. IEEE Int. Conf. Robot. Automat.*, 1012–1019. 20, 30, 70

CHEBARO, Y., DONG, X., LAGHAEI, R., DERREUMAUX, P., AND MOUSSEAU, N. 2009. Replica exchange molecular dynamics simulations of coarse-grained proteins in implicit solvent. *J Phys. Chem. B. 113(1)*, 267–274. 13

CHENG, P. AND LAVALLE, S. M. 2001. Reducing metric sensitivity in randomized trajectory design. *Proc. IEEE/RSJ Int. Conf. Intel. Rob. Sys.*, 43–48. 23, 33

CHEREZOV, V., ROSENBAUM, D. M., HANSON, M. A., RASMUSSEN, S. G., THIAN, F. S., KOBILKA, T. S., CHOI, H. J., KUHN, P., WEIS, W. I., KOBILKA, B. K., AND STEVENS, R. C. 2007. High-resolution crystal structure of an engineered human beta2-adrenergic G protein-coupled receptor. *Science 318*, 1258–1265. 63, 64

CHERFILS, J. AND JANIN, J. 1993. Protein docking algorithms: simulating molecular recognition. *Current Opinion in Structural Biology 3*, 265–269. 17

CHIANG, T., APAYDIN, M., BRUTLAG, D., HSU, D., AND LATOMBE, J. 2007. Using stochastic roadmap simulation to predict experimental quantities in protein folding kinetics: Folding rates and phi-values. *J. Comp. Biol. 14(5)*, 578–593. 5

CHOSET, H., LYNCH, K. M., HUTCHISON, S., KANTOR, G., BURGARD, W., KAVRAKI, L. E., AND THRUN, S. 2005. *Principles of Robot Motion*. The MIT Press. 19

CONNOLLY, M. L. 1986. Shape complementarity at the hemoglobin $\alpha_1\beta_1$ subunit interface. *Biopolymers 25,* 7, 1229–1247. 17

CORTÉS, J. 2004. Motion planning algorithms for general closed-chain mechanisms. Ph.D. thesis, Institut National Polytechnique de Toulouse - ENSEEIHT. 20

CORTÉS, J., JAILLET, L., AND SIMÉON, T. 2007. Molecular disassembly planning with rrt-like algorithms. *IEEE Int. Conf. on Robotics and Automation*. 23, 26, 33

CORTÉS, J., JAILLET, L., AND SIMÉON, T. 2009. Disassembly path planning for complex articulated objects. *IEEE Transactions on Robotics 24(2)*. 23

CORTÉS, J., SIMÉON, T., DE ANGULO, V. R., GUIEYSSE, D., REMAUD-SIMÉON, M., AND TRAN, V. 2005. A path-planning approach for computing large-amplitude motions of flexible molecules. *Bioinformatics Journal, Vol 21 sup.1*. 24, 26, 68

CORTÉS, J., SIMÉON, T., REMAUD-SIMÉON, M., AND TRAN, V. 2004. Geometric algorithms for the conformational analysis of long protein loops. *Journal of Computational Chemistry, Vol 25, n7*. 56

CRAIG, J. 1989. *Introduction to Robotics: Mechanics and Control.* Addison-Wesley Longman Publishing. 51

CUI, Q. AND BAHAR, I. 2006. *Normal mode analysis: theory and applications to biological and chemical systems.* Chapman Hall, CRC Mathematical Computational Biology. 13

DALEY, D. 2008. The assembly of membrane proteins into complexes. *Curr Opi Struc Biol, Vol 18, pp 420-424*. 93, 94

DELANO, W. L. 2002. The PyMOL molecular graphics system. http://www.pymol.org. 57, 91

DEPRISTO, M. A., DE BAKKER, P. I. W., LOVELL, S. C., AND BLUNDELL, T. L. 2003. Ab initio construction of polypeptide fragments: Efficient generation of accurate, representative ensembles. *Proteins 51*, 41–55. 56

DERREUMAUX, P. AND MOUSSEAU, N. 2007. Coarse-grained protein molecular dynamics simulations. *J. Chem. Phys. 126*, 25101–25106. 12

DEVILLERS, J. 1975. *Adaptation in Natural and Artificial Systems.* University of Michigan Press. 13

DEVILLERS, J. 1996. *Genetic Algorithms in Molecular Modeling (Principles of QSAR and Drug Design).* Academic Press. 13

DUGGAN, C., MUTLU, D., AND WORSNOPP, T. 2000. Overview of the robotic claw project. http://lims.mech.northwestern.edu/∼design/mechatronics/2000/Team24/encoder_assembly.html. 70

DUHOVNY, D., NUSSINOV, R., AND WOLFSON, H. J. 2002. Efficient unbound docking of rigid molecules. In *Proceedings of the Second International Workshop on Algorithms in Bioinformatics*. Lecture Notes In Computer Science, vol. 2452. 185–200. 17

ENOSH, A., FLEISHMAN, S., BEN-TAL, N., AND HALPERIN, D. 2006. Prediction and simulation of motion in pairs of transmembrane alpha-helices. *Bioinformatics 23*, 212–218. 25

EWING, T. AND KUNTZ, I. 1997. Critical evaluation of search algorithms for automated molecular docking and database screening. *Journal Comput Chem, Vol 18, pp 1175-1189*. 15

FERRÉ, E. AND LAUMOND, J. 2004. An iterative diffusion algorithm for part disassembly. *Proc. IEEE Int. Conf. Robot. Automat.*, 3194–3150. 23, 30, 70

FETROW, J. 1995. Omega loops: nonregular secondary structures significant in protein function and stability. *FASEB Journal,Vol 9, pp 708-717.*. 8

FINKELSTEIN, A. 1997. Can protein unfolding simulate protein folding? *Protein Engineering 10(8)*, 843–845. 9

FOGEL, E. AND HALPERIN, D. 2008. Polyhedral assembly partitioning with infinite translations or the importance of being exact. *Proc.Workshop on the Algorithmic Foundations of Robotics*. In press. 70, 75, 78, 79, 84

FOSS, T., KELLY, M., WISEMAN, R., WILSON, I., AND KELLY, J. 2005. Kinetic stabilization of the native state by protein engineering: implications for inhibition of transthyretin amyloidogenesis. *J. Mol. Biol. 347*, 841–854. 97

FOSS, T., WISEMAN, R., AND KELLY, J. 2005. The pathways by which the tetrameric protein transthyretin disassociates. *Biochemistry, Vol 44, pp 15525-15533*. 95, 97

FRAUENFELDER, H., SLIGAR, S. G., AND WOLYNES, P. G. 1991. The energy landscapes and motions of proteins. *Science 254*, 1598–1603. 49

FREDDOLINO, P. L., ARKHIPOV, A., SHIH, A. Y., YIN, Y., CHEN, Z., AND SCHULTEN, K. 2009. Application of residue-based and shape-based coarse graining to biomolecular simulations. In *Multiscale Simulation Methods in Molecular Sciences*, J. Grotendorst, N. Attig, S. Blugel, and D. Marx, Eds. Vol. 42. Institute for Advanced Simulation, 445–466. 12

FRENKEL, D. 2004. Introduction to monte carlo methods. In *Computational Soft Matter: From Synthetic Polymers to Proteins*, N. Attig, K. Binder, H. Grubmuller, and K. Kremer, Eds. Vol. 23. John von Neumann Institute for Computing, 29–60. 13

GABB, H. A., JACKSON, R. M., AND STERNBERG, M. J. E. 1997. Modelling protein docking using shape complementarity, electrostatics and biochemical information. *Journal of Molecular Biology 272,* 1, 106–120. 17

GENEST, D., GARNIER, N., ARRAULT, A., MAROT, C., MORIN-ALLORY, L., AND GENEST, M. 2008. Ligand escape pathways from the ligand-binding domain of pparγ receptor as probed by molecular dynamics simulations. *Journal Biophy, Vol 95, pp 4193-4204*. 15

GERAERTS, R. AND OVERMARS, M. H. 2002. A comparative study of probabilistic roadmap planners. *Proc. Workshop on the Algorithmic Foundations of Robotics*, 43–57. 22

GERAERTS, R. AND OVERMARS, M. H. 2005. On improving the clearance for robots in high-dimensional configuration spaces. *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems*, 4074–4079. 22

GERSTEIN, M., LESK, A., AND CHOTHIA, C. 1994. Structural mechanisms for domain movements in proteins. *Biochemistry, Vol 33, pp 6739-6749*. 8

GOHLKE, H., KUHN, L. A., AND CASE, D. A. 2004. Change in protein flexibility upon complex formation: Analysis of ras-raf using molecular dynamics and a molecular framework approach. *PROTEINS: Structure, Function, and Bioinformatics 56*, 322–337. 14

GRAY, J. J., MOUGHAN, S. E., WANG, C., SCHUELER-FURMAN, O., KUHLMAN, B., ROHL, C. A., AND BAKER, D. 2003. Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations. *Journal of Molecular Biology 331,* 1, 281–299. 17

GROMIHA, M. AND SELVARAJ, S. 2002. Importance of long-range interactions for determining the folding rate and transition state structures of two-state proteins. *Genome Informatics*, 348–349. 5

HALPERIN, D., LATOMBE, J., AND WILSON, R. 2000. A general framework for assembly planning: The motion space approach. *Algorithmica 26*, 577–601. 30, 69, 70, 84

HALPERIN, I., MA, B., WOLFSON, H., AND NUSSINOV, R. 2002. Principles of docking: An overview of search algorithms and a guide to scoring functions. *Proteins 47(4)*, 409–443. 17

HAN, L. 2004. Hybrid probabilistic roadmap - monte carlo motion planning for closed chain systems with spherical joints. *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, 920–926. 20

HEGLER, J. A., LTZER, J., SHEHU, A., CLEMENTI, C., AND WOLYNES, P. G. 2009. Restriction versus guidance in protein structure prediction. *Proc. Nat. Acad. Sci. USA 106(36)*, 15302–15307. 14

HENZLER, A. AND RAREY, M. 2010. In pursuit of fully flexible protien-ligand docking: Modeling the bilateral mechanism of binding. *Mof Inf, 29, pp 164-173*. 14

HOLLEMAN, C. AND KAVRAKI, L. E. 2000. A framework for using the workspace medial axis in prm planners. *ICRA 2*, 1408–1413. 21

HOLLEMAN, C., KAVRAKI, L. E., AND WARREN, J. 1998. Planning paths for a flexible surface patch. *Proc. IEEE Int. Conf. on Robotics and Automation*, 21–26. 21

HOMEM DE MELLO, L. S. AND LEE, S. 1991. *Computer-Aided Mechanical Assembly Planning*. Kluwer Academic Publishers, Boston. 29

HORNAK, V., OKUR, A., RIZZO, R., AND SIMMERLING, C. 2006. Hiv-1 protease flaps spontaneously open and reclose in molecular dynamics simulations. *Proc. Nat. Acad. Sci. USA. 103*, 915–920. 13

HORNAK, V. AND SIMMERLING, C. 2007. Targeting structural flexibility in hiv-1 protease inhibitor binding. *Drug Discovery Today 12*, 132–138. 13

HORNBERG, A., ENEQVIST, T., OLOFSSON, A., LUNDGREN, E., AND SAUER-ERIKSSON, A. E. 2000. A comparative analysis of 23 structures of the amyloidogenic protein transthyretin. *J. Mol. Biol. 320*, 649–669. 95, 97

HSU, D., JIANG, T., REIF, J., AND SUN, Z. 2003. The bridge test for sampling narrow passages with probabilistic roadmap planners. *Proc. IEEE Int. Conf. Robot. Autom. ICRA*, 4420–4426. 20, 21

HSU, D., KINDEL, R., LATOMBE, J.-C., AND ROCK, S. 2000. Randomized kinodynamic motion planning with moving obstacles. *Proc. IEEE Int. Conf. on Robotics and Automation*. 22

HSU, D., SÁNCHEZ-ANTE, G., CHENG, H.-L., AND LATOMBE, J.-C. 2006. Multi-level free-space dilation for sampling narrow passages in prm planning. *ICRA*, 1255–1260. 21

IZRAILEV, S., STEPANIANTS, S., ISRALEWITZ, B., KOSZTIN, D., LU, H., MOLNAR, F., WRIGGERS, W., AND SCHULTEN, K. 1998. Steered molecular dynamics. In *Computational Molecular Dynamics: Challenges, Methods, Ideas. Vol. 4 of Lecture Notes in Computational Science and Engineering.*, P. Deuflhard, J. Hermans, B. Leimkuhler, A. Mark, S. Reich, and R. Skeel, Eds. Springer-Verlag, Berlin, 39–65. 16, 49

JACOBS, D. J., RADER, A., KUHN, L. A., AND THORPE1, M. 2001. Protein flexibility predictions using graph theory. *PROTEINS: Structure, Function, and Genetics 44*, 150–165. 14

JAILLET, L., CORTÉS, J., AND SIMÉON, T. 2008. Transition-based RRT for path planning in continuous cost spaces. *Proc. IEEE/RSJ Int. Conf. Intel. Rob. Sys.*, 22–26. 68, 84

JAILLET, L., CORTÉS, J., AND SIMÉON, T. 2010. Sampling-based path planning on configuration-space costmaps. *IEEE Transactions on Robotics*. 23, 85, 89, 90

JAILLET, L. AND SIMÉON, T. 2004. A prm-based motion planner for dynamically. *Proc. IEEE IROS*, 1606–1611. 20

JAILLET, L. AND SIMÉON, T. 2006. Path deformation roadmaps. *7th Int. Workshop on Algorithmic Foundations of Robotics*. 20

JAILLET, L. AND SIMÉON, T. 2009. Path deformation roadmaps : compact graphs with useful cycles for motion planning. *The International Journal of Robotics Research, Vol 27(11-12)*. 22

JAILLET, L., YERSHOVA, A., LAVALLE, S. M., AND SIMÉON, T. 2005. Adaptive tuning of the sampling domain for dynamic-domain rrts. *Proceedings IEEE International Conference on Intelligent Robots and Systems*. 23

JENSEN, M., YIN, Y., TAJKHORSHID, E., AND SCHULTEN, K. 2007. Sugar transport across lactose permease probed by steered molecular dynamics. *Biophys J. 93*, 92–102. 62, 63

JONES, G., WILLETT, P., GLEN, R., LEACH, A., AND TAYLOR, R. 1997. Development and validation of a genetic algorithm for flexible docking. *Journal Mol Biol, Vol 267, pp 727-748*. 13, 15

KABACK, H. R., SAHIN-TÓTH, M., AND WEINGLASS, A. B. 2001. The kamikaze approach to membrane transport. *Nature Rev. Moll. Cell. Biol. 2(8)*, 610–620. 49, 61

KARPLUS, M. AND MCCAMMON, J. 2002. Molecular dynamics simulations of biomolecules. *Nature Struct. Biol. 9*, 646–652. 12, 13

KATCHALSKI-KATZIR, E., SHARIV, I., EISENSTEIN, M., FRIESEM, A. A., AFLALO, C., AND VAKSER, I. A. 1992. Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. In *Proceedings of the National Academy of Sciences of the United States of America*. Vol. 89. 2195–2199. 17

KATONA, G., CARPENTIER, P., , V. N., AMARA, P., ADAM, V., OHANA, J., TSANOV, N., AND BOURGEOIS, D. 2007. Raman-assisted crystallography reveals end-on peroxide intermediates in a nonheme iron enzyme. *Science 316*, 449–453. 49

KAVRAKI, L. E., KOLOUNTZAKIS, M., AND LATOMBE, J.-C. 1996. Analysis of probabilistic roadmaps for path planning. *ICRA*, 3020–3026. 21

KAVRAKI, L. E. AND LATOMBE, J.-C. 1998. Probabilistic roadmaps for robot path planning. *Practical Motion Planning in Robotics: Current Approaches and Future Directions, K. Gupta and A. del Pobil (eds), John Wiley, pp. 33-53*. 29

KAVRAKI, L. E., SVESTKA, P., VESTKA, L. E. K. P., CLAUDE LATOMBE, J., AND OVERMARS, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation 12*, 566–580. 19, 20

KIRILLOVA, S., CORTÉS, J., STEFANIU, A., AND SIMÉON, T. 2008. An NMA-guided path planning approach for computing large-amplitude conformational changes in proteins. *Proteins 70*, 131–143. 14, 25

KONGAR, E. AND GUPTA, S. 2001. Genetic algorithm for disassembly process planning. *Proc. SPIE Int. Conf. on Environmentally Conscious Manufactoring II*, 54–62. 70

KOZAKOV, D., BRENKE, R., COMEAU, S. R., AND VAJDA, S. 2006. Piper: An fft-based protein docking program with pairwise potentials. *Proteins 65,* 2, 392–406. 17

KRAMER, B., METZ, G., RAREY, M., AND LANGAUER, T. 1999. Ligand docking and screening with flexx. *Med Chem Res, Vol 9, pp 463-478*. 15

KUFFNER, J. J. 2004. Effective sampling and distance metrics for 3D rigid body path planning. *Proc. IEEE Int. Conf. Robot. Automat.*, 3993–3998. 32

KULAJTA, C., THUMFART, J., HAID, S., DALDAL, F., AND KOCH, H. 2006. Multi-step assembly pathway of the cbb_3-type cytochrome c oxidase complex. *Journal Mol Biol, Vol 355, pp 989-1004*. 92

KUMAR, S., MA, B., TSAI, C., WOLFSON, H., AND NUSSINOV, R. 1999. Folding funnels and conformation transitions via hinge-bending motions. *Cell Biochem Biophsys, 31(2), pp 141-164.*. 7

KURNIAWATI, H. AND HSU, D. 2006. Workspace-based connectivity oracle - an adaptive sampling strategy for prm planning. *Proceedings of the International Workshop on the Algorithmic Foundations of Robotics (WAFR*. 21

LAMBERT, A. 2000. Optimum disassembly sequence generation. *Proc. SPIE Conf. on Environmentally Consious Manufactoring*, 56–67. 70

LAMBERT, A. 2003. Disassembly sequencing: A survey. *Int. Journal of Production Research 41*, 3721–3759. 69, 70

LAMBERT, A. AND GUPTA, S. 2005. *Disassembly modeling for Assembly, Maintenance, Reuse and Recycling*. CRC Press, Florida. 69

LAMIRAUX, F. AND KAVRAKI, L. E. 2001. Planning paths for elastic objects under manipulation constraints. *International Journal of Robotics Research 20*, 188–208. 21

LATOMBE, J. 1991. *Robot Motion Planning*. Kluwer Academic Publishers, Boston. 19, 29, 31

LAVALLE, S. 1998. Rapidly-exploring random trees: A new tool for path planning. *TR 98-11, Computer Science Dept., Iowa State University.*. 19, 20, 22, 23, 29, 30, 32

LAVALLE, S. 2006. *Planning Algorithms*. Cambridge University Press, New York. 19, 29, 31

LAVALLE, S. M. AND KUFFNER, J. J. 2001a. Randomized kinodynamic planning. *I. J. Robotic Res. 20,* 5, 378–400. 23

LAVALLE, S. M. AND KUFFNER, J. J. 2001b. Rapidly-exploring random trees : Progress and prospects. In *Algorithmic and Computational Robotics: New Directions*, B. Donald, K. Lynch, and D. Rus, Eds. A.K. Peters, Boston, 293–308. 23, 32, 33, 70

LAVERY, R. AND SACQUIN-MORA, S. 2007. Protein mechanics: a route from structure to function. *J. Biosci. 32(5)*, 891–898. 87

LEACH, A. R. 2001a. Dorset Press. 13

LEACH, A. R. 2001b. *Roadmap Methods for Protein Folding*. Dorset Press. 9

LENSINK, M. F. AND MÉNDEZ, R. 2008. Recognition-induced conformational changes in protein-protein docking. *Curr. Pharm. Biotechnol. 9*, 77–86. 49

LI, L., CHEN, R., AND WENG, Z. 2003. Rdock: refinement of rigid-body protein docking predictions. *Proteins 53(3)*, 693–707. 17

LIEN, J.-M., THOMAS, S. L., AND AMATO, N. M. 2003. A general framework for sampling on the medial axis of the free space. *In Proc. IEEE Int. Conf. Robot. Autom. (ICRA*, 4439–4444. 21

LORENZEN, S. AND ZHANG, Y. 2007. Monte carlo refinement of rigid-body protein docking structures with backbone displacement and side-chain optimization. *Protein Sci 16(12)*, 2716–2725. 17

LORIOT, S., CAZALS, F., LEVITT, M., AND BERNAUER., J. 2009. A geometric knowledge-based coarse-grained scoring potential for structure prediction evaluation. *JOBIM*. 12

LOZANO-PÉREZ, T. 1983. Spatial planning : A configuration space approach. *IEEE Trans Comp, Vol 32(2), pp 108-120*. 31

LOZANO-PÉREZ, T. AND WILSON, R. H. 1993. Assembly sequencing for arbitrary motions. *Proc. IEEE Int. Conf. Robot. Automat.*, 527–532. 30

LUCIA K. DALE, N. M. A. 2001. Probabilistic roadmaps - putting it all together. *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1940–1947. 20, 22

LUDEMANN, S. K., LOUNNAS, V., AND WADE, R. C. 2000. How do substrates enter and products exit the buried active site of cytochrome P450cam? 1. random expulsion molecular dynamics investigation of ligand access channels and mechanisms. *J. Mol. Biol. 303*, 797–811. 16, 49

MA, J. 2005. Usefulness and limitations of normal mode analysis in modeling dynamics of biomolecular complexes. *Structure 13(3)*, 373–380. 13

MARRINK, S., RISSELADA, H., YEFIMOV, S., TIELEMAN, P., AND DE VRIEST, A. H. 2007. he martini force field: Coarse grained model for biomolecular simulations. *J. Phys. Chem. 111*, 7812–7824. 12

MILLER, M., KEARSLEY, S., UNDERWOOD, D., AND SHERIDAN, R. 1994. Flog: a system to select 'quasi-flexible' ligands complementary to a receptor of known three-dimensional structure. *J Comput Aided Mol Des. 8(2)*, 153–174. 15

MIYASHITA, O., ONUCHIC, J. N., AND WOLYNES, P. G. 2003. Nonlinear elasticity, proteinquakes, and the energy landscapes of functional transitions in proteins. *Proc Natl Acad Sci USA 100*, 12570–12575. 14

MOLL, M., SCHWARZ, D., AND KAVRAKI, L. E. 2007. *Roadmap Methods for Protein Folding*. Humana Press. 5

MOREIRA, I. S., FERNANDES, P. A., AND RAMOS, M. J. 2010. Protein-protein docking dealing with the unknown. *Journal of Computational Chemistry 31,* 2, 317–342. 17

MORRIS, G., GOODSELL, D., HALLIDAY, R., HUEY, R., HART, W., BELEW, R., AND OLSON, A. 1998. Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. *Journal Comput Chem, Vol 19, pp 1639-1662*. 13, 15

MOUAWAD, L. AND PERAHIA, D. 1993. Diagonalization in a mixed basis: a method to compute low-frequency normal modes for large macromolecules. *Biopolymers 33*, 599–611. 14

NATARAJAN, B. K. 1988. On planning assemblies. *Proc. ACM Symp. Comput. Geom.*, 299–308. 79

NESATYYA, V. J. AND LASKINB, J. 2002. Dissociation of non-covalent protein complexes by triple quadrupole tandem mass spectrometry: comparison of monte carlo simulation and experiment. *Int. J. Mass Spec. 221*, 245–262. 18

NIELSEN, C. L. AND KAVRAKI, L. E. 2000. A two level fuzzy prm for manipulation planning. *In Proceedings of the International Conference on Intelligent Robots and Systems*, 1716–1722. 20

NIJTMANS, L., TAANMAN, J., MUIJSERS, A. O., SPEIJER, D., AND BOGERT, C. V. D. 1998. Assembly of cytochrome-c oxidase in cultured human cells. *Eur. J. Biochem. 254*, 389–394. 92

NOOREN, I. AND THORNTON, J. 2003. Diversity of protein-protein interactions. *The EMBO Journal, Vol.22, No.14, pp.3486-3492*. 9

NOREL, R., LIN, S. L., WOLFSON, H. J., AND NUSSINOV, R. 1994. Shape complementarity at protein-protein interfaces. *Biopolymers 34,* 7, 933–940. 17

NOREL, R., LIN, S. L., WOLFSON, H. J., AND NUSSINOV, R. 1995. Molecular surface complementarity at protein-protein interfaces: the critical role played by surface normals at well placed, sparse, points in docking. *Journal of Molecular Biology 252,* 2, 263–273. 17

NORTHRUP, S. H. AND ERICKSON, H. P. 1992. Kinetics of proteinprotein association explained by brownian dynamics computer simulation. *Proc Natl Acad Sci USA 89(8)*, 3338–3342. 18

OH, J. AND KAPLAN, S. 2002. Oxygen adaptation: The role of the ccoq subunit of the cbb3 cytochrome c oxidase of rhodobacter sphaeroides 2.4.1. *BioChem. Soc. Trans. 277(18)*, 16220–16228. 91

OTADUY, M. A. AND LIN, M. C. 2003. Clods: Dual hierarchies for multiresolution collision detection. *In Proc. of Eurographics Symposium on Geometry Processing*, 94–101. 21

PARK, S. H., REYES, J. A., GILBERT, D. R., KIM, J. W., AND KIM, S. 2009. Prediction of protein-protein interaction types using association rule based classification. *Bioinformatics 10(36)*. 9

PETERS, A., KULAJTA, C., PAWLIK, G., DALDAL, F., AND KOCH, H. 2008. Stability of the cbb3-type cytochrome oxidase requires specific ccoq-ccop interactions. *Journal Bacteri. 190(16)*, 5576–5586. 91, 92

PETREK, M., KOSINOVA, P., KOCA, J., AND OTYEPKA, M. 2007. Mole: A voronoi diagram-based explorer of molecular channels, pores, and tunnels. *Structure 15(11)*, 1357–1363. 16

PETREK, M., OTYEPKA, M., BANAS, P., KOSINOVA, P., KOCA, J., AND DAMBORSKY, J. 2006. Caver: a new tool to explore routes from protein clefts, pockets and cavities. *Bioinformatics 7)*, 316–325. 16

PIERCE, B. AND WENG, Z. 2008. A combination of rescoring and refinement significantly improves protein docking performance. *Proteins 72(1)*, 270–279. 17

PITCHER, R., BRITTAIN, T., AND WATMOUGH, N. 2002. Cytochrome cbb3 oxidase and bacterial microaerobic metabolism. *BioChem. Soc. Trans. 30*, 653–658. 91

POGORELOV, T., AUTENRIETH, R., ROBERTS, E., AND LYTHEY-SCHULTEN, Z. 2007. Cytochrome c_2 exit strategy : Disassociation studies and evolutionary implications. *Journal Phys Chem, Vol 111, pp 618-634*. 18

RENAUD, M. 2000. A simplified inverse kinematic model calculation method for all 6R type manipulators. In *Current Advances in Mechanical Design and Production VII*, M. F. Hassan and S. M. Megahed, Eds. Pergamon, New York, 57–66. 56

RITCHIE, D. 2008. Recent progress and future directions in protein-protein docking. *Curr. Prot. Pep. Sci. 9(1)*, 1–15. 17

RODRIGUEZ, S., THOMAS, S., PEARCE, R., AND AMATO, N. M. 2006. Resampl: A region-sensitive adaptive motion planner. *Proc. Int. Wkshp. on Alg. Found. of Rob. (WAFR)*. 21

RUIZ DE ANGULO, V., CORTÉS, J., AND SIMÉON, T. 2005. BioCD: An efficient algorithm for self-collision and distance computation between highly articulated molecular models. In *Robotics: Science and Systems*, S. T. snd G. Sukhatme, S. Schaal, and O. Brock, Eds. MIT Press, Cambridge, 6–11. 56

SAHA, M., LATOMBE, J., CHANG, Y.-C., AND PRINZ, F. 2005. Finding narrow passages with probabilistic roadmaps: The small-step retraction method. *Autonomous Robots 19(3)*, 301–319. 20, 21

SANCHEZ, G. AND LATOMBE, J.-C. 2002. On delaying collision checking in prm planning – application to multi-robot coordination. *INTERNATIONAL JOURNAL OF ROBOTICS RESEARCH 21*, 5–26. 21

SARCABAL, P., REMAUD-SIMÉON, M., WILLEMOT, R., DE MONTALK, G. P., SVENSSON, B., AND MONSAN, P. 2000. Identification of key amino acid residues in neisseria polysaccharea amylosucrase. *FEBS Letters, Vol 474*, 33–37. 57

SCHANDA, P., FORGE, V., AND BRUTSCHER, B. 2007. Protein folding and unfolding studied at atomic resolution by fast two-dimensional nmr spectroscopy. *PNAS 104*, 11257–11262. 49

SCHLICK, T. 2002. *Molecular modeling and simulation - an interdisciplinary guide.* New York: Springer. 13

SEDLAK, E. AND ROBINSON, N. C. 2009. Sequential disassociation of subunits from bovine heart cytochrome c oxidase by urea. *Biochemistry 48*, 8143–8150. 92

SEKHAVAT, S., SVESTKA, P., LAUMOND, J., AND OVERMARS, M. 1998. Multilevel path planning for nonholonomic robots using semi-holonomic subsystems. *Int. J. Robot. Res., Vol 17(8)*, 840–857. 20, 35

SELKOE, D. 2003. Folding proteins in fatal ways. *Nature 426*, 900–904. 9

SHARMA, V., PUUSTINEN, A., WIKSTROM, M., AND LAAKKONEN, L. 2006. Sequence analysis of the cbb3 oxidases and an atomic model for the rhodobacter sphaeroides enzyme. *Biochemistry 45*, 5754–5765. 91

SHARMA, V., WIKSTROM, M., AND KAILA, V. 2010. Redox-coupled proton transfer in the active site of cytochrome cbb3. *Biochim. Biophys. Acta - Bioenergetics 4C*, 46462–46472. 91

SHEHU, A., CLEMENTI, C., AND KAVRAKI, L. E. 2006. Modeling protein conformational ensembles: From missing loops to equilibrium fluctuations. *Proteins: Structure, Function and Bioinformatics 65*, 164–179. 24

SHERWOOD, P. 2000. Hybrid quantum mechanics/molecular mechanics approaches. In *Modern Methods and Algorithms of Quantum Chemistry*, J. Grotendorst, Ed. Vol. 3. John von Neumann Institute for Computing, 285–305. 12

SIMÉON, T., CORTÉS, J., LAUMOND, J., AND SAHBANI, A. 2004. Manipulation planning with probabilistic roadmaps. *The International Journal of Robotics Research 23(7-8)*. 20

SIMÉON, T., LAUMOND, J.-P., AND LAMIRAUX, F. 2001. Move3D: A generic platform for path planning. *Proc. IEEE Int. Symp. Assembly and Task Planning*, 25–30. 35, 74

SIMÉON, T., LAUMOND, J.-P., AND NISSOUX, C. 2000. Visibility-based probabilistic roadmaps for motion planning. *Advanced Robotics 14,* 6, 477–493. 20, 21, 22

SIMONS, K. T., KOOPERBERG, C., HUANG, E., AND BAKER, D. 1997. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. *J. Mol. Biol. 268(1)*, 209–225. 14

SINGH, A. P., LATOMBE, J.-C., AND BRUTLAG, D. L. 1999. A motion planning approach to flexible ligand binding. *American Association for Artificial Intelligence*. 25

SKOV, L., MIRZA, O., HENRIKSEN, A., DE MONTALK, G. P., REMAUD-SIMÉON, M., SARCABAL, P., WILLEMOT, R., MONSAN, P., AND GAJHEDE, M. 2001. Amylosucrase, a glucan-synthesizing enzyme from the α-amylase family. *Journal Biol. Chem., Vol 276(27)*, 25273–25278. 57

SKOV, L., MIRZA, O., SPROGOE, D., DAR, I., REMAUD-SIMÉON, M., ALBENNE, C., MONSAN, P., AND GAJHEDE, M. 2002. Oligosaccharide and sucrose complexes of amylosucrase. structural implications for the polymerase activity. *Journal Biol. Chem., Vol 277(49)*, 47741–47747. 57, 58

SMIRNOVA, I., KASHO, V., CHOE, J.-Y., ALTENBACH, C., HUBBELL, W. L., AND KABACK, H. R. 2007. Sugar binding induces an outward facing conformation of lacy. *Proc. Natl. Acad. Sci. USA 104*, 16504–16509. 61, 62

SMITH, G. R. AND STERNBERG, M. J. E. 2002. Prediction of protein–protein interactions by docking methods. *Current Opinion in Structural Biology 12,* 1, 28–35. 17

SNOEYINK, J. AND STOLFI, J. 1994. Objects that can not be taken apart with two hands. *Discrete Comput. Geom. 12*, 367–384. 79, 80

SONG, G. AND AMATO, N. M. 2001. Using motion planning to study protein folding pathways. *Proc. Int. Conf. Comput. Molecular Biology*, 287–296. 26

SONG, G. AND AMATO, N. M. 2004. A motion planning approach to folding: From paper craft to protein folding. *IEEE Transactions on Robotics and Automation 20(1)*, 60–71. 26

SONG, G., MILLER, S., AND AMATO, N. M. 2001. Customizing prm roadmaps at query time. *In Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1500–1505. 22

SOUSA, S. F., FERNANDES, P., AND RAMOS, M. 2006. Protein-ligand docking: Current status and future challenges. *Proteins: Structure, Function, and Bioinformatics 65*, 15–26. 14, 15

STENBERG, F., HEIJNE, G., AND DALEY, D. 2007. Assembly fo the cytochrome bo_3 complex. *Journal Mol Biol, Vol 371, pp 765-773*. 85, 93, 94

SUNDARAM, S., REMMLER, I., AND AMATO, N. 2001. Disassembly sequencing using a motion planning approach. *Proc. IEEE Int. Conf. Robot. Automat.*, 1475–1480. 70, 75, 76, 78

SVESTKA, P. 1996. On probabilistic completeness and expected complexity for probabilistic path planning. Utrecht, the Netherlands: Utrecht University: Information and Computing Sciences. 20

TAANMAN, J. AND WILLIAMS, S. 2001. Assembly of cytochrome c oxidase: what can we learn from patients with cytochrome c oxidase deficiency? *Biochem. Soc. Trans. 29(4)*, 446–451. 92

TAMA, F. AND SANEJOUAND, Y. 2001. Conformational change of proteins arising from normal mode analysis. *Protein Eng. 14*, 1–6. 14

TEAGUE, S. 2003. Implications of protein flexibility for drug discovery. *Nat Rev Drug Discov, Vol 2(7), pp 527-541.*. 8

TERFLOTH, L. AND GASTEIGER, J. 2001. Neural networks and genetic algorithms in drug design. *Drug discovery today 6(15)*, 102–108. 13

THIEL, W. 2009. Qm/mm methodology: Fundamentals, scope, and limitations. In *Multiscale Simulation Methods in Molecular Sciences*, J. Grotendorst, N. Attig, S. Blugel, and D. Marx, Eds. Vol. 42. Institute for Advanced Simulation, 203–214. 12

TOTROV, M. AND ABAGYAN, R. 2008. Flexible ligand docking to multiple receptor conformations: a practical alternative. *Curr Opin Struct Biol, 18(2), pp 178-184*. 15

TOZZINI, V. 2005. Coarse-grained models for proteins. *Curr. Opi. Struct. Biol. 15*, 144–150. 12

URMSON, C. AND SIMMONS, R. 2003. Approaches for heuristically biasing RRT growth. *Proc. IEEE/RSJ Int. Conf. Intel. Rob. Sys.*, 1178–1183. 23, 33

VAUQUELIN, G. AND VON MENTZER, B. 2007. *G Protein-Coupled Receptors: Molecular Pharmacology from Academic Concept to Pharmaceutical Research*. John Wiley & sons Ltd, Chichester. 63

VIETH, M., HIRST, J., DOMINY, B., DAIGLER, H., AND BROOKS, C. I. 1998. Assessing search strategies for flexible docking. *Journal Comput Chem, Vol 19, pp 1623-1631*. 15

WANG, T. AND DUAN, Y. 2009. Ligand entry and exit pathways in the β_2-adrenergic receptor. *J. Mol. Biol. 392*, 1102–1115. 64, 65, 66

WARREN, G., PEISHOFF, C., AND HEAD, M. 2008. Docking algorithms and scoring functions: State-of-the-art and current limitations. In *Computational and Structural Approaches to Drug Discovery: Ligand–Protein Interactions*, R. M. Stroud and J. Finer-Moore, Eds. The Royal Society of Chemistry, 137–154. 14

WEINER, P. K. AND KOLLMAN, P. A. 1981. Amber: Assisted model building with energy refinement. a general program for modeling molecules and their interactions. *J. Comput. Chem. 2(3)*, 287–303. 13

WELLS, S., MENOR, S., HESPENHEIDE, B., AND THORPE, M. F. 2005. Constrained geometric simulation of diffusive motion in proteins. *Phys. Biol. 2*, 127–136. 52

WILMARTH, S. A., AMATO, N. M., AND STILLER, P. F. 1999. Maprm: A probabilistic roadmap planner with sampling on the medial axis of the free space. *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1024–1031. 20, 21

WILSON, R., KAVRAKI, L., LOZANO-PEREZ, T., AND LATOMBE, J. 1995. Two-handed assembly sequencing. *Int. Journal of Robot. Research 14-4*, 335–350. 70, 84

WILSON, R. H. 1992. On geometric assembly planning. Ph.D. thesis, Stanford University. 30

WISEMAN, R., JOHNSON, S. M., KELKER, M., FOSS, T., WILSON, I., AND KELLY, J. 2005. Kinetic stabilization of an oligomeric protein by a single ligand binding event. *J. Am. Chem. Soc. 127*, 5540–5551. 97

YAFFE, E., FISHELOVITCH, D., WOLFSON, H., HALPERIN, D., AND NUSSINOV, R. 2008. Molaxis: a server for identification of channels in macromolecules. *Nucleic Acids Res 36)*, 210–215. 16

YAO, P., DHANIK, A., MARZ, N., PROPPER, R., KOU, C., LIU, G., VAN DEN BEDEM, H., LATOMBE, J., LANDSBERGZ, I. H., AND ALTMAN, R. 2008. Efficient algorithms to explore conformation spaces of flexible protein loops. *IEEE/ACM Trans. on Comput. Biol. and Bioin., Vol 5(4)*, 534–545. 24

YERSHOVA, A., JAILLET, L., SIMÉON, T., AND LAVALLE, S. M. 2005. Dynamic-domain rrts: Efficient exploration by controlling the sampling domain. *Proceedings IEEE International Conference on Robotics and Automation*. 23, 32

YUN, M. R., LAVERY, R., MOUSSEAU, N., ZAKRZEWSKA, K., AND DERREUMAUX, P. 2006. Artist: An activated method in internal coordinate space for sampling protein energy landscapes. *PROTEINS-STRUCTURE FUNCTION AND BIOINFORMATICS 63(4)*, 967–975. 14

ZACHARIAS, M. 2003. Protein-protein docking with a reduced model accounting for side chain flexibility. *Proteins Sci, Vol 12*, 1271–1282. 12, 17, 87, 89

ZACHARIAS, M. 2005. Attract: Protein-protein docking in capri using a reduced protein model. *Proteins, Vol 60(2)*, 252–256. 17, 87

ZAVODSZKY, M. I. AND KUHN, L. A. 2004. Side-chain flexibility in protein-ligand binding: The minimal rotation hypothesis. *Protein Science, Vol 14, Issue 4, pp 1104-1114*. 7

ZHAO, F., PENG, J., AND XU, J. 2010. Fragment-free approach to protein folding using conditional neural fields. *BIOINFORMATICS 26*, 310–317. 14

ZHOU, H. X. 1997. Enhancement of proteinprotein association rate by interaction potential: Accuracy of prediction based on local boltzmann factor. *Biophys 73(5)*, 2441–2445. 18

ZHOU, Y., GUAN, L., FREITES, J. A., AND KABACK, H. R. 2008. Opening and closing of the periplasmic gate in lactose permease. *Proc. Natl. Acad. Sci. USA 105*, 3774–3778. 62

ZUFFEREY, R., PREISIG, O., HENNECKE, H., AND THONY-MEYER, L. 1996. Assembly and function of the cytochrome cbb3 oxidase subunits in bradyrhizobium japonicum. *Journal Biol. Chem. 271(15)*, 9114–9119. 92