1-1-2014

Grasping and Assembling with Modular Robots

Jungwon Seo
University of Pennsylvania, meclory@gmail.com

Grasping and Assembling with Modular Robots

Abstract
A wide variety of problems, from manufacturing to disaster response and space exploration, can benefit from robotic systems that can firmly grasp objects or assemble various structures, particularly in difficult, dangerous environments. In this thesis, we study the two problems, robotic grasping and assembly, with a modular robotic approach that can facilitate the problems with versatility and robustness.

First, this thesis develops a theoretical framework for grasping objects with customized effectors that have curved contact surfaces, with applications to modular robots. We present a collection of grasps and cages that can effectively restrain the mobility of a wide range of objects including polyhedra. Each of the grasps or cages is formed by at most three effectors. A stable grasp is obtained by simple motion planning and control. Based on the theory, we create a robotic system comprised of a modular manipulator equipped with customized end-effectors and a software suite for planning and control of the manipulator.

Second, this thesis presents efficient assembly planning algorithms for constructing planar target structures collectively with a collection of homogeneous mobile modular robots. The algorithms are provably correct and address arbitrary target structures that may include internal holes. The resultant assembly plan supports parallel assembly and guarantees easy accessibility in the sense that a robot does not have to pass through a narrow gap while approaching its target position. Finally, we extend the algorithms to address various symmetric patterns formed by a collection of congruent rectangles on the plane.

The basic ideas in this thesis have broad applications to manufacturing (restraint), humanitarian missions (forming airfields on the high seas), and service robotics (grasping and manipulation).

Degree Type
Dissertation

Degree Name
Doctor of Philosophy (PhD)

Graduate Group
Mechanical Engineering & Applied Mechanics

First Advisor
Vijay Kumar

Second Advisor
Mark Yim

Keywords
assembly planning, modular robotics, robotic grasping

GRASPING AND ASSEMBLING WITH MODULAR ROBOTS

Jungwon Seo

A DISSERTATION

in

Mechanical Engineering and Applied Mechanics

Presented to the Faculties of the University of Pennsylvania in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

2014

Vijay Kumar, Supervisor of Dissertation
Professor of Mechanical Engineering and Applied Mechanics

Mark Yim, Co-Supervisor of Dissertation
Professor of Mechanical Engineering and Applied Mechanics

Prashant Purohit, Graduate Group Chairperson
Associate Professor of Mechanical Engineering and Applied Mechanics

Dissertation Committee:
Camillo J. Taylor, Associate Professor of Computer and Information Science
Nicolas H. Hudson, Google Robotics / Boston Dynamics

GRASPING AND ASSEMBLING WITH MODULAR ROBOTS

To my family.

Acknowledgments

First and foremost, I would like to thank my advisors, Dr. Vijay Kumar and Dr. Mark Yim, for their guidance, inspiration, encouragement, passion, and patience throughout my graduate studies. Without their support, it would have been impossible for me to get through academic and personal challenges. I would also like to thank Dr. Camillo Taylor and Dr. Nicolas Hudson for serving on my thesis committee and providing valuable comments and insights. Special thanks to my friends and the staff of MEAM, MRSL, ModLab, and GRASP. I am enormously grateful to the love and support of my parents, especially in loving memory of my father. My deepest thanks go to my wife, Sura, and our lovely daughter, Gio Leonor.

ABSTRACT

GRASPING AND ASSEMBLING WITH MODULAR ROBOTS

Jungwon Seo

Vijay Kumar

Mark Yim

A wide variety of problems, from manufacturing to disaster response and space exploration, can benefit from robotic systems that can firmly grasp objects or assemble various structures, particularly in difficult, dangerous environments. In this thesis, we study the two problems, robotic grasping and assembly, with a modular robotic approach that can facilitate the problems with versatility and robustness.

First, this thesis develops a theoretical framework for grasping objects with customized effectors that have curved contact surfaces, with applications to modular robots. We present a collection of grasps and cages that can effectively restrain the mobility of a wide range of objects including polyhedra. Each of the grasps or cages is formed by at most three effectors. A stable grasp is obtained by simple motion planning and control. Based on the theory, we create a robotic system comprised of a modular manipulator equipped with customized end-effectors and a software suite for planning and control of the manipulator.

Second, this thesis presents efficient assembly planning algorithms for constructing planar target structures collectively with a collection of homogeneous mobile modular robots. The algorithms are provably correct and address arbitrary target

structures that may include internal holes. The resultant assembly plan supports parallel assembly and guarantees easy accessibility in the sense that a robot does not have to pass through a narrow gap while approaching its target position. Finally, we extend the algorithms to address various symmetric patterns formed by a collection of congruent rectangles on the plane.

The basic ideas in this thesis have broad applications to manufacturing (restraint), humanitarian missions (forming airfields on the high seas), and service robotics (grasping and manipulation).

Contents

List of Tables

List of Figures

Part I

Introduction and Background

Chapter 1

Introduction

The early 1960s saw Unimate, the first industrial robot created by George Devol, working on an automobile assembly line. Since this installation of Unimate, a wide variety of robot technologies have been developed to perform various manipulation tasks that can help or completely replace humans in environments ranging from a manufacturing plant to outer space or the bottom of the sea. For example, a robot played an important role in capturing the Boston Marathon bombing suspect in 2013 by helping law enforcement authorities remove the tarpaulin of the boat. The DARPA Robotics Challenge[1] reflects growing interest in developing robots that can physically assist humans with challenging manipulation tasks, particularly for responding disasters.

The work presented in this thesis contributes to two of the most fundamental

[1] `http://www.theroboticschallenge.org/`

problems in robotic manipulation: robotic grasping and assembly. The capability to grasp or assemble objects can provide a sufficient functional basis for a wide variety of tasks which robots can contribute to, for example, manufacturing, disaster response, space exploration, assisted living, and medical operation.

The two problems are closely related to each other. According to Mason (2001), grasping an object is a kind of assembly if we consider assembly as a fundamental process employed in manipulation tasks; assembly is an application task that builds on a wide range of subtasks including grasping. In the literature, the problems are indeed sharing many common issues; for example, contact analysis has been important in both problems.

Our approach to robotic grasping and assembly takes account of modular robot systems. For grasping, we consider a modular manipulator whose arms are reconfigurable by attaching or detaching modular links and end-effectors as needed. For assembly, we consider modular units with identical geometry that can collectively be assembled into various target structures. The theories and algorithms we present here are suitable for the modular frameworks. In fact, the tasks of grasping and assembly can benefit from such modular systems. For example, the modular manipulator can easily adapt to the sizes and shapes of various objects; the parallelism of the multiple modular building blocks makes the system robust to failures. In addition, mass production of standardized modules can make individual modules less expensive and robotic systems built from such modules more affordable.

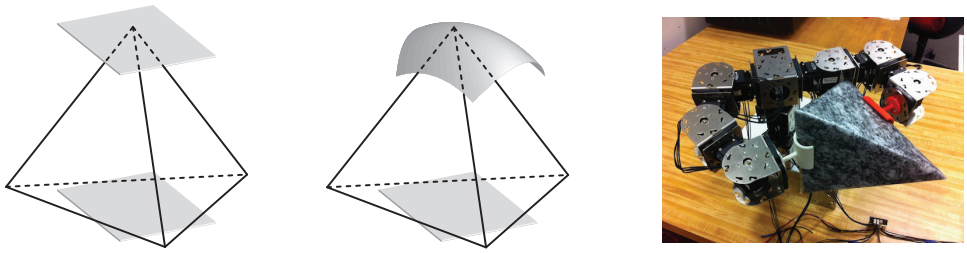
Section 1.1 formally defines our research problems. Section 1.2 enumerates the contributions of the thesis. Section 1.3 describes our research vision based on the thesis. Section 1.4 gives an outline of the thesis.

1.1 Problem Statement

This thesis addresses two central problems in robotic manipulation.

First, we are concerned with developing a theoretical framework for robotic grasping using effectors (or "fingers") with curved contact surfaces; we also present one application of the idea to a scenario of grasping objects with a modular robot system. Effectors with appropriate curvature properties can be effective for restraining the mobility of an object. For example, consider the two grasps shown in Figures 1.1a and 1.1b, each of which has one point and one planar contact; the grasp of Figure 1.1b is more restrictive in that the object cannot actually escape from the effectors due to the concavity of the effector contacting the vertex. Figure 1.1c illustrates a grasp by a modular manipulator equipped with two end-effectors with curved contact surfaces.

Most practical solutions in robotic grasping involve specially-designed hardware and control algorithms that are tailored only to a couple of objects to be handled or grasped. A robot system that can grasp a wide variety of object shapes without many different types of effectors or complex multi-fingered hands can save time and cost in a wide range of scenarios from handling material in a warehouse to



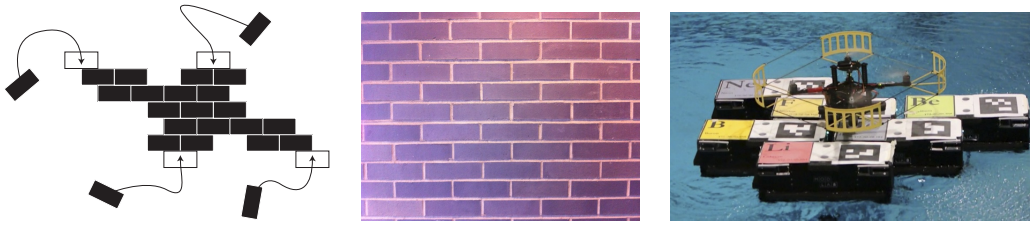
(a) (b) (c)

Figure 1.1: The tetrahedron is grasped by (a) the two planar effectors, (b) the planar and concave effectors, and (c) the modular manipulator.

clearing rubble in an unstructured environment. In addition, it is imperative to develop planning algorithms that can guarantee the stability of the process of grasp acquisition and robustness to sensing/positioning errors.

Second, we address the development of planning algorithms for assembling arbitrary planar target structures with congruent, rectangular building blocks, which can be applied to constructing floating structures on water with modular, robotic boats. Figure 1.2 illustrates the scenario we are concerned with: in Figure 1.2a, the rectangular mobile units are being assembled into the growing structure, which locally looks like the common brick wall (Figure 1.2b), in a parallel manner. Figure 1.2c shows an example target structure assembled with physical robots.

We seek to create efficient algorithms guaranteeing complete, correct assembly and supporting parallel execution, mimicking the process seen in human workers collaboratively constructing a brick wall. Prior works on assembly planning (see Section 2.2) discussed automated geometric reasoning based only on local information. However, such a local-scale analysis is not sufficient to guarantee accessibility; for



(a) (b) (c)

Figure 1.2: (a) Robots, depicted as solid, black rectangles, are being assembled into the planar structure, which locally looks like the common brick wall shown in (b). (c) A landing platform autonomously assembled with robotic boats.

example, parts may have to pass through narrow corridors on the way to their target positions, which may necessitate difficult maneuvers. Decentralized approaches to robotic self-assembly (see Section 2.3) presented planning algorithms that scale well; however, it can be hard to guarantee completeness with such algorithms.

1.2 Thesis Contributions

The main contributions of this thesis fall into two distinct areas.

First, we present a novel theory of three types of immobilizing grasps and cages that can effectively restrain the mobility of any object modeled as a polyhedron. The grasps and cages are formed by at most three effectors with appropriate geometry, which can simply be a planar, cylindrical, or spherical surface. We apply the theory to implement hardware and software for stable object grasping with a modular robot system, which can adapt to the sizes and shapes of a wide variety of objects. Our work is based on a conservative assumption that two bodies in contact can only push each other (unilateral contact) and there is neither friction nor compliance in

contact. The conservativeness can be seen clearly if we consider how the octopus firmly grasps a prey by virtue of the frictional, soft, bilateral contacts made by the tentacles. The conservativeness will allow us to apply our approach to a wide range of scenarios without such capable hardware like the tentacles of the octopus.

Second, we present two novel planning algorithms that can be applied to constructing planar structures with congruent, rectangular mobile robots, which collectively form the brick wall pattern. The algorithms can address arbitrary target structures[2]; moreover, target structures without internal holes can be assembled in a parallel manner. It takes $O(m)$ time to run the algorithms where m is the number of the modular units constituting the structure. Following the resultant assembly plan guarantees easy accessibility: each robot is guaranteed a path with a finite clearance between itself and the growing structure. We also show that the algorithms can be extended to assemble structures of other symmetric patterns that can be formed with congruent rectangles.

1.3 Vision

In order to illustrate the applications of the ideas in this thesis, we present two vignettes.

[2]See Section 9.2.

Vignette 1: Modular robot system for disaster response A team of autonomous boats sails for an island that has just been struck by an earthquake and the resultant tsunami. After anchoring off the coast of the island, some of the boats unload a swarm of small modular robots. While the boats form an emergency landing strip on the water, the robots assemble themselves into a team of spider-like robots, which can move around the island cluttered with rubble. The multi-limbed robots remove obstacles, search for survivors, and build temporary ground shelter with bricks scattered over the ground.

Vignette 2: Modular robot system for space exploration Another group of the same modular robots deployed from a lunar lander forms a three-armed torso that is to be mounted on a rover. The three-armed rover performs sample acquisition and coring with two of the arms immobilizing a rock sample and the remaining one operating a tool.

The vignettes may sound far-fetched, but this thesis can be the first step toward the vision; in Chapter 14, we propose possible 5-, 10-, and 15-year milestones that can be reached by extending the ideas described in this thesis.

In addition, we also envision industrial applications. For example, *material handling* is defined as the movement, storage, control and protection of materials, goods and products throughout the process of manufacturing, distribution, consumption

and disposal[3]. According to the *U.S. Roadmap for Material Handling & Logistics*[4] published in January, 2014, all that movement and handling accounts for 8.5 percent of gross domestic product (at least $1.33 trillion) in the United States and the total continues to grow at roughly 4 percent annually. Robotic grasping and assembly, the topic of the thesis, provide a sufficient functional basis for automating many tasks involved with material handling.

1.4 Organization of This Work

The thesis is organized as follows. In the following chapter, we review relevant literature in the areas of robotic grasping, robotic assembly, and modular robotics.

Part II presents our work on robotic grasping with modular robots. This part builds on our previous work presented in Seo et al. (2012), Seo and Kumar (2012), Seo et al. (2013b). Chapter 3 introduces concepts and terminology necessary to develop our theory and algorithms. Chapter 4 discusses three types of immobilizing grasps using curved effectors that can be applied to a wide range of objects including polyhedra. Chapter 5 discusses three types of cages derived from the immobilizing grasps and explains how to establish sufficient conditions for caging. Chapter 6 presents an algorithm for synthesizing the immobilizing grasps and cages. Chapter 7 extends our theory by adding more types of grasps and cages. Chapter 8 discusses

[3] `http://www.mhi.org`

[4] `http://www.mhlroadmap.org`

the implementation of our approach on a modular robot system, with experiments.

Part III is concerned with developing assembly planning algorithms for constructing modular structures. This part builds on our previous work presented in Seo et al. (2013a), O'Hara et al. (2014). Chapter 9 describes our approach to designing the algorithms. Chapter 10 presents our first algorithm that supports parallel assembly and discusses its correctness. Chapter 11 presents our second algorithm, which can address target structures with internal holes, and discusses its correctness. Chapter 12 discusses the implementation of the algorithms, with experiments. Chapter 13 addresses how to extend the algorithms to more general patterns.

We conclude in Chapter 14, with suggestions for future work.

Chapter 2

Literature Review

This chapter outlines literature on robotic grasping, robotic assembly, and modular robotics that is relevant to this thesis.

2.1 Robotic Grasping

Robotic grasping has been an active research area over the past few decades. We here divide the prior work into two categories: one focusing on immobilizing objects by making contacts (prehensile approach) and the other based on caging (non-prehensile approach). For each approach, we introduce theoretical aspects and examples of robotic systems. See Bicchi and Kumar (2000) for a more general survey.

2.1.1 Prehensile Approach

We begin with discussing the closure properties of grasps. A grasp is defined as *force closed* (Nguyen 1988) if and only if it can resist any external wrench. If a grasp is force closed with frictionless contacts, it is said to be *form closed* (Lakshminarayana 1978) or *immobilized* (Rimon and Burdick 1998a). Trinkle (1992) proposed a quantitative test formulated as a linear program for detecting form closure. Markenscoff et al. (1990) showed that it is possible to immobilize a three-dimensional object with seven frictionless point contacts, using first-order theories based on contact normals. Algorithms for synthesizing force or form closed grasps were presented by Ponce et al. (1997), Borst et al. (1999), Van der Stappen et al. (1999). Rimon and Burdick (1998a,b) developed a second-order mobility theory for rigid bodies in contact where the curvature properties at the contacts are taken into account. Czyzowicz et al. (1991) showed that $n + 1$ frictionless point contacts suffice to immobilize a general n-dimensional polytope by using the effects of relative curvature; thus, for a three-dimensional object, four frictionless point contacts suffice for immobilization.

It has been discovered that the stability of a grasp depends on the geometry of the grasp, contact forces, and material properties. Mason and Salisbury (1985) established a framework for testing the stability of a grasp: a grasp is stable if its stiffness matrix is positive definite. Cutkosky and Kao (1989) showed that grasp stability is a function of local geometry, fingertip models, and the compliance of

the fingers. Nguyen (1989) proved that all force closed grasps can be made stable. Howard and Kumar (1996) established a framework for analyzing grasp stability that takes compliance, contact forces, and the local curvature properties of the bodies in contact into account. It was shown that immobilization implies dynamic stability with elastic contacts (Rimon and Burdick 1998a,b).

In practice, having a large number of contacts can be beneficial to grasp stability; however, synthesizing such a grasp can be computationally intractable. Pollard (2004) presented an efficient algorithm for synthesizing many-contact grasps based on user-provided examples. Similar approaches can also be seen in the literature on whole-body grasping (Hsiao and Lozano-Perez 2006) and enveloping grasping (Trinkle et al. 1988). Napier (1956) showed that there are two approaches to achieving stability in human grasping: power grip and precision grip. Whole-body grasps and enveloping grasps are in the same vein of the human power grip, where an object is held by a large number of contacts between the flexed fingers and the palm.

Since Hanafusa et al. (1977) presented one of the earliest examples of robotic hands, various robotic hands have been developed. Our work is relevant with the approach to building simple yet versatile end-effectors that can be seen in Jacobsen et al. (1986), Ulrich et al. (1988), Dollar and Howe (2010), Kragten et al. (2011), Mason et al. (2012). Another relevant approach can also be seen in the literature on modular fixturing (Brost and Goldberg 1996, Ponce 1996). Recently, there has been growing interest in developing robotic systems that can grasp/manipulate ob-

jects with some autonomy. Saxena et al. (2008) presented a vision-based approach to robotic grasping and demonstrated real systems that can grasp previously unknown objects using two-dimensional images; similar approaches can be seen in Morales et al. (2002), Bowers and Lumia (2003). Hudson et al. (2012) developed an autonomy system that can perform dexterous, high-precision tasks such as key insertion.

2.1.2 Non-Prehensile Approach

In contrast to the prehensile approach, the literature on caging investigates how to arrange "obstacles" (that is, robotic fingers or effectors) around an object so as to bound its mobility without necessarily making contact. Caging allows us to sidestep some difficult issues such as modeling contacts or optimizing contact forces although the caged object may have some freedom to move. Rimon and Blake (1999) formulated a technique for computing cages of two-fingered hands; Davidson and Blake (1998a) extended the result to three-fingered hands. Vahedi and van der Stappen (2008a,b,c, 2009) provided an algorithm for synthesizing cages of two and three fingers around polygonal objects and formalized the concepts of squeezing and stretching cages for polygonal objects, which were generalized by Rodriguez and Mason (2009) to address objects in Euclidean spaces of arbitrary dimension. Allen et al. (2012) presented a simpler algorithm for computing two-fingered cages for polygons based on contact space analysis. Wan et al. (2012)

proposed a solution to synthesizing three-finger cages on the plane where two of the fingers are fixed. Other interesting approaches include Zamfirescu (1995), Maehara (2011), Fruchard (2012) where they investigated how to cage objects with just a single circle. Rodriguez et al. (2012) discussed the relationship between caging and grasping: they investigated when a cage can be a useful waypoint to an equilibrium grasp.

Recently, there have been efforts to take advantage of caging to robustify robotic tasks. Davidson and Blake (1998b) presented error-tolerant, vision-based planar grasping by closing fingers that form a cage. Gopalakrishnan and Goldberg (2002) presented a simple gripper with two vertical, parallel cylindrical jaws that can stably grasp objects by forming a cage on concavities. Diankov et al. (2008) proposed a motion planning algorithm for performing manipulation tasks with cages, relaxing task constraints. Yokoi et al. (2009) presented an approach to transporting objects using cages formed by not only robots but also the environment such as walls. Cappelleri et al. (2011a,b) employed cages formed by micro-manipulators for transporting and manipulating micro-scale polygonal parts. Dogar and Srinivasa (2011) showed that simple manipulation such as quasi-static pushing can help robots stably cage and grasp objects even in clutter. There is a body of literature featuring decentralized approaches to caging; we refer the reader to Section 2.3.3.

2.2 Assembly Planning

Robotic assembly is a broad topic that is involved with a wide variety of issues in manipulating objects, which include grasping, caging, fixturing, pushing, and part orienting. See Mason (2001) for a general introduction. We here focus on the literature on assembly planning. According to Halperin et al. (2000), assembly planning is defined as the problem of finding and sequencing the motions that put the initially separated parts of an assembly together to form the assembled product. Lozano-Perez (1976) is one of the earliest works that focus on specific issues in planning mechanical assembly. The problem is generally approached by considering how to establish a *dis*assembly plan from a final product.

2.2.1 Assembly Sequencing

Assembly sequencing is a variant of assembly planning that received early attention. In assembly sequencing, the parts of an assembly are often assumed to be free-flying, sidestepping issues such as how to physically perform assembly operations and focusing on the geometric constraints imposed by the product itself. However, Assembly sequencing is a hard problem; Natarajan (1988), Kavraki and Kolountzakis (1995) discuss the PSPACE-hardness or NP-completeness of instances of assembly sequencing. De Fazio and Whitney (1987) presented a method for generating all valid assembly sequences based on a user input on the geometric relationships of the parts. Homem de Mello and Sanderson (1990) presented the

hypergraph representation of assembly plans that combines all feasible assembly sequences for a given product; the representation enables the selection of the best assembly plan and parallel execution of assembly operations. Ko and Lee (1987), Arkin et al. (1989), Wilson and Rit (1990) also presented similar approaches. Wilson and Latombe (1994) presented the notion of a non-directional blocking graph, representing the geometric interferences among the parts in an assembly, which allows assembly sequences to be computed in polynomial time.

2.2.2 Beyond Traditional Assembly Sequencing

The traditional approach to assembly sequencing has been generalized in many directions. Latombe et al. (1997), Thomas et al. (2003), Ostrovsky-Berman and Joskowicz (2006) investigated assembly planning for toleranced parts. Halperin et al. (2000) presented a general framework for assembly planning that can address additional constraints such as toleranced parts, stability, and tool use. Romney (1997) presented a method to concurrently generate an assembly sequence and design a fixture to hold intermediate subassemblies. Mosemann et al. (1998), Rakshit and Akella (2014) presented assembly/disassembly sequencing that takes part stability into account in the presence of external forces such as gravity and friction.

Assembly planning can also be understood as a variant of robot motion planning where the goal is to assemble robotic parts into one coherent structure (LaValle 2006). Sundaram et al. (2001) presented an approach for disassembly sequencing

based on sampling-based robot motion planning. Similar approaches can also be seen in Ferre and Laumond (2004), Le et al. (2009).

2.3 Modular Approach to Robotic Tasks

According to Yim et al. (2009), modular (self-reconfigurable) robots are robots composed of a large number of repeated modules that can rearrange their connectedness to form a large variety of structures. Modular robots promise to be versatile, robust, and cost-efficient (Yim et al. 2009), but the advantages may compromise performance as observed by Yim et al. (2007b). We here review the applications of modular robotics to self-assembly, locomotion, and manipulation tasks.

2.3.1 Self-Assembly

There is extensive literature on modular self-assembly; here the focus is on generating a wide range of structures with possibly congruent modules arranged in a two- or three-dimensional grid structure. Murata et al. (1994) presented a mechanical system composed of repeated modular units and a decentralized software system for self-assembly. Kotay et al. (1998) proposed Molecule, a robotic module that can self-reconfigure, along with an efficient motion planning algorithm. Yim et al. (2001) defined a class of metamorphic robotic system capable of approximating any three-dimensional shapes and presented distributed control algorithms for reconfiguration. Butler et al. (2004) presented dynamic reconfiguration algorithms for a

general model of self-reconfigurable robots, with applications to real systems. There is a body of literature addressing cube style modular systems with a wide variety of modes of locomotion between voxels (pixels); examples include Hosokawa et al. (1998), Kurokawa et al. (1998), Rus and Vona (2001), Vassilvitskii et al. (2002), Gilpin et al. (2008), Romanishin et al. (2013).

Difficulties in fabricating small modules with onboard sensing, computation, and actuation often result in outsourcing some functions. White et al. (2004) proposed a self-reconfigurable robotic system where simple modules, without moving parts, exploit Brownian motion in their environment for locomotion. Bishop et al. (2005) introduced a self-organizing modular robotic system where each module floats passively on an air table and docks to others upon random collisions. Werfel and Nagpal (2008) presented a decentralized algorithm for constructing three-dimensional target structures with a bipartite system comprising passive, cubic blocks and mobile robots that move the blocks. White et al. (2009) presented a chain of tetrahedron-shaped modules that can be folded into three-dimensional target shapes by an external actuator. Petersen et al. (2011) presented a bipartite system for constructing walls, composed of mobile robots and passive building blocks that can be manipulated by the robots.

2.3.2 Locomotion

Modular robots have showed their versatility in locomotion. Yim (1994) presented statically stable locomotion gaits with his modular robot system, Polypod. Yim et al. (2000) presented PolyBot, a modular robot system that can locomote over a variety of terrain and switch between two locomotion modes by self-reconfiguration. Yim et al. (2007a) implemented a simple robotic system that can recover after disassembly from high-energy events. Sastra et al. (2009) implemented dynamic rolling, which was proven to be a fast and energy-efficient way of locomotion, with a modular robot system, CKbot. Burdick et al. (1994), Sfakiotakis and Tsakiris (2007), Lipkin et al. (2007), Hatton and Choset (2010) investigated gait generation for modular hyper-redundant (snake-like) robots.

2.3.3 Manipulation

The literature on multi-robot manipulation provides techniques for handling objects in a cooperative manner. When multiple robots manipulate a common object, it is necessary to control both the motion of the object and the internal forces exerted on the object (Murray et al. 1994). In addition, the dynamics of such systems is typically subject to unilateral constraints: the robots can only push or pull (for example, by cable tension) the object. Murray (1996), Sugar and Kumar (1999), Cheng et al. (2009), Fink et al. (2011), Bernard et al. (2011), Sreenath and Kumar (2013) provided solutions to the problem with applications to ground or aerial trans-

port. A group of literature shows that robotic object handling can be facilitated by caging with a team of multiple robots: Kosuge et al. (1999), Pereira et al. (2004), Montemayor and Wen (2005), Fink et al. (2008) presented approaches to developing decentralized control algorithms. Some literature on cooperative transport took inspiration from the behaviors of social insect colonies (Kube and Bonabeau 2000, Berman et al. 2011).

Part II

Grasping with Modular Robots

Chapter 3

Preliminaries: Caging and Grasping

This chapter introduces concepts and terminology relating caging and grasping that we use in developing our theory and algorithms.

3.1 Caging

A *cage* around an object bounds its mobility (Figure 3.1a): the caged object cannot be moved arbitrarily far from its original position without penetrating the surrounding effectors forming the cage. Equivalently, a cage can also be defined in terms of the mobility of the surrounding effectors by regarding the object as an obstacle: according to Rodriguez et al. (2012), a cage is a configuration of effectors that lies in a compact, connected component of the *free space* (LaValle 2006) of the system

of the effectors (note that the system is assumed to move as a single rigid body).

Rodriguez et al. (2012) formalized the concept of an *F-cage*, which is generally stricter than that of a cage. Let F be a scalar function defined on effector configurations. Then an F-cage is a configuration of the effectors that cages an object even if they have freedom to move while maintaining the value of F. An F-cage is an *F-squeezing* (*stretching*) cage if it still cages the object even if the effectors have freedom to move while decreasing (increasing) the value of F.

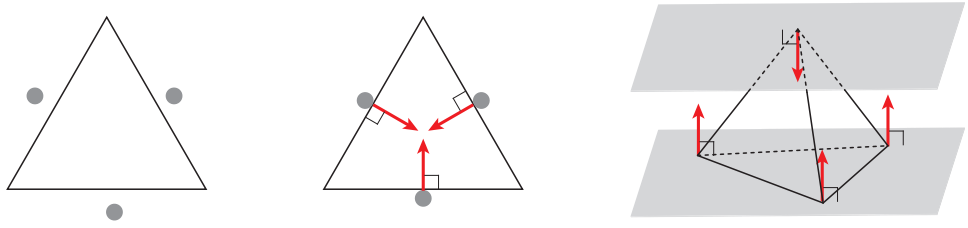


Figure 3.1: (a) Caging the triangle with the three point effectors. (b) Immobilizing the regular triangle with the three point effectors located at the center of each edge. (c) Clamping the tetrahedron with the two planar effectors contacting the vertex-face pair. The red arrows are involved unit contact wrenches under the assumption of frictionless, rigid, unilateral contact in (b), (c), and all upcoming figures.

3.2 Contact

We are mainly concerned with an object modeled as a polyhedron in contact with effector surfaces. There can then be three types of contact geometry: *point*, *line*, and *planar contact* (Mason and Salisbury 1985, Mason 2001). For example, Figure 3.1c shows one point and one planar contact between the two planar effectors and the object. A contact can apply a *contact wrench* (Murray et al. 1994, Mason 2001),

contact force/moment pair, given by the positive linear combinations of *unit contact wrenches* that are normalized, linearly independent vectors that span the vector space of the contact wrenches (see Figures 3.1b and 3.1c). Our work is based on an assumption that all contacts are frictionless, rigid, unilateral (unilateral contacts can only push an object); then, the unit contact wrenches are the vectors of the normalized screw coordinates only of the inward-pointing contact normals (Mason 2001). If we assume frictional, soft, bilateral contacts, more types of unit contact wrenches should be considered (Murray et al. 1994); this shows the conservativeness of the assumption.

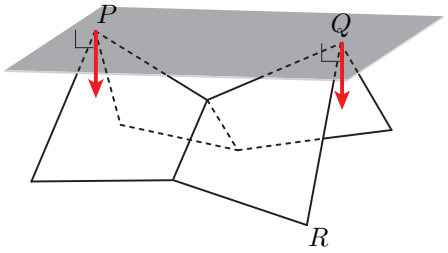


Figure 3.2: The plane is contacting the virtual edge $\overline{PQ}$ between the two "real" vertices P and Q.

A line contact can be made on a *virtual edge* (Peshkin and Sanderson 1986) that is composed only of the vertices delimiting the edge without the interior (Figure 3.2). It can be seen that a "real" edge and a virtual edge are indistinguishable in terms of their ability to produce contact wrenches, under the assumption of frictionless, rigid, unilateral contact, if we are given a sufficiently large effector surface that can contact all the delimiting vertices: in Figure 3.2, even if the virtual edge were real (like edge $\overline{QR}$), the number of unit contact wrenches would still be the same, that

is, two. Similarly, a planar contact can also be made on a virtual face.

3.3 Grasping

To *grasp* an object is to restrain its mobility by making contacts with effectors. A grasp can be in *equilibrium* if the net contact wrench, the sum of all contact wrenches, can be made zero in such a way that not all contact wrenches are equal to zero (Rimon and Burdick 1998a). If there is no object *twist*, linear/angular velocity pair (Murray et al. 1994), consistent with the contacts of an equilibrium grasp, the object is said to be *immobilized to the first order* (Rimon and Burdick 1998a) (or *form-closed* (Mason 2001)), that is, the configuration of the object is an isolated point in the free space. Even if such a twist exists, any finite motion may be restricted by considering *surface curvature effects*. For example, in Figure 3.1b, the object may instantaneously rotate about its centroid (so a first-order kinematic analysis does not predict immobility), but any finite rotation results in penetrating the effectors. This idea is formalized using the concept of *second-order immobility* (Rimon and Burdick 1998a,b). Seven (four) point effectors are required to immobilize a polyhedral object to the first (second) order with frictionless, rigid, unilateral contacts (Markenscoff et al. 1990, Czyzowicz et al. 1991). Such immobility conditions are purely geometric; information on contact geometry is thus sufficient to investigate first- or second-order immobility. An equilibrium grasp is called a *grasping cage* (Rodriguez et al. 2012) if it can also cage the object; a grasping cage is not

necessarily an immobilizing grasp.

3.4 Clamping

Clamping (Bose et al. 1996), also known as *parallel-jaw grasping*, is one way to realize equilibrium grasps by holding an object between two parallel planar "jaws." An object that is clamped can only move on the plane of the jaws, without penetrating them. If the jaws can exert frictional forces, the grasp can be force closed: an arbitrary contact wrench can be applied. Consider the *antipodal pair* of a convex polyhedral object, which is the intersection of the object with a pair of parallel support planes; an antipodal pair can thus be a vertex-vertex, vertex-edge, vertex-face, edge-edge, edge-face, or face-face pair. According to Bose et al. (1996), all convex polyhedra can be clamped with parallel-jaw grippers. The grippers are then on one of the last four types of antipodal pairs, that is, vertex-face, edge-edge, edge-face, or face-face; such an element pair can determine the *width* of a polyhedron, the minimum distance between two parallel supporting planes. Figure 3.1c shows a clamp on a vertex-face pair.

Chapter 4

Immobilizing Objects with Curved Effectors

In this chapter, we discuss how to immobilize a three-dimensional object using at most three contacts, each of which can be a point, line, or planar contact made with a curved effector that can be represented as a two-dimensional manifold with boundary. In Sections 4.1, 4.2, and 4.3, we present three types of immobilizing grasps that can be applied to objects modeled as polyhedra; we also discuss the geometry of effectors that can realize the grasps. In Section 4.4, we show that any polyhedron can be immobilized by the grasps and discuss how the assumption of polyhedral objects can be relaxed.

4.1 Immobilizing with Three Point Contacts

In this section, we show that it is possible to immobilize a polyhedron with three point contacts. We begin with choosing three vertices where the contacts can be made. See Figure 4.1a. Let P and Q be two vertices of the given polyhedron where two point contacts can be made with a pair of parallel planes perpendicular to $\overline{PQ}$; the inward-pointing (toward the interior of the polyhedron) contact normals at P and Q point toward each other. Let R be a vertex, not on ξ (the line of $\overline{PQ}$), where a point contact can be made with a plane such that the ray of the inward-pointing contact normal intersects ξ.

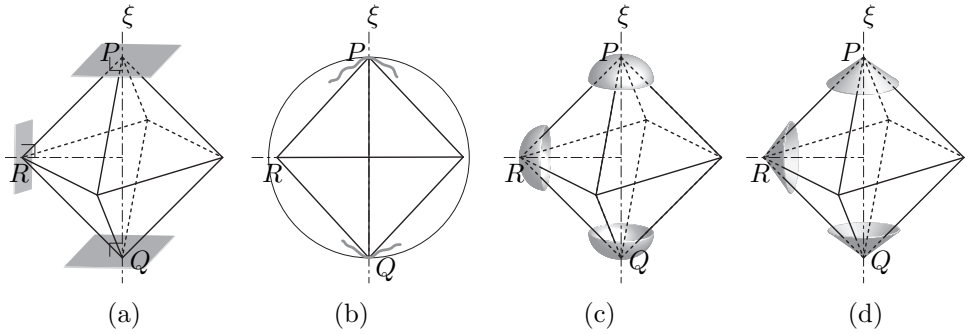


Figure 4.1: (a) (P, Q, R) is a vertex-vertex-vertex triple where three point contacts immobilizing the octahedron can be made. (b) The front view of the octahedron with the two curved effectors contacting P and Q. The effectors can locally be embedded inside the ball whose diameter has endpoins P and Q, except for the points contacting P and Q. (c) An immobilizing grasp by the effectors with a spherical surface. (d) An immobilizing grasp by the cone-shaped effectors.

Three point contacts on P, Q, and R made by curved effectors can immobilize the polyhedron if (1) a neighborhood of the effector surface around the point contacting P (Q), except for the point itself, can be embedded inside the ball whose diameter has endpoints P and Q (Figure 4.1b) and (2) a neighborhood of the effector

surface around the point contacting R, except for the point itself, can be embedded inside the cylinder of radius $d(R, \xi)$ with axis ξ, where $d(R, \xi)$ denotes the shortest distance between R and ξ. If condition (1) is satisfied, as can be seen in Figure 4.1b, the polyhedron cannot move at all except for rotating about ξ because $\overline{PQ}$ is at a configuration that is isolated in the free space: any finite displacement of $\overline{PQ}$ results in penetrating the effectors at P or Q. Furthermore, if condition (2) is satisfied, the polyhedron cannot rotate about ξ because the rotation results in penetrating the effector at R. Two example grasps are shown in Figures 4.1c and 4.1d: effectors with a spherical surface of a sufficiently small radius (Figure 4.1c) or effectors with a cusp (Figure 4.1d) can satisfy the two conditions.

4.2 Immobilizing with a Point, a Line, and a Planar contact

We here show that it is possible to immobilize a polyhedron with one point, one line, and one planar contact. We begin with choosing a vertex, an edge, and a face where the contacts can be made. See Figure 4.2a. At P and $\square QRST$, a point and a planar contact can be made with a pair of parallel planes. The ray of the inward-pointing contact normal at the vertex intersects the interior of the face; the inward-pointing contact normals at the face point toward the vertex. Among the edges incident to P, choose the one with the least slope with respect to the face,

that is, $\overline{PS}$.

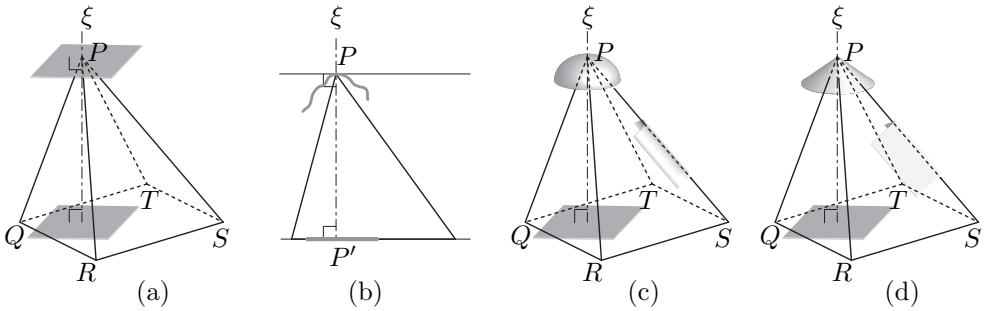


Figure 4.2: (a) $(P, \overline{PS}, \square QRST)$ is a vertex-edge-face triple where a point, a line, and a planar contact immobilizing the pyramid can be made. (b) The front view of the object with the two effectors contacting the vertex and the face. The curved effector contacting the vertex, P, can locally be embedded inside the space between the two supporting planes, except for the point contacting P. (c) An immobilizing grasp by the effectors with a spherical, a cylindrical, and a planar surface. (d) An immobilizing grasp where the point (line) contact is made by the cone-shaped (V-shaped) effector.

Three (one point, one line, and one planar) contacts on the vertex (P), edge ($\overline{PS}$), and face ($\square QRST$) by curved effectors can immobilize the polyhedron if (1) the contact area of the planar contact contains a neighborhood of P', the foot of perpendicular from P to the face (Figure 4.2b), (2) a neighborhood of the effector surface around the point contacting P, except for the point itself, can be embedded inside the half-space below (toward the interior of the polyhedron) the plane contacting P (Figure 4.2b), and (3) a neighborhood of the effector surface around the line segment contacting $\overline{PS}$, except for the line segment itself, can be embedded inside the cone formed by rotating $\overline{PS}$ about ξ (the line of $\overline{PP'}$). If conditions (1) and (2) are satisfied, as can be seen in Figure 4.2b, the polyhedron cannot move at all except for rotating about ξ because the two effectors at the vertex and face are not only clamping it but also restricting any translation. Furthermore, if condition

(3) is satisfied, the polyhedron cannot rotate about ξ because the rotation results in penetrating the effector at $\overline{PS}$. Two example grasps are shown in Figures 4.2c and 4.2d.

4.3 Immobilizing with Two Line Contacts

We here show that it is possible to immobilize a polyhedron with two line contacts. We begin with choosing two edges where the contacts can be made. See Figure 4.3a. $\overline{PQ}$ and $\overline{RS}$ are two skew edges of the given polyhedron where two line contacts can be made with a pair of parallel planes that are perpendicular to ξ, the common perpendicular of the edges. ξ and the two edges intersect in the interior of the edges. The inward-pointing contact normals at one edge point toward the other edge.

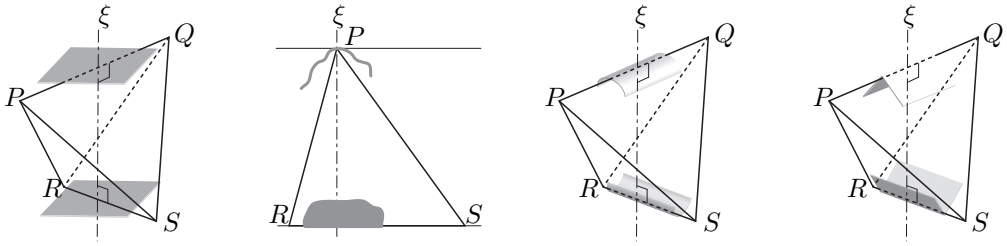


Figure 4.3: (a) $(\overline{PQ}, \overline{RS})$ is an edge-edge pair where two line contacts immobilizing the tetrahedron can be made. (b) The front view of the object with the two curved effectors contacting the edges. The curved effectors can locally be embedded inside the space between the two supporting planes (except for the line segments contacting the object). (c) An immobilizing grasp by the effectors with a cylindrical surface. (d) An immobilizing grasp by the two V-shaped effectors.

Two line contacts on the edges by curved effectors can immobilize the polyhedron if a neighborhood of the effector surface around the line segment contacting $\overline{PQ}$

($\overline{RS}$), except for the line segment itself, can be embedded in the half-space below (above) the plane contacting it (Figure 4.3b). Note that the two contact areas (the line segments) and ξ should intersect. If the condition is satisfied, as can be seen in Figure 4.3b, the effectors are not only clamping the polyhedron but also restricting any planar motion: one of the effectors only allows the object to translate along the line of its contact, but such motion is not allowed by the other effector. Two example grasps are shown in Figures 4.3c and 4.3d.

4.4 Analysis

This section presents an analysis showing that the three types of grasps discussed in Sections 4.1, 4.2, and 4.3 are complete: every polyhedron can be immobilized by applying at least one of the grasps. We break the analysis into two theorems.

First, we show that the grasp of three point contacts discussed in Section 4.1 suffices to immobilize all polyhedra.

Theorem 1. *Every polyhedron can be immobilized by three frictionless, rigid, unilateral point contacts made by appropriately concave effectors.*

Proof. Every polyhedron has a vertex-vertex pair (P, Q) that admits two parallel supporting planes perpendicular to $\overline{PQ}$ and contacting only P and Q, respectively, in such a way that the interior of the polyhedron is between the planes. Consider the collection of the vertices of the polyhedron. Let (P, Q) be a pair of vertices

determining the maximum distance between two vertices of the collection. Consider two planes Π_P and Π_Q perpendicular to $\overline{PQ}$ and contacting the polyhedron respectively at P and Q. No other vertex of the polyhedron can be located on Π_P and Π_Q because (P, Q) determines the maximum distance: Π_P (Π_Q) is supporting the polyhedron only at P (Q). Therefore, (P, Q) is a desired vertex pair.

Next, we can find an additional vertex, not on ξ (the line of $\overline{PQ}$), which admits a supporting plane whose inward-pointing normal at the vertex directly points to ξ. Consider one vertex that is the most distant from ξ and denote the vertex as R; let ξ' be a line parallel to ξ and passing through R. If R is the only vertex of the polyhedron on ξ', R must be a pointed vertex that admits a supporting plane whose inward-pointing normal at R directly points toward ξ. R is then the desired vertex. In case there are multiple vertices lying on ξ', first choose a point on ξ' as its origin, then pick the vertex that is the most distant from the origin, and denote the vertex as R. It can be seen that R is the desired vertex.

We finally get immobility with three appropriately concave effectors respectively contacting P, Q, and R as explained in Section 4.1. Note that P, Q, and R are on the convex hull of the polyhedron; otherwise, P and Q do not determine the maximum distance and R is not the most distant from ξ, either. □

We now show that other two types of grasps discussed in Sections 4.2 and 4.3 are also complete in the sense that every polyhedron whose vertices are in *general position*, not admitting any pair of either two planes or a line and a plane that are

parallel, can be immobilized by at least one of those grasps.

Theorem 2. *Every polyhedron whose vertices are in general position such that there is no pair of either two planes or a line and a plane that are parallel, where each line (plane) is determined by a distinct collection of two (three) vertices, can be immobilized by either (1) one point, one line, and one planar contact or (2) two line contacts made by appropriately concave effectors; the contacts are frictionless, rigid, unilateral.*

Proof. Consider the convex hull of the given polyhedron. Because the convex hull does not have any pair of either two faces or an edge and a face that are parallel, it can be clamped at one of its antipodal vertex-face or edge-edge pairs. The convex hull (and thus the original polyhedron) can then be immobilized using the pair by applying the grasp discussed in Section 4.2 or 4.3. □

Note that the grasps employed in Theorem 2 might have contacts on virtual edges or faces that are on the convex hull, but not belonging to the original polyhedron.

Remark : The three types of grasps can actually be applied to immobilizing a wider range of objects in addition to polyhedra. Essentially, if there is a set of the right contacts made with effectors having the right curvature properties, as discussed in Sections 4.1, 4.2, and 4.3, then the object is immobilized. First, the place where a point contact is made does not have to be actually pointed like a polyhedron vertex. For example, see the point contacts at P, Q, and R in Figure 4.1c; for the

effectors to immobilize the object, it is only required that the actual geometry of the object around P, Q, or R does not intersect the effector surface, whether the actual geometry is smooth or pointed. Similarly, the place where a line (planar) contact is made does not have to be a perfect polyhedron edge (face).

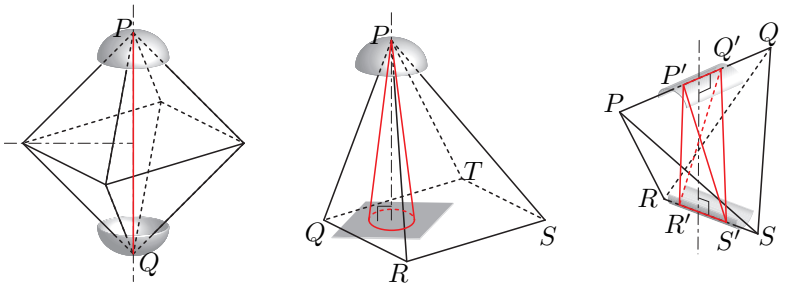
Chapter 5

Caging Objects with Curved Effectors

Based on the three types of immobilizing grasps discussed in Chapter 4, it can be seen that every polyhedron can be caged by two curved effectors.

Corollary 1. *Every polyhedron can be caged by two appropriately concave effectors, each of which is accommodating a single vertex, edge, or face.*

Proof. According to Section 4.1, 4.2, or 4.3, it is possible to cage a polyhedron with two curved effectors making contacts at a vertex-vertex, vertex-face, or edge-edge pair. For example, Figure 5.1 shows three types of cages obtained by the two effectors at the antipodal pairs in Figures 4.1c, 4.2c, and 4.3c, respectively. According to Theorems 1 and 2, every polyhedron can be caged by at least one of the three types of cages. □



(a) (b) (c)

Figure 5.1: Cages of two curved effectors. The inscribed shapes colored red (the line segment in (a), the right circular cone in (b), and the tetrahedron in (c)) are used to establish sufficient conditions for caging in Section 5.1.

Although the effectors are contacting the objects in Figure 5.1, a cage does not necessarily have to make contacts with the object. In order to see if an object is caged, it is necessary to take overall effector geometry into account because it is required to verify the compactness of the component of the free space to which the configuration of the object belongs, which is a nonlocal property. For example, the "depth" of a curved effector is critical in our discussion on caging here. In contrast, recall that we need only local curvature properties in establishing the immobilizing grasps in Chapter 4.

In this chapter, we address how to establish sufficient conditions for caging three-dimensional objects in analytical (Section 5.1) and empirical (Section 5.2) manners. In Section 5.3, we present a theoretical analysis on how to acquire stable grasps from the cages and discuss how our assumption of polyhedral objects can be relaxed.

5.1 Analytical Method

In this section, we present three types of cages that are derived from the three types of immobilizing grasps. Considering simpler object geometry that can be inscribed in the original shape, we discuss how to establish sufficient conditions for caging in an analytical manner.

5.1.1 Caging on a Vertex-Vertex Pair

Consider a vertex-vertex pair of a given polyhedron such as (P, Q) in Figure 5.1a, allowing us to establish the immobilizing grasp discussed in Section 4.1. We here investigate how to cage the two vertices assumed to be rigidly connected (see the red line segment $\overline{PQ}$ in Figure 5.1a).

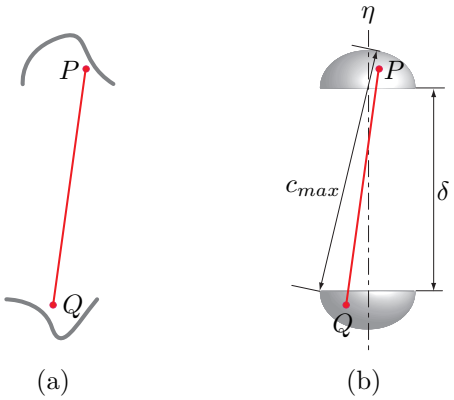


Figure 5.2: (a) The planar view shows P and Q contained in the curved effectors. (b) A cage of two hemispherical effectors.

Consider two curved effectors that can contain the vertices, as shown in Figure 5.2a. If the maximum clearance between the effectors is less than $d(P, Q)$, the distance between P and Q, they can cage the line segment $\overline{PQ}$. For example, con-

sider a pair of effectors of the same hemisphere that are facing each other and only allowed to relatively translate on their common axis, η (Figure 5.2b). The maximum clearance between the two effectors is then the distance between one point on the boundary of one effector and the foot of perpendicular from the point to the other effector, denoted as c_{max} in the figure. If δ, the distance between the two planes on which the boundaries of the effectors lie (Figure 5.2b), is small enough to guarantee $c_{max} < d(P, Q)$, $\overline{PQ}$ (and thus the original object) is caged.

5.1.2 Caging on a Vertex-Face Pair

Consider a vertex-face pair of a given polyhedron such as $(P, \square QRST)$ in Figure 5.1b, allowing us to establish the immobilizing grasp discussed in Section 4.2. We here investigate how to cage a right circular cone that can be inscribed in the convex hull of the vertex and face (see the red cone with apex P in Figure 5.1b).

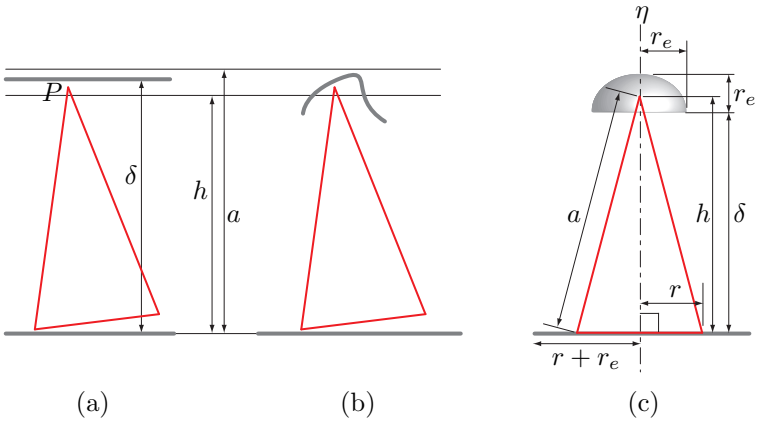


Figure 5.3: (a) The planar view shows the red cone in Figure 5.1b is containted between the two parallel planar effectors. (b) The curved and planar effectors are caging the cone. (c) A cage of the hemispherical and planar effectors. The hemispherical effector is only allowed to translate along its axis η.

First, suppose that two infinitely large planar effectors are clamping the right circular cone at the apex and base. We now allow the effectors to move in such a way that they remain parallel to each other (Figure 5.3a). If their distance δ is less than a, the side length of the cone, the cone can stably be clamped again by decreasing δ; moreover, the distance between the apex and the effector at the base is always larger than h, the height of the cone. It can then be seen that any curved effector containing the vertex in such a way that its maximum height is less than a and the boundary lies below h cages the cone along with the planar effector at the base (Figure 5.3b). For example, consider a hemispherical effector configured such that its boundary is parallel to the planar effector and only allowed to relatively translate along its axis, η (Figure 5.3c). Then the cone (and thus the original object) is caged with the effectors if δ, the distance between the planar effector and the plane of the boundary of the hemispherical effector, is small enough to guarantee (1) the apex is contained in the hemispherical effector, (2) $\delta + r_e < a$, and (3) $\delta < h$: (2) and (3) guarantee that the apex cannot escape from the hemispherical effector containing it by the analysis above. If the base belongs to a real face of the original object, the planar effector at the base only has to be as large as a disk of radius $r + r_e$, in order to support the base (Figure 5.3c). Otherwise (if the face is virtual), the planar effector should be at least as large as the face itself.

5.1.3 Caging on an Edge-Edge Pair

Consider an edge-edge pair of a given a polyhedron such as $(\overline{PQ}, \overline{RS})$ in Figure 5.1c, allowing us to immobilize the polyhedron as discussed in Section 4.3. We here investigate how to cage a tetrahedron that can be inscribed in the convex hull of the two edges (see the red tetrahedron $P'Q'R'S'$ in Figure 5.1c).

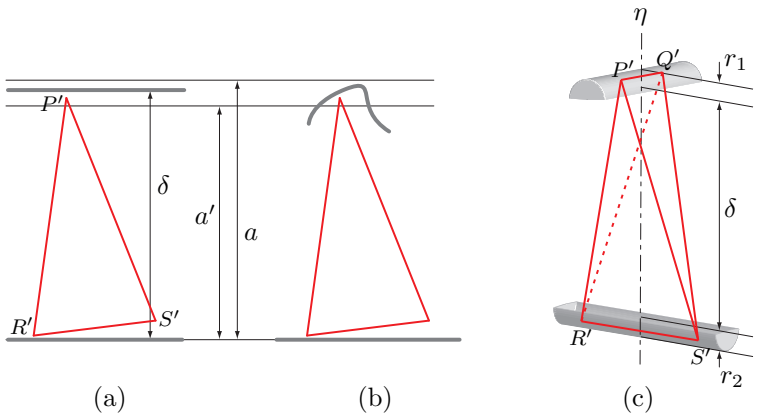


Figure 5.4: (a) The planar view shows the red tetrahedron in Figure 5.1c is contained between the two parallel planar effectors. (b) The planar effector at the bottom and the curved effector are caging the tetrahedron. (c) A cage of the two half-cylindrical effectors. They are only allowed to translate along their common perpendicular, η.

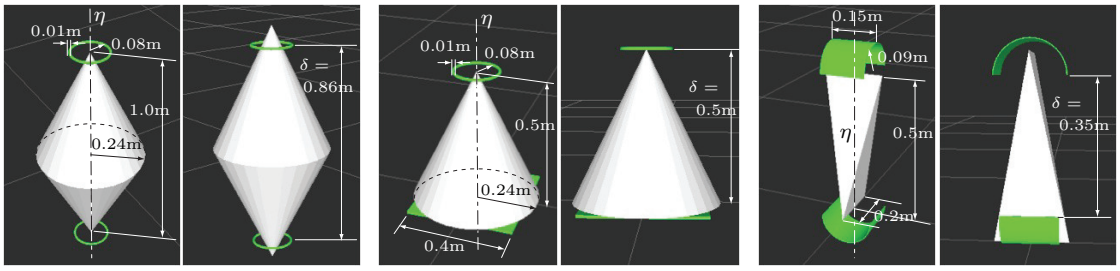
First, suppose that two infinitely large planar effectors are clamping the tetrahedron $P'Q'R'S'$. We now allow the effectors to move in such a way that they remain parallel to each other (Figure 5.4a). If their distance δ is less than a, the smallest value among $d(\overline{P'Q'}, R')$, $d(\overline{P'Q'}, S')$, $d(\overline{R'S'}, P')$, and $d(\overline{R'S'}, Q')$ (each term denotes the shortest distance between the edge and the vertex), the tetrahedron can stably be clamped again by decreasing δ. Because of the stability, the heights of P' and Q' from the planar effector at the bottom has a lower bound, a', which can be found by rotating the tetrahedron about $\overline{R'S'}$ lying on the planar effector. It can

then be seen that any curved effector containing edge $\overline{P'Q'}$ in such a way that its maximum height is less than a and the boundary lies below a' (Figure 5.4b), cages the tetrahedron along with the planar effector at the bottom. Similarly, the heights of R' and S' are upper bounded; then, the planar effector at the bottom can also be replaced by a curved effector. For example, consider half-cylindrical effectors configured such that their boundaries are parallel to each other and only allowed to relatively translate along η (Figure 5.4c). Then the tetrahedron (and thus the original object) is caged with the effectors if δ, the distance between the two planes on which the boundaries of the effectors lie, is small enough to guarantee (1) the edges are contained in the effectors, (2) $\delta + r_1 + r_2 < a$, and (3) r_1 (r_2) is large enough to contain the lowest (highest) positions of P' and Q' (R' and S'): (2) and (3) guarantee that the edges cannot escape from the effectors containing them by the analysis above. The half-cylindrical effectors should be at least as long as the edges of the original object that they are containing.

5.2 Empirical Method

Sufficient conditions for caging can also be established in an empirical manner by making use of off-the-shelf motion planning algorithms, particularly in case effector geometry is analytically challenging.

First, we consider a scenario where two torus-shaped effectors are caging an object on a vertex-vertex pair. Figure 5.5a shows how we set up experiments with



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

Figure 5.5: Establishing cages in an empirical manner. (a), (b) Caging on a vertex-vertex pair. (c), (d) Caging on a vertex-face pair. (e), (f) Caging on an edge-edge pair.

the cone-like object. The effectors are only allowed to translate along η, their common axis of rotation. Let δ denote the distance between the two planes of the two circles, each of which represents the center of each torus tube. Changing the value of δ, we see if a motion planning algorithm can find a path for the object to reach a target configuration sufficiently far from the grip of the effectors. When $\delta = 0.86$m (Figure 5.5b), the motion planner (an RRT-based algorithm) failed to find such a path in 10 trials (each trial took $0.01 \sim 0.06$ seconds on a 2.53GHz/4GB machine). Therefore, we can empirically conclude that the object is caged if $\delta \leq 0.86$m.

Second, we consider a scenario where a torus-shaped and a planar effector are caging an object on a vertex-face pair. Figure 5.5c shows our experimental setup with the cone object: the square-shaped planar effector is contacting the base of the cone; the torus-shaped effector is allowed to translate along η, its axis of rotation collinear to the axis of the cone. When $\delta = 0.5$m (Figure 5.5d), where δ is the distance between the planar effector and the plane on which the circle representing

the center of the torus tube lies, the motion planner failed to find an escaping path for the object in 10 trials (each trial took $0.01 \sim 0.04$ seconds on a 2.53GHz/4GB machine). Therefore, we can empirically conclude that the object is caged if $\delta \leq$ 0.5m.

Third, we consider a scenario where two open-ended, half-cylindrical effectors are caging an object on an edge-edge pair. In the experimental setup shown in Figure 5.5e, the effectors are only allowed to translate along η, their common perpendicular. The motion planner could not find an escaping path for the object when $\delta = 0.35$m (Figure 5.4d), where δ is the distance between the two planes on which the boundaries of the effectors lie, in 10 trials (each trial took $0.02 \sim 0.04$ seconds on a 2.53GHz/4GB machine). Therefore, we can empirically conclude that the object is caged if $\delta \leq 0.35$m.

5.3 Analysis

This section presents an analysis showing how to get a grasping cage (recall Section 3.3) from any of the cages discussed so far. We also discuss how to relax the assumption of polyhedral objects.

Theorem 3. *For the cages discussed in Sections 5.1 and 5.2, a grasping cage is obtained if the two effectors are controlled such that the relative velocity is along η and δ monotonically decreases until contact is established, which is assumed to be frictionless, rigid, unilateral (see Figures 5.2b, 5.3c, 5.4c, and 5.5 for η and δ).*

Proof. We first show that δ is a *grasping function* (Rodriguez et al. 2012) for the cages. In each cage, the configuration space of the two effectors can be represented as $\mathcal{M} = SE(3) \times SE(3)$; δ is a semi-algebraic scalar function $\delta : \mathcal{M} \to \mathbb{R}$ invariant with respect to the rigid transformations of the effectors as a whole in that it is the distance between the effectors. Furthermore, the preimages of δ do not cage the object below (above) a certain value m (M) such that $m < M$, for example, $m = 0$ and $M = h$ in Figure 5.3c. Then δ is a grasping function according to Rodriguez et al. (2012).

In addition, the cages are *δ-squeezing cages* (Section 3.1) in that the object remains caged even if δ decreases. Then, there exists a path in $\mathcal{M}$ that leads the effectors into a configuration that can realize a grasping cage. Furthermore, in terms of the one-dimensional set representing the relative configuration space of the two effectors, δ can be considered as a convex, that is, linear, function. Then, by the result of Rodriguez et al. (2012), we get to a grasping cage only by moving the effectors such that δ monotonically decreases. □

Rodriguez et al. (2012) discovered that the role of grasping functions in grasping is analogous to that of Lyapunov functions in stability analysis. Specifically, Theorem 3 shows that an object caged by any of our cages can be stabilized by simply moving the effectors closer to each other by relative translation. A translation that monotonically decreases δ in the cages will be referred to as a *squeezing motion* in the remaining discussion. During a squeezing motion, we may add more contacts

to further secure the grasp.

Remark : The three types of cages, each of which makes use of a vertex-vertex, a vertex-face, or an edge-edge pair, can actually be applied to a wider range of objects in addition to polyhedra. Essentially, if the polyhedron model of an object is geometrically conservative such that the vertices, edges, and faces of the model are inscribed in the actual object geometry, then caging the polyhedron guarantees that the actual object is also caged. The stability of squeezing motions is also guaranteed by the conservativeness. This observation implies that the sufficiency of our caging conditions allows us to address errors/uncertainties in sensing and control relating to grasp acquisition. Note, however, that if the polyhedron approximation is too conservative, the cage might intersect the actual object geometry.

Chapter 6

Synthesizing Grasps and Cages

In this chapter, we present an algorithm for finding element pairs/triples for the immobilizing grasps and cages discussed in the previous chapters.

Our pseudocode is presented in Algorithm 1. Given the model of an object, the output of the algorithm specifies where to place effectors in order to obtain the immobilizing grasps/cages. The following paragraphs elaborate each line of the algorithm.

Algorithm 1 Finding element pairs/triples for immobilizing/caging

Input: Object model: a polygonal mesh.
Output: Element pairs and triples labeled with instructions on placing effectors.

1: Compute the convex hull of the mesh.
2: For the convex hull, search for element triples (pairs) for the grasps of Sections 4.1 and 4.2 (Section 4.3).
3: Append instructions on placing effectors to each element pair or triple.

Line 1: We first compute the convex hull of the object model. The convex hull is essentially our object in the algorithm by considering that the vertices, edges, and

faces lying on the convex hull suffice to guarantee the completeness of our grasps and cages (see Theorems 1, 2 and Corollary 1). It takes $O(n \log n)$ expected time to compute the convex hull of a given polyhedron, where n is the number of the vertices (de Berg et al. 2000).

Line 2: We then compute the antipodal pairs of the convex hull that are vertex-vertex (for the grasps of Section 4.1), vertex-face (for the grasps of Section 4.2), and edge-edge pairs (for the grasps of Section 4.3). Given a polyhedron with n vertices, its antipodal pairs can be computed in $O(n^2)$ time by applying a technique introduced by Brown (1979). Among them, choose the ones that admit supporting planes as illustrated in Figures 4.1a, 4.2a, and 4.3a; it takes $O(1)$ time to see if each pair satisfies the condition. For each vertex-vertex pair, we search for another vertex that is the most distant from the line of the two vertices (see the proof of Theorem 1); as a result, the pair is augmented into a triple of the three vertices in $O(n)$ time. For each vertex-face pair, we search for the edge with the least slope with respect to the face, among the ones incident to the vertex (recall Figure 4.2a); as a result, the pair is augmented into a triple of the vertex, edge, and face in $O(n)$ time.

Line 3: Finally, each element pair or triple from Line 2 is labeled with instructions on how to place effectors. First, the label shows if the constituent elements are `real` (lying on the original mesh) or `virtual` (lying only on the convex hull). Second, the label shows the positions and orientations at which effectors should aim.

For a vertex-vertex-vertex triple, we specify the coordinates of the vertices and the inward-pointing normals of the three supporting planes. For a vertex-edge-face triple, for example, P, $\overline{PS}$, and $\square QRST$ in Figure 4.2a, we specify the coordinates of P, the midpoint of $\overline{PS}$, and the foot of perpendicular from P to $\square QRST$; the inward-pointing normals of the two supporting planes; and the normal vector to $\overline{PS}$ that directly points to ξ. For an edge-edge pair, for example, $\overline{PQ}$ and $\overline{RS}$ in Figure 4.3a, we specify the coordinates of the two intersections between ξ and the two edges, along with the inward-pointing normals of the two supporting planes. Figure 6.1 shows such a label for an edge-edge pair. The labeling can be done in $O(1)$ time for each pair or triple.

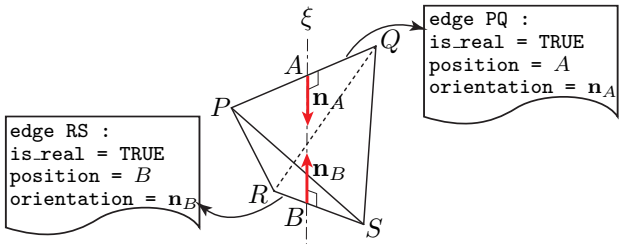
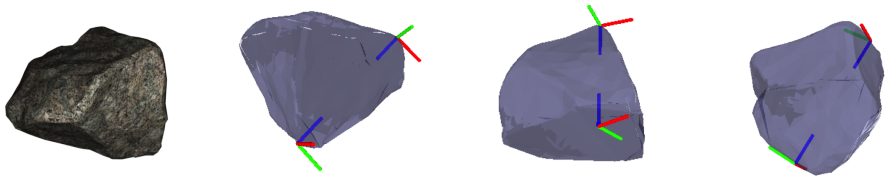


Figure 6.1: The label for the edge-edge pair, $(\overline{PQ}, \overline{RS})$.

Figure 6.2a shows an example object; Figures 6.2b, 6.2c, and 6.2d show some element pairs found by running the algorithm. The convex hull of the rock model has 162 vertices, 480 edges, and 320 faces; it took 0.02 seconds to find 21 vertex-vertex, 9 vertex-face, and 18 edge-edge pairs with our C++ implemention running on a 2.53GHz/4GB machine. The vertex-vertex and vertex-face pairs (Figures 6.2b and 6.2c) can be augmented into the element triples for immobilizing grasps as

explained in Line 2.



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

Figure 6.2: (a) A polyhedral rock model with 1,000 faces (courtesy: Malcolm Lambert, Intresto Pty Ltd.). (b), (c), and (d) respectively show a vertex-vertex pair, a vertex-face pair, and an edge-edge pair found by running our algorithm. The reference frames are positioned and oriented such that the origin is at the contact position and the z-axis is along the contact normal, according to the label of each element pair.

A given object model can be immobilized by making contacts with any of the resultant element pairs or triples: we contact target elements with curved effectors, whose curvature is sufficiently large, positioned and oriented as the label specifies. For caging purposes, we do not have to search for the third element in Line 2. Furthermore, effectors do not necessarily have to be controlled to make contacts with their target elements: the target positions for a pair of effectors caging an object can be receded from the positions specified in the label as long as their distance δ remains less than δ^*, the largest acceptable value of δ for the cage to be valid, along the line of the inward-pointing normals. Note, in fact, that the target positions can be receded even farther because the caging conditions are sufficient.

Remark : Input meshes to Algorithm 1 do not have to be *closed*, that is, homeomorphic to a closed manifold (compact manifold without boundary), because the computations do not depend on the closedness. This implies that the algorithm

can be applied to object models that are imperfectly perceived due to, for example, visual occlusion.

Because all the elements found by Algorithm 1 (possibly except for the edge of a vertex-edge-face triple) are on the convex hull, that is, the outer frontier of the object, some of them can be virtual. An effector for a virtual edge or face should be large enough to contact all the vertices delimiting the virtual element. If it were not for an effector with a sufficiently large surface for a given virtual element, we would instead need to make contact with a smaller real element in the interior of the convex hull. Then, we may forgo computing the convex hull and proceed with the original mesh although it can take more time and the returned elements may be less easier to access in case they are in the interior of the convex hull.

Chapter 7

Extending Our Theory

This chapter discusses how to extend the collection of grasps/cages discussed in Chapters 4 and 5: more types of grasps/cages can be added by employing other types of antipodal pairs. We also discuss grasping and caging two-dimensional objects with curved effectors on the plane.

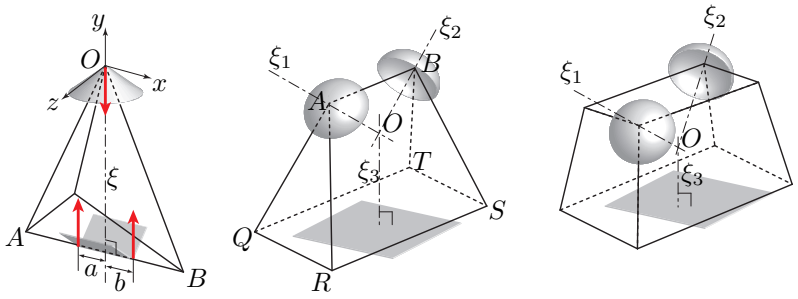


Figure 7.1: (a) An immobilizing grasp at an antipodal vertex-edge pair with the cone- and V-shaped effectors. The vertex, O, is the origin of the reference frame whose x-axis is parallel to the edge $\overline{AB}$ and y-axis is collinear to ξ. (b) An immobilizing grasp at an antipodal edge-face pair. (c) An immobilizing grasp at an antipodal face-face pair.

We first show that it is possible to immobilize a polyhedron with one point and

one line contact made by curved effectors, which are frictionless, rigid, unilateral. See Figure 7.1a. O and $\overline{AB}$ are a vertex-edge pair of the given polyhedron where a point and a line contact can be made with a pair of parallel planes that are perpendicular to ξ, the line connecting the vertex and its foot of perpendicular to the edge (the foot of perpendicular is in the interior of the edge). The inward-pointing contact normal at the vertex points toward the edge and vice versa. With respect to the reference frame whose origin is O, the three unit contact wrenches (the red arrows) are:

$$(\mathbf{s}_1, \mathbf{s}_{01}) = (0, -1, 0, 0, 0, 0)$$

$$(\mathbf{s}_2, \mathbf{s}_{02}) = (0, 1, 0, 0, 0, b)$$

$$(\mathbf{s}_3, \mathbf{s}_{03}) = (0, 1, 0, 0, 0, -a)$$

For a differential twist $(\mathbf{t}, \mathbf{t}_0) = (t_1, t_2, t_3, t_4, t_5, t_6)$ to be consistent with the unit contact wrenches, each of the reciprocal products of the twist and the unit contact wrenches must be zero:

$$-t_5 = 0$$

$$t_5 + bt_3 = 0$$

$$t_5 - at_3 = 0$$

Then, the solutions are of the form:

$$(\mathbf{t}, \mathbf{t}_0) = (t_1, t_2, 0, t_4, 0, t_6)$$

because t_3 and t_5 must be zero. If O is being contacted by the apex of a cone-shaped effector, as can be seen in Figure 7.1a, the velocity of O cannot have nonzero x- and z-components with respect to the reference frame; otherwise, O will penetrate the effector. In other words, t_4 and t_6 can also be made zero; then, the solutions must actually be of the form:

$$(\mathbf{t}, \mathbf{t}_0) = (t_1, t_2, 0, 0, 0, 0)$$

This is a zero-pitch screw (pure rotation) where the first three components give the direction of the rotation axis, which should pass through O. Such rotation instantaneously moves $\overline{AB}$ in a direction perpendicular to the plane of O, A, and B, but can completely be restricted by contacting $\overline{AB}$ with a V-shaped effector, as can be seen in Figure 7.1a. The polyhedron is then immobilized with the point and line contacts on the vertex-edge pair; at the same time, the two effectors are also forming a cage, with their distance along ξ as a grasping function similarly to the previous two-effector cages.

We additionally show that it is possible to immobilize a polyhedron with one planar and two point contacts made by curved effectors, which are frictionless, rigid, unilateral. See Figure 7.1b. Consider an edge-face pair that determines the width of the convex hull of the given polyhedron, denoted as $\overline{AB}$ and $\square QRST$. At each vertex of the edge, consider a set of unit contact wrenches, each of which is perpendicular to one of the faces containing the vertex and points toward the interior of the polyhedron; then each of A and B has three such unit contact wrenches

because there are three faces meeting at each vertex. Also consider a set of unit contact wrenches at the face, each of which is perpendicular to the face toward the interior at one vertex delimiting the face; then $\square QRST$ has four such unit contact wrenches at Q, R, S, and T. The grasp of one planar and two point contacts shown in Figure 7.1b can always be made in equilibrium, by considering that the edge-face pair even admits a clamp. To be more specific, consider the following linear feasibility program

$$\text{Find } \{c_i\} \text{ such that } \sum c_i \hat{\mathbf{w}}_i = \mathbf{0}, c_i > 0, \text{ and } \sum c_i = 1 \tag{7.0.1}$$

where $\{\hat{\mathbf{w}}_i\}$ is the collection of all the unit contact wrenches. The grasp can be made in equilibrium by configuring the effectors according to a solution to the program: the terms of $\sum c_i \hat{\mathbf{w}}_i$ can be divided into three groups affiliated with the two vertices and face; each effector should be configured to exert the sum of one group of the wrenches, which is the positive linear combination of the unit contact wrenches at the element the effector is contacting. For example, in Figure 7.1b, the two hemispherical effectors are configured such that their axes of symmetry, ξ_1 and ξ_2, pass through A and B, the points of contact, and intersect at O, whose foot of perpendicular to the plane of $\square QRST$ lies on the area of the planar contact. Under the equilibrium, the object can only move on the plane of the planar contact. If the two curved effectors on the two vertices are sufficiently concave, the object can in fact be immobilized because any finite motion of $\overline{AB}$ results in penetrating the effectors. This idea can also be applied to immobilizing a polyhedral object on a

face-face antipodal pair (Figure 7.1c).

The objects in Figures 7.1b and 7.1c are also caged by the three effectors. One approach to establishing a grasping function here is to assume a control strategy that moves the three effectors as a three-fingered gripper system with one parameter that controls the opening of the gripper, as also discussed by Davidson and Blake (1998a) for three point fingers. Let δ denote the opening parameter of the gripper: as δ increases, the three effectors shown in Figures 7.1b and 7.1c monotonically move away from O by translation along the axes (ξ_1, ξ_2, and ξ_3). In each of the grasps shown in Figures 7.1b and 7.1c, all the instantaneous motions of the object penetrate the effectors, which in turn guarantees that the configuration of the object is completely isolated from its free space (Rimon and Burdick 1998a). Then it can be seen that the opening parameter δ is a grasping function similarly to Theorem 3. Constructing caging conditions for three effectors may be harder than two-effector cages; for point effectors, refer to Davidson and Blake (1998a).

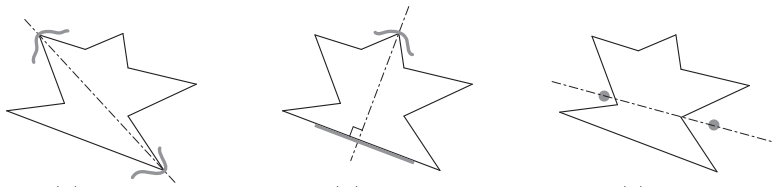


Figure 7.2: The polygon is immobilized by (a) two point contacts at the vertex-vertex pair determining the diameter (b) a point and a line contact at the vertex-edge pair determining the width. (c) The polygon is caged by two point effectors. If the effectors were contacting the concave vertices, the polygon would be immobilized.

Two-dimensional grasping : Our approach can also be applied to grasping two-dimensional objects with effectors on the plane, which can be represented as one-dimensional manifolds with boundary. Every polygonal object can be immobilized on the plane by two curved effectors of appropriately chosen dimensions. In Figure 7.2a, the two curved effectors are immobilizing the object by two point contacts; in Figure 7.2b, the object is immobilized by a point and a line contact. The two types of grasps are complete: every polygon can be immobilized by two point contacts made by appropriately concave effectors on a pair of vertices determining the diameter (the maximum distance between two vertices); every polyhedron whose vertices are in general position such that there is no pair of two parallel lines, where each line is determined by a distinct collection of two vertices, can be immobilized by a point and a line contact made by appropriately concave effectors on a vertex-edge pair determining the width.

As a corollary, every polygon can be caged by two curved effectors: the effectors in Figures 7.2a and 7.2b are also caging the polygon. Sufficient conditions for caging can be derived by applying the results of Sections 5.1 and 5.2 (see the similarity between Figure 5.2 and Figure 7.2a (or Figure 5.3 and Figure 7.2b)).

We can also add more types of grasps and cages; for example, as shown in Figure 7.2c, two point effectors suffice to immobilize/cage some concave polygons as discussed by Vahedi and van der Stappen (2008a).

Chapter 8

A Modular Approach to Whole-Arm Grasping

In this chapter, we address how our theory can be applied to a scenario where two collaborating manipulator arms, which can be the arm-torso chain of a humanoid robot or two collaborating industrial robot arms, are grasping objects with the curved effectors as their end-effectors. The stability of the two-effector cages discussed in Chapter 5 allows us to add more contacts not necessarily from the end-effectors; the scenario may then be called *whole-arm grasping* as can be seen in Figure 8.1. Whole-arm grasping can particularly be effective for grasping large, bulky objects such as rocks, with relatively small end-effectors. Without fabricating dedicated end-effectors, the curved shapes may be emulated in some ways, for example, cupping the fingers of a multi-fingered end-effector.

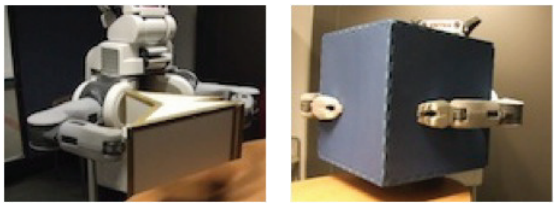


Figure 8.1: Two whole-arm grasps by the PR2.

Section 8.1 presents our algorithm for whole-arm grasping. Section 8.2 discusses the implementation of whole-arm grasping on a modular robot system. Section 8.3 presents a set of experiments.

8.1 Approach and Algorithm

Our approach to whole-arm grasping is composed of two phases: *preshaping* and *squeezing*. In the preshaping phase, a robot cages an object with its two curved end-effectors as discussed in Chapter 5. In the squeezing phase, the robot performs a squeezing motion for the end-effectors. During the squeezing motion, not only the end-effectors but also other links can be made contact the object without adversely affecting the stability of the object if we assume that the robot is position controlled without compliance, in addition to the assumption of frictionless, rigid, unilateral contact:

Corollary 2. *Suppose that the end-effectors of a robot, which is position controlled without compliance, are caging an object, as discussed in Chapter 5. A grasping cage is obtained if the robot is controlled such that the end-effectors are performing a*

squeezing motion until contact, which is assumed to be frictionless, rigid, unilateral, is established.

Proof. The same argument as the proof of Theorem 3 can also be applied here by regarding (1) $\mathcal{M}$ as the configuration space of the robot itself and (2) δ, the distance between the two end-effectors, as the grasping function again. □

The corollary shows that as long as the end-effectors are squeezing, the final state is guaranteed to be a stable equilibrium grasp that can be composed of contacts from the whole body of the robot. Our approach can facilitate planning and control for grasping: in the preshaping phase, the robot can aim at any of the cages, whose collection is not a set of measure zero in the configuration space; the squeezing phase can be performed in a blind manner, only by position control, without direct feedback of the object pose. In fact, the two-phase approach has some similarities with multi-fingered grasping (Miller et al. 2003): approaching an object followed by "closing" the hand.

Our pseudocode is presented in Algorithm 2. The algorithm takes as input an initial configuration of the robot $\mathbf{c}_i \in \mathcal{M}_R$, where $\mathcal{M}_R$ is the configuration space of the robot; it returns a reference trajectory for the robot to follow, $\gamma(s) : [0, 1] \rightarrow \mathcal{M}_R$, where s is a non-dimensional parameter increasing with time. The following paragraphs elaborate each line of the algorithm.

Line 1: We first construct $\mathbf{c}_p, \mathbf{c}_s \in \mathcal{M}_R$ that are supposed to describe configurations at which preshaping and squeezing should aim, respectively (Figure 8.2).

Algorithm 2 MOTION PLANNING FOR WHOLE-ARM GRASPING

Input: Robot's initial configuration, $\mathbf{c}_i \in \mathcal{M}_R$

Output: Reference trajectory for the robot, $\gamma(s) : [0, 1] \to \mathcal{M}_R$

1: Construct two configurations: $\mathbf{c}_p \in \mathcal{M}_R$ for preshaping, $\mathbf{c}_s \in \mathcal{M}_R$ for squeezing.
2: Plan a trajectory γ_{ip} from $\mathbf{c}_i$ to $\mathbf{c}_p$ for the preshaping.
3: Plan a trajectory γ_{ps} from $\mathbf{c}_p$ to $\mathbf{c}_s$ for the squeezing (possibly in parallel with Line 2).
4: Concatenate γ_{ip} and γ_{ps} into γ, the resultant trajectory from $\mathbf{c}_i$ to $\mathbf{c}_s$ via $\mathbf{c}_p$.

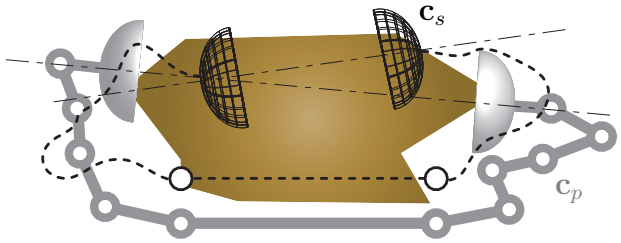


Figure 8.2: At the configuration $\mathbf{c}_p$ (in grey), the curved end-effectors of the arm-chain are caging the object. The wireframe shows the configurations of the end-effectors at $\mathbf{c}_s$, after the squeezing motion.

They can thus be interpreted as desirable waypoints for whole-arm grasping. $\mathbf{c}_p$ is constructed such that the two end-effectors cage the object in a kinematically feasible manner. $\mathbf{c}_s$ is constructed such that the robot deliberately intersects the object. These tasks are essentially inverse kinematics problems.

Line 2: We plan for a trajectory from $\mathbf{c}_i$ to $\mathbf{c}_p$. During the motion, we do not want the robot to interact with the object; thus any collisions should be avoided. This can be considered as an ordinary path planning problem where off-the-shelf algorithms are available. Ultimately, some manipulations such as quasi-static pushing (Mason 2001) may be needed to reach $\mathbf{c}_p$.

Line 3: We now plan for a trajectory from $\mathbf{c}_p$ to $\mathbf{c}_s$ realizing a squeezing motion. The geometry of the object can be ignored; however, the robot should be treated

as a closed kinematic chain in the sense that the two end-effectors are only allowed to approach to each other by relative translation. This is generally a hard problem due to the kinematic closure, but can be solved efficiently for planar chains as will be discussed in Section 8.2.

It is sufficient to control the robot to follow the resultant trajectory in a quasi-static manner, by considering that only does the relative configuration of the object and robot matter in caging and squeezing. In fact, the motion of the robot is necessarily interrupted on the way because it is planned to "collide" with the object. Under the assumption of frictionless, rigid, unilateral contact, the configuration where the robot, which is position controlled without compliance, stops moving is an acceptable grasp that can realize a caged, equilibrium grasp by Corollary 2.

Remark : Algorithm 2 can also be applied to grasping with frictional, compliant contact as explained in the following paragraphs.

First, with friction, we can actually have more candidates for an acceptable grasp. When friction is present, the robot might get stuck on the way during a squeezing motion because nonzero friction can cause *jamming* and *wedging* (Mason 2001). However, both phenomena imply force-closure, which in turn implies involved wrenches are in equilibrium. Thus a jammed or wedged configuration can also be an acceptable grasp. In conclusion, friction helps improve the stability of a grasp, as also discussed by Rimon and Burdick (1998a).

Second, nonrigid objects can also be grasped stably with a squeezing motion. If

we assume a rigid robot moving with a stiff position control servo loop, a nonrigid object will be deformed during a squeezing motion by contact forces exerted by the robot. As long as the caging conditions are satisfied with the deformed geometry, the object will not be lost during the squeezing motion. Howard and Kumar (1996) presented criteria to determine the stability of grasps based on local curvature properties and applied forces. Our grasps have an additional advantage that they are also cages; therefore, information on the relative configuration between the robot and object is sufficient to investigate the stability of a squeezing motion. Previous works address the stability of grasped objects in some special cases. For example, if an object is immobilized, the object remains locally dynamically stable under Gesley and Hertzian stiffness models (Rimon and Burdick 1998b). A jammed or wedged state implies force-closure, which can also be made stable (Nguyen 1989).

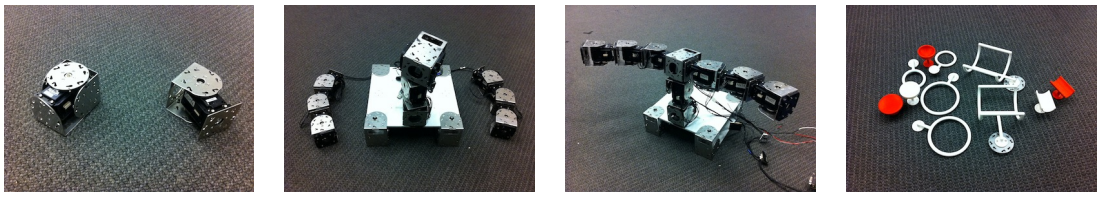
The stability of a nonrigid object under a squeezing motion can be verified by online or offline computation. For online computation, we need to keep track of points of interest (for example, the vertices P and Q in Figure 5.1a) by, for example, visual deformation servoing (Navarro-Alarcon et al. 2014), and verify the corresponding caging conditions. For offline catalog work, we need to consider how to map the deformation of a wide variety of nonrigid objects by, for example, measuring contact forces or joint efforts. A squeezing motion should terminate as soon as the stability of the object is jeopardized.

8.2 Implementing Whole-Arm Grasping

We here present hardware/software that implements our modular approach to whole-arm grasping.

8.2.1 Hardware

Our robot is assembled with CKbot modules[5], our chain style modular robot system. Each CKbot module can be used as an one degree of freedom swivel or elbow joint (Figure 8.3a). Figure 8.3b shows three subassemblies: one spine and two (left and right) arms. The spine, composed of two swivel joints and one elbow joint, provides all the three rotational degrees of freedom by realizing z-y-z Euler angles. The arms are planar manipulators composed only of elbow joints assembled such that their axes of rotation are parallel. Although Figure 8.3b shows arms composed of three modules, the modular architecture allows us to assemble more links easily.



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

Figure 8.3: (a) Two types of CKbot modules providing one rotational degree of freedom. The left one (swivel joint) provides continuous rotation; the right one (elbow joint) is limited to 180 degrees rotation. (b) Two 3-d.o.f. planar arms and a 3-d.o.f. spine between them. (c) A finished two-armed robot. (d) 3D printed end-effectors that can be docked to the robot.

[5] `http://www.modlabupenn.org/ckbot`

The two arms are attached to the top of the spine such that the whole armchain is again a planar open kinematic chain. The planar architecture suffices to realize our immobilizing grasps, discussed in Chapter 4, for the following reason. For each point, line, or planar contact, consider the net wrench that is the combination of all contact wrenches exerted at the contact. The three (or two) net wrenches in any of our immobilizing grasps should be coplanar (from a *planar pencil*); otherwise, it is impossible to even guarantee equilibrium, which is necessary for immobility. Our cages and squeezing motions discussed in Chapter 5 can also be realized by the planar architecture because the aligned effectors are only required to approach to each other. In Figure 8.3c, the finished two-armed robot is shown. Figure 8.3d shows 3D printed curved end-effectors compatible with CKbot. The modular architecture allows us to quickly adapt to the sizes and shapes of a wide variety of objects by attaching more arm links or exchanging end-effectors.

8.2.2 Software

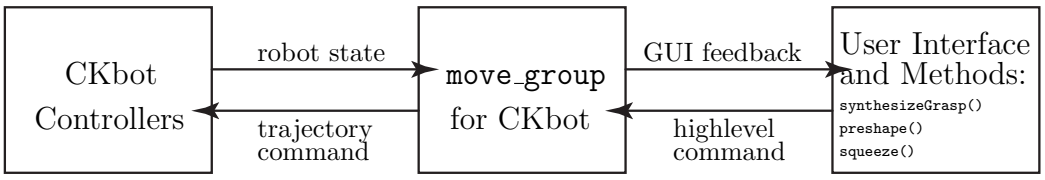


Figure 8.4: Software architecture.

Our software implementing whole-arm grasping for the hardware platform is organized as ROS[6] packages written in C++ and Python. Figure 8.4 shows the

architecture of our software. The `move_group` node in the center, organized as MoveIt![7] packages, works as an integrator between the hardware and a human user. The software provides a user with high-level methods for grasp synthesis (Algorithm 1) and motion planning for preshaping and squeezing (Algorithm 2); the robot is then position controlled to follow the resultant reference trajectory. In the following paragraphs, we further explain how Algorithm 2 is implemented for our robot with the planar armchain.

First, the inverse kinematics solver for Line 1 of Algorithm 2 is based on the well-studied closed-form solutions of the inverse kinematics of a planar 3R manipulator (planar manipulator with three revolute joints); see Figure 8.5a. For an arm with three CKbot modules (exactly a planar 3R manipulator), there can be two solutions commonly known as the "elbow-up" and "elbow-down" configurations. The method can also be applied to solving the inverse kinematics of an arm with n CKbot modules, where $n > 3$, by converting the complete problem into appropriate subproblems that the solver can address.

Second, the squeezing in Line 3 of Algorithm 2 is based on energy-based motion planning. An algorithm presented by Iben et al. (2009) generates an interpolation sequence without any self-intersections between two simple polygons. The algorithm also allows us to monotonically change link lengths; this allows us to implement squeezing motions. The resultant motion basically reconfigures the two polygons

[6] `http://www.ros.org`

[7] `http://moveit.ros.org`

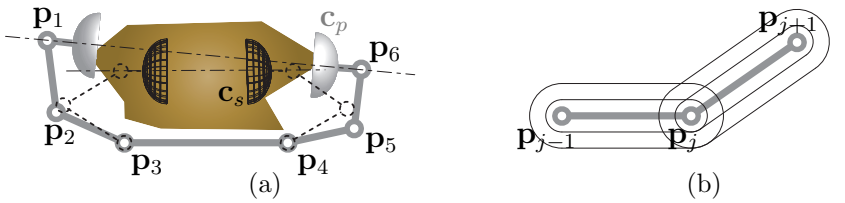


Figure 8.5: (a) The planar armchain is composed of two planar 3R manipulators connected to the base (the longest link). At the configuration $\mathbf{c}_p$ (in grey), the end-effectors are caging the object. $\mathbf{c}_s$ shows a target configuration which a squeezing motion can aim at. (b) Around each link $\overline{\mathbf{p}_x\mathbf{p}_y}$ of the planar 2R manipulator, two level sets of $d(\mathbf{x}, \overline{\mathbf{p}_x\mathbf{p}_y})$ are shown. Such a level set allows us to model the actual collision hull of a link that may not be a line segment.

"towards each other" according to a metric defined between a pair of polygons, for example, the ℓ^2-norm on the vector of vertex coordinates. Whenever the direct reconfiguration increases the value of an energy function such as

$$\mathcal{E} = \sum \frac{1}{d(\mathbf{p}_i, \overline{\mathbf{p}_j\mathbf{p}_{j+1}})^2} \tag{8.2.1}$$

where the denominator can be the squared shortest distance between joint $\mathbf{p}_i$ and link $\overline{\mathbf{p}_j\mathbf{p}_{j+1}}$ connecting the joints $\mathbf{p}_j$ and $\mathbf{p}_{j+1}$ $(i \neq j, j+1)$ (see Figure 8.5a for the notation), we follow the downward gradient of $\mathcal{E}$ to avoid collisions. This energy-based planning can be performed efficiently even for a hyper-redundant arm.

Remark : The energy-based motion planning algorithm explained in the previous paragraph is applicable to line segment links without joint limits, that is, $\theta_i \in [-\pi, \pi]$ where θ_i is the angle of joint $\mathbf{p}_i$ ($\theta_i = 0$ when the two incident links are collinear). We further discuss how to adapt the algorithm so as to address joint limits and link shapes that are not line segments.

First, given narrower joint ranges ($\theta_i \in [-\ell_i, u_i]$ where $0 < \ell_i, u_i < \pi$ for each θ_i),

suppose that at a certain instant there are one or more joint angles close to their limit, that is, $\theta_j \in [-\ell_j, -\ell_j + \varepsilon_1]$ or $\theta_j \in [u_j - \varepsilon_2, u_j]$ for some $\varepsilon_1, \varepsilon_2 > 0$, and j. If the armchain is described as a concave polygon at that instant, we propose to apply an *expansive motion* (Connelly et al. 2003) to unfolding all joints such that each θ_j can return to the "safe" range, that is, $\theta_j \in [-\ell_j + \varepsilon_1, 0]$ or $\theta_j \in [0, u_j - \varepsilon_2]$. During an expansive motion for a closed chain, every joint is monotonically unfolded ($|\theta_i|$ is decreasing for all i) until the chain is convexified. Such a motion always exists as long as the chain is described as a simple, concave polygon and is computed by convex optimization (Connelly et al. 2003). In case the armchain is described as a convex polygon at the instant, there also exists such an angle-monotone motion to bring the joint angles back to the safe range (Aichholzer et al. 2001).

Second, in order to address nonzero link volume, we use $(d(\mathbf{p}_i, \overline{\mathbf{p}_j\mathbf{p}_{j+1}}) - \delta_j)^2$ as the denominator of each term of $\mathcal{E}$ where δ_j is determined for each link $\overline{\mathbf{p}_j\mathbf{p}_{j+1}}$ such that the collection of $\mathbf{x}$'s on the level set $d(\mathbf{x}, \overline{\mathbf{p}_j\mathbf{p}_{j+1}}) - \delta_j = 0$ can address the collision hull of the link (Figure 8.5b). Note that two adjacent links overlap each other around the joint connecting them; in practice, this issue can be addressed by using, for example, offset links.

8.3 Experiments

A set of experiments were run to evaluate our hardware/software implementation and its performance on the task of grasping objects. We first see how accurately

our robot can position its end-effectors; we then proceed into grasping objects.

8.3.1 End-Effector Positioning

Positioning end-effectors is critical to successful grasping; thus, we first evaluated the positioning accuracy of our hardware/software implementation. We assembled a robot with two arms; each arm is composed of three link modules as already shown in Figure 8.3c. We set up three target configurations for the robot such that (1) the tips of the right and left arms are level at the same height (Experiment 1), (2) the tip of the right arm is higher than that of the left arm (Experiment 2), and (3) the tip of the right arm is lower than that of the left arm (Experiment 3). For each target configuration, a total of 25 trials were conducted and we measured the actual, final positions of the tip of the right arm using the Vicon motion capture system[8]. Figure 8.6 illustrates the results; Table 8.1 enumerates the data points measured. The average positioning error of the 75 trials was 3.73 centimeters.

In computer simulations, there were no errors in positioning end-effectors; thus, the errors in the real experiments are mostly due to hardware such as tolerance, mechanical play, motor backlash, or compliance. Obviously, the positioning errors are not negligible by considering that each arm is appriximately 30 centimeters long. The arm length of the robot can be compared with the arm length of infants (the mean of the upper arm length of infants aged 3-5 months is 12.8 centimeters

[8] `http://www.vicon.com`

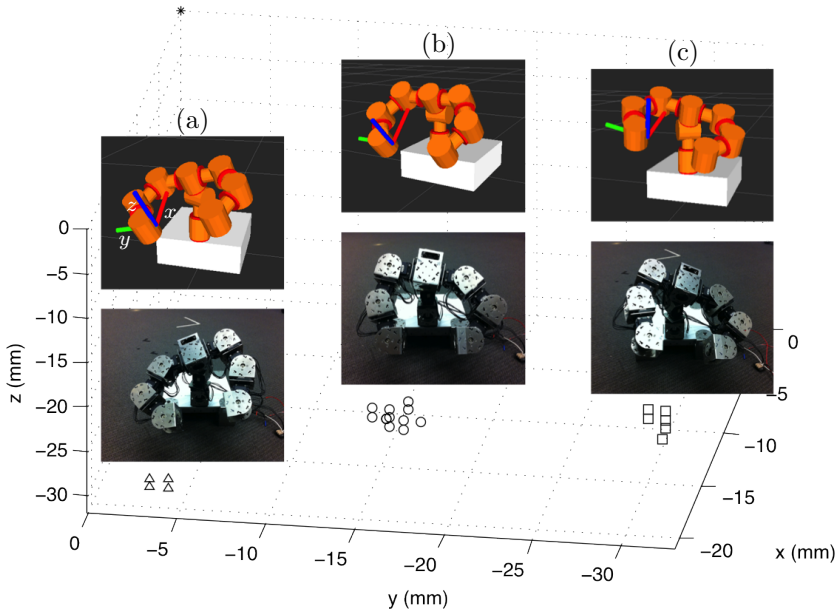


Figure 8.6: The figure illustrates the results of Experiments 1, 2, and 3. In (a), (b), and (c), the simulated robot in the upper panel shows the target configuration for Experiments 3, 1, and 2, respectively; the real robot was controlled to the targets as shown in the lower panels. The '△', '◯', and '□' marks represent data points showing the actual, final positions of the tip of the right arm in Experiments 3, 1, and 2, respectively, with respect to the reference frame attached at the tip of the right arm of the simulated robots (the red, green, and blue axes are the x-, y-, and z-axis of the frame). In principle, the data points were expected to coincide with the origin of the frame (see the '$*$' mark at the top of the graph).

(McDowell et al. 2008)); for comparison, it has been known that neonate infants can position their hands within 1.5 centimeters from objects, which they touch and occasionally grasp (Bower 1970).

8.3.2 Whole-Arm Grasping

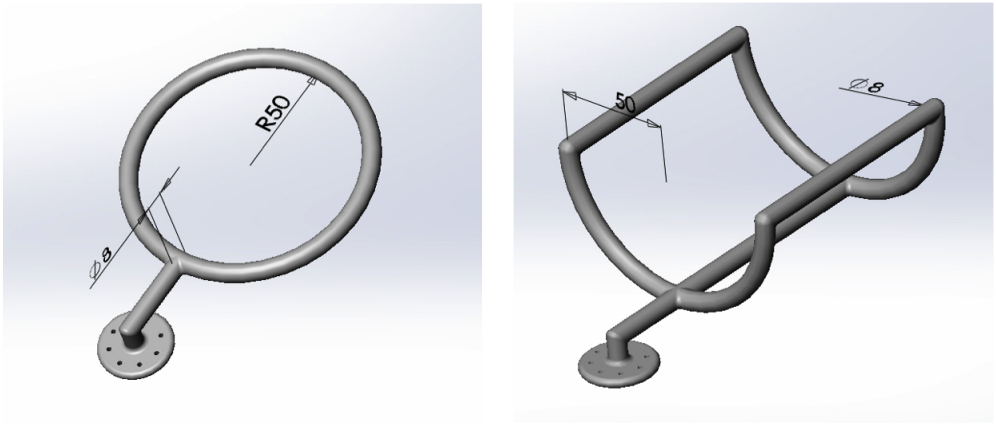
The positioning experiments suggest the appropriate sizes of end-effectors for successful grasping. Figure 8.7 shows the CAD models of our end-effectors fabricated

x-, y-, z-error (mm)	frequency	x-, y-, z-error (mm)	frequency	x-, y-, z-error (mm)	frequency
(-17, -16, -24)	6	(-19, -32, -20)	19	(-19, -3, -31)	19
(-17, -15, -24)	3	(-19, -32, -21)	2	(-19, -4, -30)	3
(-18, -17, -23)	3	(-19, -31, -19)	1	(-19, -4, -31)	2
(-17, -16, -23)	3	(-19, -32, -19)	1	(-19, -3, -30)	1
(-17, -15, -23)	2	(-19, -31, -20)	1		
(-17, -16, -25)	2	(-20, -32, -21)	1		
(-18, -16, -23)	2				
(-17, -17, -22)	1				
(-18, -18, -23)	1				
(-17, -17, -23)	1				
(-18, -17, -24)	1				

Table 8.1: The results of Experiments 1, 2, and 3 are summarized in the first, second, and third column, respectively. Each entry of the column shows a triple of numbers that represent x-, y-, and z-directional positioning errors, with respect to the reference frame shown in Figure 8.6, measured in millimeters with the number of times it appeared.

for grasping experiments. The dimensions of the effectors were determined to address the x- and z-directional positioning errors (recall Figure 8.6 for the axes and Table 8.1 for the error data). For example, for the torus-shaped effector in Figure 8.7a, the inner radius of the torus was determined to be 46 millimeters, larger than the maximum positioning error on the xz-plane from Experiments 1, 2, and 3; for the cylindrical effector in Figure 8.7b, the inner radius of the cylinder was also determined to be 46 millimeters for the same reason. The end-effectors are supposed to be squeezed along the y-axis; thus, the y-directional errors can be addressed by our approach that takes advantage of squeezing.

We ran three sets of experiments to evaluate the capability of our system: grasping an object using its vertex-vertex pair (Experiment 4), vertex-face pair (Experiment 5), and edge-edge pair (Experiment 6). In each set of experiment, a total of 25



(a) (b)

Figure 8.7: (a) A torus-shaped end-effector. (b) A cylindrical end-effector; in the model, material usage was minimized to reduce weight and cost. The units are in millimeters.

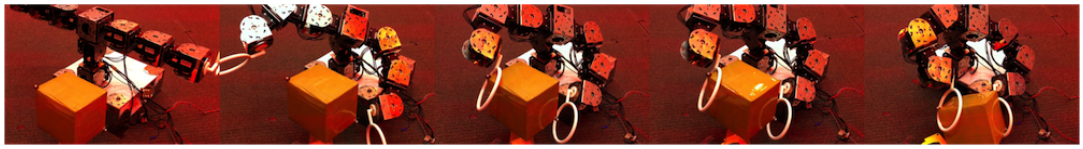
trials were conducted and we verified if the robot could grasp an object, perceived by the Vicon system, via preshaping and squeezing without any position calibration. The results are summarized in Table 8.2. The snapshots in Figure 8.8 show

grasp type	successes/trials
Experiment 4: vertex-vertex grasping	22/25
Experiment 5: vertex-face grasping	23/25
Experiment 6: edge-edge grasping	25/25

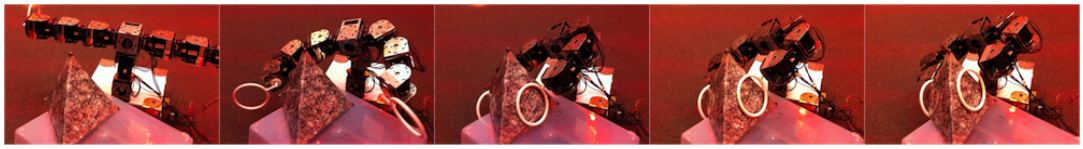
Table 8.2: The results of Experiments 4, 5, and 6.

the robot grasping objects via preshaping and squeezing. The robot was controlled until joint torque saturation occurred during the squeezing. In the failed trials, the two arms were moving in an asynchronized manner (possibly due to an unexpected communication error) or the end-effectors were dynamically interacting with the object (or the support under the object).

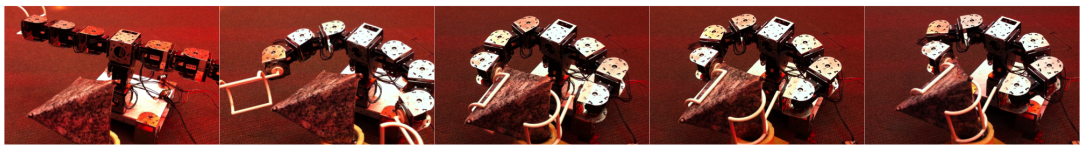
Finally, Table 8.3 shows the time frame to perform preshaping and squeezing,



(a)



(b)



(c)

Figure 8.8: (a) The robot is grasping the box with the two torus-shaped end-effectors that can cage and grasp the vertex-vertex pair. (b) The robot is grasping the tetrahedral object also with the two torus-shaped end-effectors that can cage and grasp the vertex-face pair; one of the torus-shaped effectors was used to contact the face. (c) The robot is grasping the tetrahedral object with the two cylindrical end-effectors that can cage and grasp the edge-edge pair.

for 25 trials. In the table, the planning times show how long it takes to run the respective software components (`preshape()` or `squeeze()`, recall Figure 8.4) on a 2.53GHz/4GB machine. The executing times show how long it takes for the robot to execute the computed plans. It can be seen that if preshaping and squeezing are planned in a parallel manner, as mentioned in Algorithm 2, they may be executed in a seamless manner without interruption because the time to preshape can be compared with the time to plan for squeezing. The methods `preshape()` and `squeeze()` were implemented in Python; they can be made run faster by using low-level programming languages.

Preshaping				Squeezing	
planning time		executing time		planning time	
min	max	min	max	min	max
0.21s	0.41s	4.7s	6.4s	1.72s	6.41s

Table 8.3: The time frame to perform the two phases of whole-arm grasping.

Part III

Assembling with Modular Robots

Chapter 9

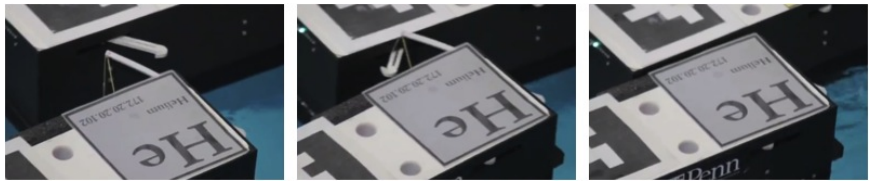
Approach to Assembling Planar Structures with Rectangular Modules

Part III of this thesis is concerned with assembly planning: in which order parts, that is, robots, can be assembled into a target structure without getting stuck. In this chapter, we discuss our approach to assembly planning for constructing planar target structures with rectangular modular units. Section 9.1 introduces our hardware framework. Section 9.2 formally defines target structures we will aim at. Section 9.3 describes our approach to assembling a given target structure. The following two chapters, Chapters 10 and 11, present two algorithms elaborating the approach.

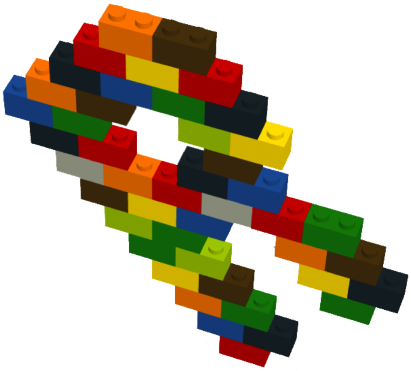
9.1 Hardware Framework

In this section, we briefly introduce our hardware framework that motivated the development of the algorithms to be presented. For full details on the hardware design, refer to O'Hara et al. (2014).

(a)



(b)



(c)

Figure 9.1: (a) The left panel shows two robots floating on water in a swimming pool. The right panel shows the bottom of the robot having four waterjet nozzles at the four corners. (b) Robot "He" is docking to another robot (from left to right). (c) Lego bricks can only dock on top of (or under) other bricks, in the same way that our robots dock.

9.1.1 Shape and Locomotion

Our *robots* are mobile robotic boats having the same rectangular footprint on water. The left panel of Figure 9.1a shows two robots floating on water. The rectangular shape can fill the plane without gaps; considering the shape also comes from the idea of using 20ft-long ISO shipping containers, whose footprint on water is rectangular, to build large floating platforms[9] that can be used for the formation of offshore bases for disaster response, ad hoc landing strips, or refueling depots.

The right panel of Figure 9.1a shows the bottom face of a robot that is outfitted with a set of four waterjet nozzles that enable holonomic locomotion on the water surface. In other words, the rectangle representing the footprint of a robot is capable of holonomic motion on the plane: the three (two translational and one rotational) degrees of freedom on the plane can individually be controlled.

9.1.2 Docking Capability

Each robot is outfitted with a docking mechanism; it imposes a constraint that robots are assembled in a regular pattern that locally looks like a common brick wall as shown in Figure 1.2b. The pattern lends itself to docking mechanisms that require modules to approach "broad side," which in turn allows for more robust docking. A full-scale prototype with ISO shipping containers developed by *QinetiQ North America, Inc.* has established that this arrangement maximizes capture

[9]DARPA Tactically Expandable Maritime Platform

probability.[10] In addition, the pattern is structurally sound (National Concrete Masonry Association 2004).

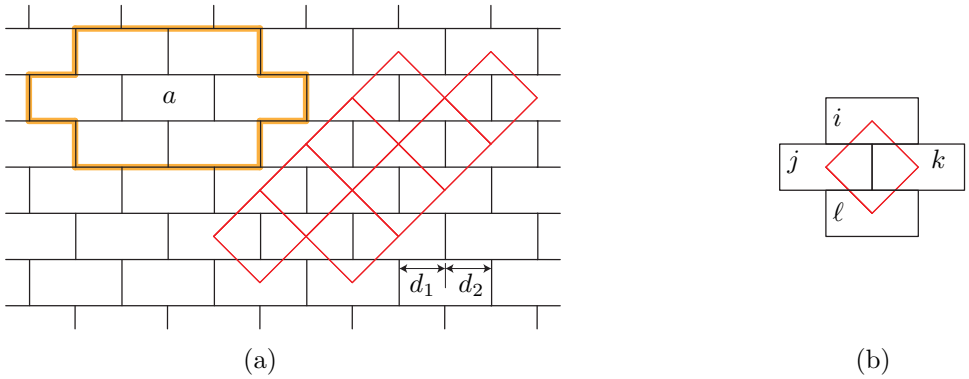


Figure 9.2: (a) The plane is tessellated with congruent rectangles to form the common brick wall pattern, where $d_1 = d_2$. Each of the red rhombi represents the lattice unit of the pattern. In the assembled structure of robots, each robot has the potential of having six adjacent robots in its *one-hop neighborhood* (imagine a robot occupying the site a and its potential six neighbors inside the orange polygon). (b) Each of the lattice units involves four rectangular areas, denoted as i, j, k, and ℓ here, each of which can accommodate a robot.

Specifically, two robots dock using a male to female connection mechanism: a hook coming from one robot engages with a cable loop from the other robot. Two robots can actually dock to each other only along their long sides (Figure 9.1b); they cannot dock along their short sides. In fact, a robot does not need to be mechanically connected to all of the neighboring robots in order to form the brick wall pattern. Consider a plane tessellated with congruent rectangles forming the pattern as shown in Figure 9.2a. According to Schattschneider (1978), we can find the *lattice unit*, the smallest area of the pattern repeated in translation as also shown in Figure 9.2a. Each of the lattice units involves four rectangular areas

[10]Communication with *QinetiQ North America, Inc.*

(Figure 9.2b). In order for the four robots occupying the areas to form a rigid structure, it is sufficient to mechanically connect them only along their long sides. In other words, even if robots in areas j and k in Figure 9.2b are not mechanically connected to each other, the four robots can form a rigid structure as long as robot pairs occupying i and j, i and k, ℓ and j, and ℓ and k are mechanically connected to each other. Because the lattice unit is repeated and two adjacent lattice units share two rectangles, it is possible for robots to even rigidly fill the plane with such a docking mechanism. Lego[11] bricks also dock in the same manner (Figure 9.1c).

Remark : Note that we cannot exactly construct some structures such as a single row of robots connected end-to-end along their short sides, with the docking mechanism. But the arrangement can simplify mechanical design guaranteeing a space-filling capability.

9.2 Target Structure

For planning purposes, our robotic boats are just regarded as congruent rectangles moving on the plane. The size and aspect ratio of the rectangle do not matter: our approach does not need to be parametrized by them. A *target structure* that we want to assemble can be visualized as the collection of rectangular *sites* that locally looks like the common brick wall pattern without a gap between two adjacent sites

[11] `http://www.lego.com`

(Figure 9.3a); each site is the footprint of a single robot in the finally assembled structure. Target structures are assumed to be *connected* and free from *narrow corridors*; the following paragraphs elaborate the two terms.

Given a target structure, we can construct an undirected graph $C = (V, E)$ that represents the mechanical connectivity among robots occupying the member sites (Figure 9.3b). V is the collection of the sites of the target structure and E consists of unordered pairs of two sites in contact that accommodate two robots mechanically docked. C is a *plane graph*, which is drawn in such a way that no two edges meet in a point other than a common end; the maximum degree of C is 4. If C is connected, we say that the target structure is connected. If there are multiple connected components, they can be assembled individually.

We also make use of C to define a narrow corridor; see Figure 9.3c. Here, C is assumed to be embedded on the corresponding target structure such that the vertices are the centroids of the sites and the edges are line segments. Consider a rhombus that can contain nine lattice units (recall Figure 9.2): the side length of the rhombus is three times as large as the edge of the lattice unit. For each face of C, let the rhombus translate along the frontier of the face without losing contact with the frontier and intersecting the exterior of the frontier. If the rhombus can reach every point on the frontier, we say that the face is free from narrow corridors. Note that the smallest faces (such as the one colored blue in Figure 9.3c) are exempt from the test because the face is in fact not an empty space. The structure of

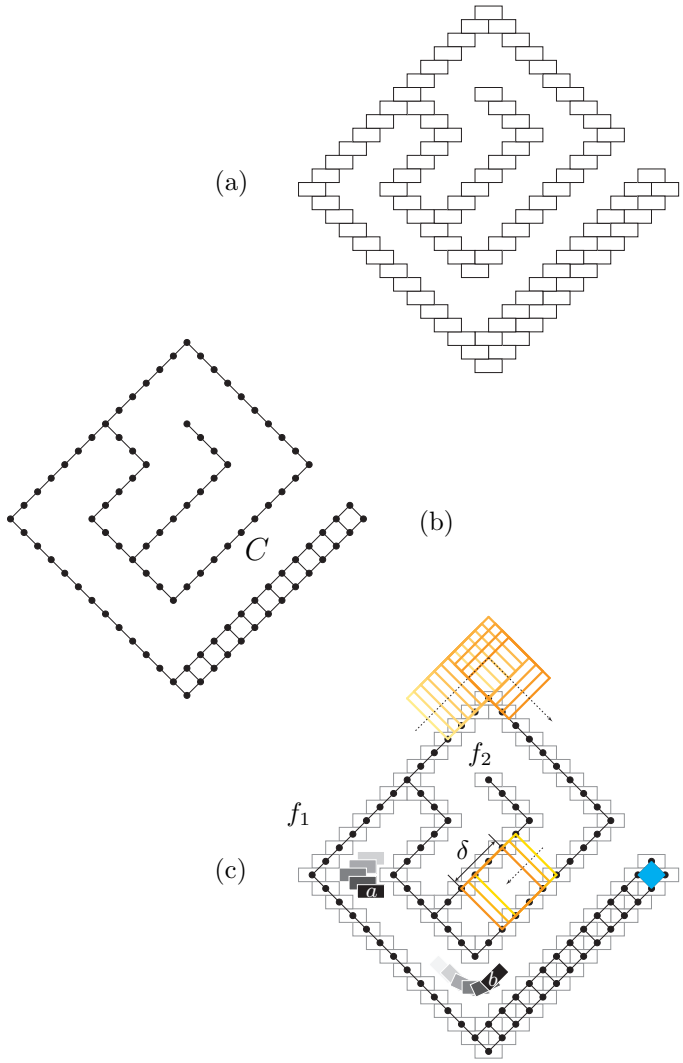


Figure 9.3: (a) A target structure. Each rectangle represents a site, to be occupied by a robot. (b) Graph C represents the mechanical connectivity of the finally assembled structure. (c) The faces of C, f_1 and f_2 (except for the smallest faces), are free from narrow corridors.

Figure 9.3a is thus free from narrow corridors; a robot (see a in Figure 9.3c) can navigate the corridors at least by omnidirectional translation, which is enabled by the holonomic locomotion. We may need to consider a larger rhombus if wider corridors are needed, for example, in case a robot moves in a nonholonomic manner like robot b in Figure 9.3c, with a large turning radius. One way to admit corridors that are narrower than the ones discussed here is to make robots smaller.

9.3 Approach

In order to assemble a target structure, one may consider incorporating multiple subassemblies after preparing them independently; however, we are interested in growing the structure as one connected component by adding one or multiple robots individually to the structure. This allows us to sidestep possible hardware and software complications in controlling and docking large subassemblies.

During assembly, we are particularly interested in guaranteeing easy accessibility. We want to avoid the scenarios shown in Figures 9.4a and 9.4b where the sites to be occupied can only be accessed by passing through a gap only as large as a side of a robot between two physical robots already assembled in the structure, where an incoming robot can in practice get stuck due to *wedging* or *jamming* (Mason 2001). Thus, such scenarios may considerably slow down assembly possibly with safety problems, if not impossible. The issues illustrated in Figures 9.4a and 9.4b are obviously local scale problems; in addition, the issues can also occur at nonlocal

scales spatially and temporally. For example, in Figure 9.4c, the robot has to pass through the gap with insufficient clearance between a and b in order to occupy a site not adjacent to a and b (spatial nonlocality); in Figure 9.4d, if we want to fill the seven unoccupied sites, at least the last robot to fill the row has to pass through a gap as large as its side (temporal nonlocality). Figures 9.4c and 9.4d thus illustrate some examples of intermediate configurations that should also be avoided during assembly.

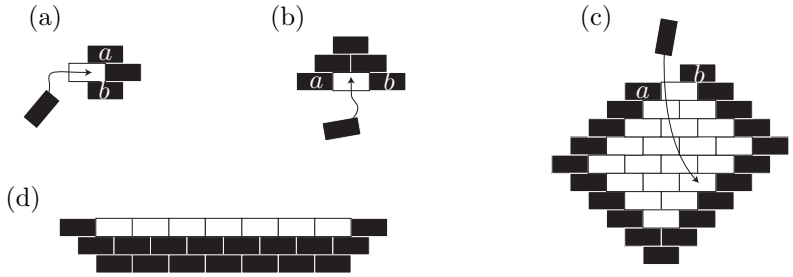


Figure 9.4: (a), (b) The gap between robots a and b can essentially block the incoming robot. (c) The gap between a and b can affect assembly that happens at a distance spatially. (d) If we want robots to occupy the seven open sites, at least the last robot has to pass through a gap just as large as its side.

Our first algorithm to be presented in Chapter 10 is applied to target structures without internal holes; for example, see the shape shown in Figure 10.1. The output of the algorithm is a directed graph representing an assembly plan allowing us to assemble a given target structure by stacking a row of robots on top of another row already assembled, in a parallel manner. This idea can be compared to raster scanning, a technique for generating an image by line-by-line construction.

Our second algorithm to be presented in Chapter 11 can be applied to target

structures with internal holes; for example, see the shape shown in Figure 9.3. We here apply "disassembly planning": given a target structure that is assumed to be occupied with robots, (1) we search for a robot that can be disassembled without suffering from the accessibility issues illustrated in Figure 9.4; (2) then remove the robot and update the structure; (3) and repeat. We get an assembly sequence by inverting the disassembly sequence; we add one robot at a time and grow the structure as one connected component.

Chapter 10

Algorithm for Parallel Assembly

The algorithm we present in this chapter can be applied to assembling target structures without internal holes. The resultant plan can be executed in a parallel manner. The algorithm enables correct assembly without the accessibility issues illustrated in Figure 9.4. Sections 10.1 and 10.2 present the algorithm and instructions on applying the algorithm. Section 10.3 discusses the correctness of the algorithm.

10.1 Algorithm

Our pseudocode is presented in Algorithm 3. We take advantage of many concepts from graph theory (Cormen et al. 2001, Diestel 2000). The algorithm takes as input an array containing a sequence of the sites of a target structure, denoted as T; see Figure 10.1 for the example target structure we use to explain the progress of the

algorithm. The order of the sequence does not matter; each site is represented by the coordinate of its centroid, which is simply referred to as the coordinate of the site. The algorithm returns a graph G_A, a directed acyclic graph constructed on the member sites. A directed acyclic graph is widely used for modeling precedences among tasks. Algorithm 3 can be made run in $O(m)$ time, where m is the number of the member sites, as will be explained in the following paragraphs elaborating each line of the algorithm.

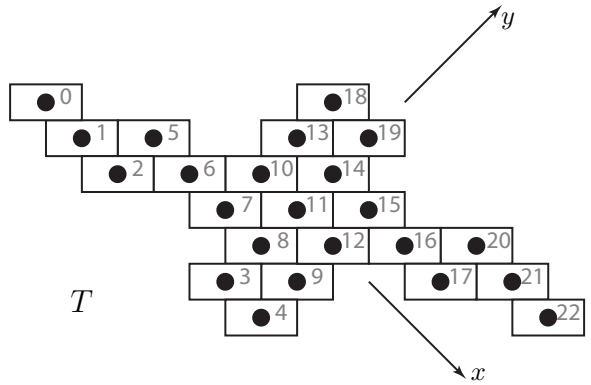


Figure 10.1: A target structure T, which can be represented as a sequence of the coordinates of the centroids (shown as the black dots) of the member sites, with respect to the xy-frame that is oriented as the grid of the lattice units (Figure 9.2). Although the order of the sequence does not matter, the site numbers are labeled to help explain the progress of the algorithm.

Line 1: We begin with declaring a graph G_A constructed on the sites of T, without any edges at this moment.

Line 2: We decompose T into the collections of consecutive sites, parallel to the x-axis of the reference frame shown in Figure 10.1; each collection is referred to as a *cell*. We then construct an undirected graph C' that represents the mechanical

Algorithm 3 ASSEMBLY-PLANNING(T)

Input: Target structure T: a sequence of sites to be occupied, $T = \langle s_1, s_2, \cdots, s_m \rangle$.
Output: G_A: a directed acyclic graph on the sites of T

1: Let $G_A = (T, \emptyset)$.
2: Decompose T into cells and construct C'.
3: Designate one seed for each cell; turn C' into a rooted tree.
4: **for** each vertex of G_A, s_i, **do**
5: Construct directed edges that leave s_i and enter its neighbors that belong to the parent cell or are more proximal to the seed in the same cell.
6: **end for**
7: Return G_A.
8: (option) Perform a topological sort of G_A.

connectivity among the cells: two cells in contact are connected in C' if they are mechanically connected in the finally assembled structure. Figure 10.2a illustrates the procedure. All the sites of a cell have the same y-coordinate that is referred to as the y-coordinate of the cell. C' is a minor of C, representing the mechanical connectivity of the finally assembled structure (recall Figure 9.3). The cell decomposition can be performed in $O(m)$ time. C' can also be constructed in $O(m)$ time: for each site of a cell, there can be at most two other cells mechanically connected to the cell by the site. Since T does not have any internal holes, C' is a tree; thus, the number of the edges of C' is $O(m)$.

Line 3: For each cell, we designate one site as the seed from which the cell starts to grow. First, we arbitrarily pick any site and denote it as $\mathfrak{s}$; $\mathfrak{s}$ is then the seed of the cell it belongs to. Now C' can be turned into a rooted tree by letting the cell be the root. For the remaining cells that are the non-root vertices of C', we enforce the following rule to designate their seed:

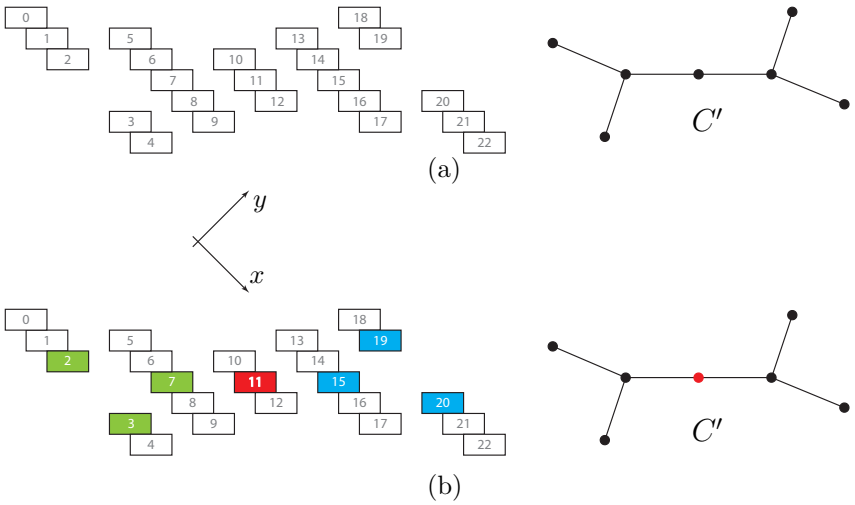


Figure 10.2: (a) The target structure shown in Figure 10.1 has been decomposed into the cells in the left panel. The right panel shows C', representing the mechanical connectivity among the cells. (b) The seeds of the cells are colored red, green, or blue. C' can be turned into a tree rooted at the red vertex, representing the cell to which site 11 belongs.

> Given a cell that is a non-root vertex of C', if its y-coordinate is larger (less) than that of its parent with respect to C', then the site with the largest (least) x-coordinate to accommodate a robot that is to be mechanically docked to a robot in the parent is designated as its seed.

See Figure 10.2b for illustration: (1) site 11 (colored red) was initially picked as $\mathfrak{s}$; (2) the blue (green) sites represent the seeds for those cells whose y-coordinate is larger (less) than that of their parent with respect to C' (according to the rule, for example, site 15 (7) has the largest (least) x-coordinate among the sites to accommodate a robot docked to the parent). A breadth-first search can be used to identify the parent-child relationships of the cells in $O(m)$ time since C' has $O(m)$ edges. Designating the seed following the rule can be done in $O(1)$ time for each

cell. This step can thus be made run in $O(m)$ time.

Lines 4 – 6: We are now in the main loop where the edge set of G_A is populated by applying the following rule:

> For each site of the target structure, denoted as s_i, consider the collection of other member sites belonging to s_i's one-hop neighborhood (recall Figure 9.2a). For each site of the collection, denoted as s_j, construct a directed edge (s_i, s_j) on G_A (1) if s_i and s_j belong to the same cell and s_i is more proximal to the seed than s_j; or (2) if s_i belongs to the parent, with respect to C', of the cell to which s_j belongs.

Figure 10.3a shows the resultant graph on the example shape. The directed edges constructed above are considered as precedences in assembly: a site is occupied only after all of its parents, with respect to G_A, are occupied. This step can be made run in $O(m)$ time: each site has at most six neighboring sites.

Line 7: The algorithm finally returns G_A, which is the assembly plan for the target structure.

Line 8: We can optionally perform topological sorting (Cormen et al. 2001) on G_A such that we can find a linear ordering of sites satisfying the precedences (Figure 10.3b). Topological sorting can be performed in $O(m)$ time here using depth-first search since G_A has $O(m)$ edges. Note that the original source vertex of the depth-first search should be $\mathfrak{s}$, from which the assembly starts.

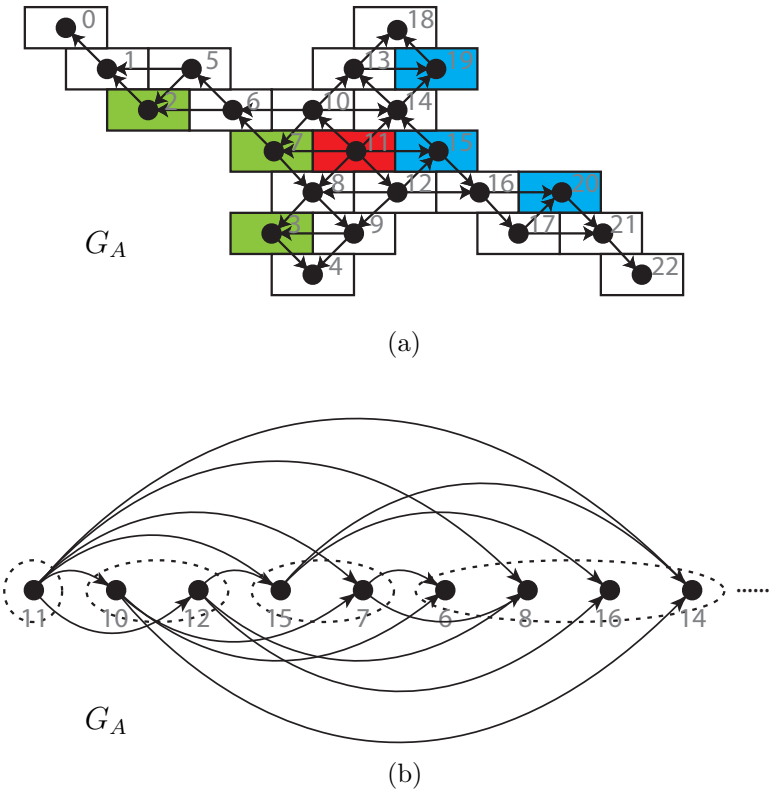


Figure 10.3: (a) G_A returned by Algorithm 3. (b) The same graph as (a), but topologically sorted. The sites grouped in each dotted boundary can be occupied simultaneously once the preceding groups are occupied.

10.2 Instructions on Applying Algorithm 3

We can think of a central commander that coordinates assembly according to G_A: an actual assembly process following G_A begins with occupying $\mathfrak{s}$ (for example, site 11 in Figure 10.3a) and proceeds with the *partial order* established by the directed edges of G_A. The structure grows as a single connected component and multiple robots can dock to the structure in a parallel manner. For example, as can be seen in Figure 10.3b, if sites 11, 10, 12, 15, and 7 are occupied, sites 6, 8, 16, and 14 can be occupied in parallel without any coordination among them because the four

docking events do not depend on one another. The number of cells can be $O(m)$, and each cell can have two growing fronts. It is then possible to get a growth rate of $O(m)$, that is, $O(m)$ robots may be assembled simultaneously. In the worst case where there is only one cell, however, we only get an $O(1)$ growth rate all the time.

Instead of the centralized execution, each assembly event can be done by local-scale decision-making that can take only $O(1)$ time. In G_A, every edge is established between two geometrically adjacent sites and a site has at most three parents. Thus, if each robot (1) has a copy of G_A and (2) is capable of local sensing and communication, it can only take $O(1)$ time for the robot to decide whether to occupy an empty site or not.

Remark : Because G_A does not prescribe a *total order*, that is, a particular, deterministic assembly sequence, it is possible to adaptively make the maximum progress on the rest of the structure in case a docking event is unexpectedly delayed. For example, in Figure 10.3a, even in case the docking event at site 6 is delayed, we can still get a structure composed of robots occupying all the other sites except for sites 0, 1, 2, 5, and 6.

10.3 Analysis

We begin with showing that G_A correctly represents the precedences among the events of occupying sites without cyclic dependency.

Lemma 1. *G_A is a directed acyclic graph.*

Proof. According to Cormen et al. (2001), a directed graph is acyclic if and only if a depth-first search of the graph yields no "back edges," which are those edges connecting a vertex to one of its ancestors in the depth-first tree.

At any vertex s of G_A which has been discovered but is yet to be finished during a depth-first search, the next vertex to visit belongs to 1) the same cell as s or 2) one of the adjacent cells with respect to C', according to Line 5 of Algorithm 3:

1. In a single cell, all edges are directed away from the seed; there cannot exist any back edge connecting two sites in the same cell.

2. Between two adjacent cells, only do edges directed from the parent to the child exist. Once we leave a cell, we cannot return to the cell because C' is a tree. Thus there cannot exist any back edge connecting two sites in two adjacent cells, respectively.

In conclusion, no back edges are yielded in a depth-first search; G_A is thus a directed acyclic graph. □

Since G_A is a directed acyclic graph, it represents well-defined ancestor-descendant relationships among its vertices; in addition, topological sorting can indeed be performed on G_A in Line 8 of Algorithm 3.

Next, we show that following the assembly rules specified in G_A guarantees that the structure grows as a single connected component and the assembly terminates:

there are no sites remaining unoccupied forever.

Lemma 2. *Given a target structure T, the assembly rules specified in G_A lead to every site of T being eventually occupied, while maintaining a connected topology.*

Proof. According to Line 5 of Algorithm 3, (1) a non-seed site of a cell is the descendant of the seed of the cell and (2) the seed of a cell, except for $\mathfrak{s}$, is the descendant of one site of its parent cell. Therefore, every member site of T is the descendant of $\mathfrak{s}$ in terms of G_A; in other words, there does not exist any member site that remains unoccupied forever, by assembling according to G_A. Moreover, every site is physically adjacent to its parents in terms of G_A. Therefore, the structure grows as a single connected structure. □

While Lemma 2 provides important guarantees about the completeness of the resultant assembly plan and the connectedness of the structure being assembled, it does not address the accessibility of a site to be occupied that might be blocked by a physical wall or flanked between two physical robots already docked in the structure (recall Figure 9.4). The following lemma shows that if we follow the assembly rules specified in G_A, there are no such accessibility problems.

Lemma 3. *Following the assembly rules specified in G_A ensures a path for a robot to access any site open for occupancy without passing through a gap that is as large as a side of a robot, formed between two robots docked in the structure.*

Proof. Consider a cell of a given target structure; it can be (1) the root of C', (2)

a cell whose y-coordinate is larger than that of its parent, or (3) a cell whose y-coordinate is less than that of its parent in terms of C'. We show that there are no accessibility problems when assembling cells in each category. Let s denote the site to be occupied and S denote the cell to which s belongs.

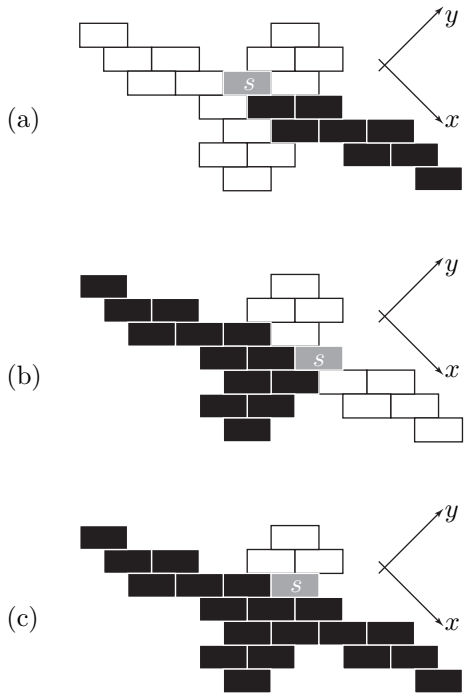


Figure 10.4: Some possible snapshots when we assemble the example shape according to G_A shown in Figure 10.3. Each panel shows the most populated structure that can be obtained without having s occupied. s can be accessed by an incoming robot through the empty space at least as wide as two rows of sites.

First, consider the case where S is the root of C'; s can be 1) its seed, 2) a site whose x-coordinate is less than that of the seed of S, or 3) a site whose x-coordinate is larger than that of the seed of S.

1. s is the seed of S: s is the first site to be occupied in the entire structure; thus, s can obviously be accessed: it is not blocked by any robots already

assembled.

2. s is a site whose x-coordinate is less than that of the seed of S: Figure 10.4a illustrates this case. In S, all the sites whose x-coordinate is less than that of s must currently be unoccupied because they are more distal from the seed of S than s. If S does not have any children in terms of C', s can be accessed through the empty space. Even if S has children, all the sites whose x-coordinate is less than (or equal to) that of s must currently be unoccupied in the children because the sites depend on s in terms of G_A. Then, s can be accessed without passing through a gap as wide as a side of a robot because there is an empty corridor at least as wide as three contiguous cells leading to s (recall that a corridor as wide as two contiguous cells is navigable for a robot as shown in Figure 9.3).

3. s is a site whose x-coordinate is larger than that of the seed of S: This case is symmetric to 2) in the previous paragraph. The same argument can also be applied here to verifying the accessibility of s.

Next, consider the case where S is a cell whose y-coordinate is larger than that of its parent in terms of C'; s can be 1) its seed, 2) a site whose x-coordinate is less than that of the seed of S, or 3) a site whose x-coordinate is larger than that of the seed of S.

1. s is the seed of S: Figure 10.4b illustrates this case. s is the first site to

be occupied in S. If S does not have any children in terms of C', s can be accessed through the empty space. Even if S has children, they are currently unoccupied; s can then be accessed through an empty corridor at least as wide as two contiguous cells.

2. s is a site whose x-coordinate is less than that of the seed of S: Figure 10.4c illustrates this case. In S, all the sites whose x-coordinate is less than that of s must currently be unoccupied because they are more distal from the seed of S than s. If S does not have any children in terms of C', s can be accessed through the empty space. Even if S has children, all the sites whose x-coordinate is less than (or equal to) that of s must currently be unoccupied in the children because the sites depend on s in terms of G_A. Therefore, s can be accessed through an empty corridor at least as wide as two contiguous cells.

3. s is a site whose x-coordinate is larger than that of the seed of S: This case is symmetric to 2) in the previous paragraph. The same argument can also be applied here to verifying the accessibility of s.

The remaining case, where S is a cell whose y-coordinate is less than that of its parent, can be addressed similarly to the previous paragraphs by symmetry. □

Chapter 11

Algorithm for Target Structures with Holes

The assembly planning algorithm we present in this chapter can be used to assemble target structures with internal holes, which the algorithm discussed in the previous chapter could not address; however, the resultant plan does not support parallel assembly. Section 11.1 presents the algorithm and Section 11.2 discusses the correctness of the algorithm.

11.1 Algorithm

Our pseudocode is presented in Algorithm 4; we also take advantage of many concepts from graph theory. Algorithm 4 also takes as input an array containing a sequence of the coordinates of the sites of a target structure; see Figure 11.1 for the

example target structure we use to explain the progress of the algorithm. The order of the sequence does not matter and the coordinate of a site actually refers to the coordinate of the centroid of the site, as in the previous algorithm. The algorithm returns a feasible assembly sequence, in essence, a reordering of the input sequence. Algorithm 4 features *dis*assembly planning; however, in the pseudocode, we just say that a "site" is disassembled from a structure, instead of a physical "robot." This is because we do not need physical robots to describe a target structure and we are ultimately interested in establishing an assembly plan as a sequence of sites. Algorithm 4 can be made run in $O(m)$ time, where m is the number of the member sites, as will be explained in the following paragraphs elaborating each line of the algorithm.

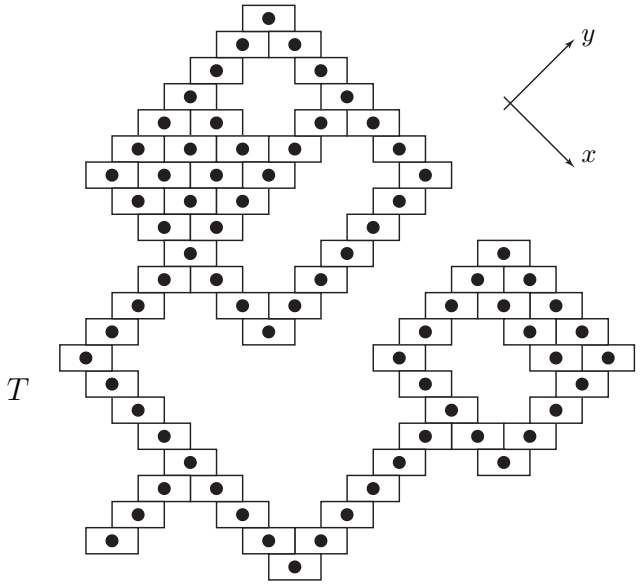


Figure 11.1: A target structure T, which can be represented as a sequence of the coordinates of the centroids (shown as the black dots) of the member sites, with respect to the xy-frame that is oriented as the grid of the lattice units (Figure 9.2).

Algorithm 4 ASSEMBLY-PLANNING(T)

Input: Target structure T: a sequence of sites to be occupied, $T = \langle s_1, s_2, \cdots, s_m \rangle$.
Output: Assembly sequence A: a reordering (permutation) of T.

1: Let A be an empty stack.
2: Construct C.
3: **while** C is not a null graph **do**
4: Construct ∂C.
5: **if** ∂C has a vertex of degree 1, say s_i, **then**
6: Push s_i to A.
7: $C = C - s_i$
8: **else**
9: Search for a *block* of ∂C that contains only one *cutvertex* of ∂C; establish a sequence of sites $\langle s_i, s_j, s_k \rangle$ to be disassembled from the block.
10: Push s_i, s_j, and s_k to A.
11: $C = C - \{s_i, s_j, s_k\}$
12: **end if**
13: **end while**
14: Return A.

Lines 1 – 2: We begin with declaring an empty stack A where we store the resultant assembly plan. We then construct a graph $C = (V, E)$ (recall Section 9.2) representing the mechanical connectivity of the target structure (Figure 11.2). C can be constructed in $O(m)$ time by considering that each site has $O(1)$ neighboring sites.

Lines 3 – 13: We are in the main loop where we delete one or more vertices in C and push them to A in each iteration until there remains no vertex to delete. Each process is elaborated in the following paragraphs.

Line 4: At the start of the iteration of the loop, we construct a graph ∂C that is the frontier of the outer face of C (Figure 11.2). This can be done in $O(m)$ time as follows. First, locate any site that is on the frontier and of degree less than 3

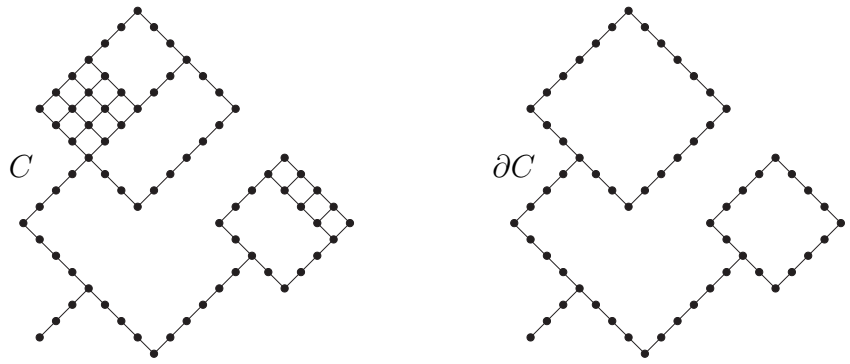


Figure 11.2: C, the graph representing the mechanical connectivity of T, and ∂C, the subgraph of C that is the frontier of the outer face.

by, for example, picking the site with the smallest x-coordinate among the ones with the largest y-coordinate. Turn clockwise (counterclockwise) from the site by consistently moving along the leftmost (rightmost) edge whenever there is a vertex of degree 3 until we return to the site. Then each vertex on the outer frontier is visited at most twice.

Lines 6 – 7: If ∂C has a vertex of degree 1, the vertex is pushed to the stack A and deleted from C along with any incident edges (Figure 11.3). These operations can be done in $O(m)$ time.

Lines 9 – 11: A block of a graph is either a *maximal 2-connected subgraph*, or a *bridge*, or an isolated vertex. For ∂C here that does not have a vertex of degree less than 2, we can always find a block with one or no cutvertex of the original graph; moreover, such a block must be a cycle (see Lemma 4 for the proof). We pick such a cycle, either $\mathcal{B}_1$ or $\mathcal{B}_2$ in Figure 11.4a, and establish a sequence of sites that can be disassembled from the cycle as follows. Suppose that we picked $\mathcal{B}_1$. Consider

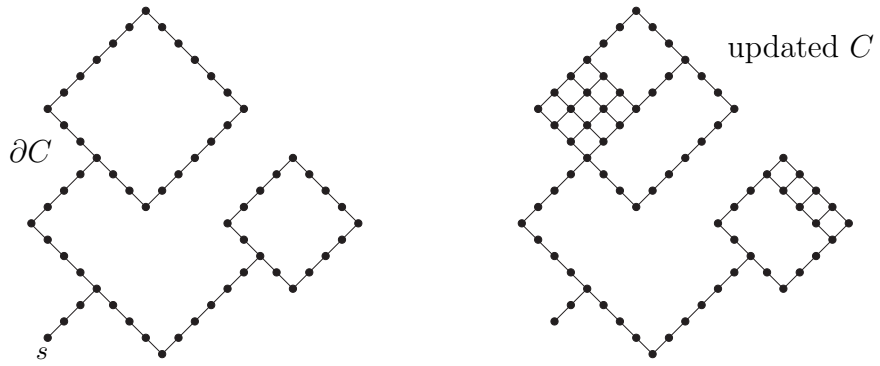


Figure 11.3: Since ∂C has a vertex of degree 1, s, the vertex is to be disassembled. The next iteration starts with the updated C.

the site with the smallest x-coordinate on the row of the largest y-coordinate; the site is denoted as s_i in Figure 11.4b. Next, we inspect if a site $s_{i,\text{SE}}$ (SE stands for "southeast" with respect to the reference frame shown in Figure 11.4a) is a vertex of C or not. If so, we establish a sequence $\langle s_i \rangle$ as the potential output of this line. If not, as can actually be seen in Figure 11.4b, we pick two more sites, s_j and s_k, and establish $\langle s_i, s_j, s_k \rangle$ as the potential output. We can repeat the process by letting s_i be the site with the largest x-coordinate on the row of the smallest y-coordinate (Figure 11.4c): we inspect if a site $s_{i,\text{NW}}$ (NW stands for "northwest" with respect to the reference frame) is a vertex of C or not, as illustrated in Figure 11.4c; according to the result, we establish either $\langle s_i \rangle$ or $\langle s_i, s_j, s_k \rangle$ as the potential output in a similar manner. Between the two sequences established for the potential output, pick one without the cutvertex for the output of Line 9. In Lines 10 – 11, we push s_i (or s_i, s_j, and s_k in the order enumerated) to A; we then delete s_i (or s_i, s_j, and s_k) from C along with any incident edges. These lines can be done in $O(m)$ time:

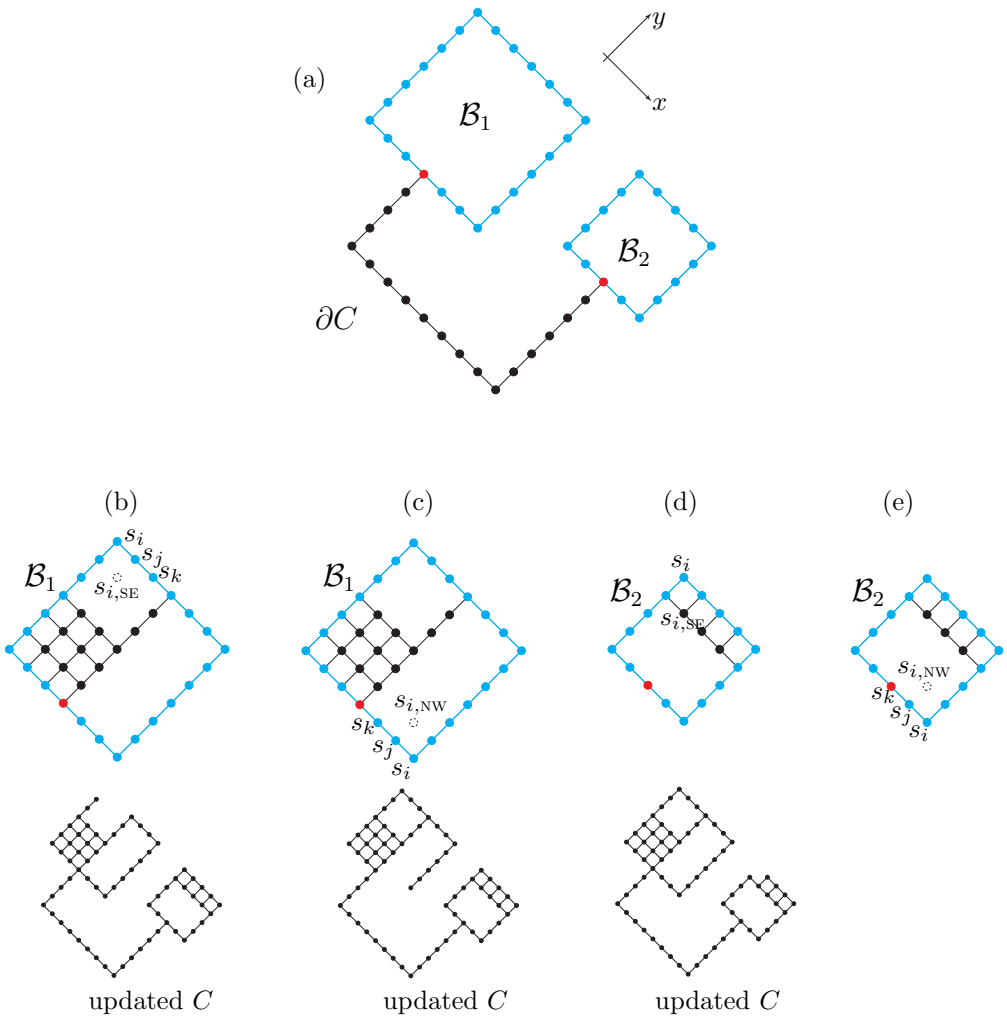


Figure 11.4: (a) ∂C currently has two blocks that are cycles, denoted as $\mathcal{B}_1$ and $\mathcal{B}_2$. Each red vertex is a cutvertex of ∂C that belongs to $\mathcal{B}_1$ or $\mathcal{B}_2$. (b), (c) Suppose that we picked $\mathcal{B}_1$. s_i, s_j, and s_k can potentially be disassembled from the structure; C will then be updated as the graph shown below. In case we picked $\mathcal{B}_2$, (d) s_i can potentially be disassembled from the structure; (e) because s_k is the cutvertex, the sequence $\langle s_i, s_j, s_k \rangle$ will not be returned.

blocks and cutvertices can be computed in $O(m)$ time (Tarjan 1972).

Line 14: The algorithm finally returns A.

Instructions : We get an assembly sequence by popping elements from A. Following the sequence, we assemble the target structure by occupying the member sites one by one. In contrast to Algorithm 3 that outputs a partial order on the sites of a target structure, the output of Algorithm 4 is a *total order* on the member sites. Therefore, parallel assembly is not supported by Algorithm 4 although we can address target structures with internal holes.

11.2 Analysis

We show the completeness and correctness of Algorithm 4: it terminates and returns a feasible assembly sequence without the accessibility issues discussed in Figure 9.4. We break the analysis into some lemmas.

Lemma 4. *Line 9 of Algorithm 4 returns a nonempty sequence.*

Proof. We begin with verifying that there exists a block of ∂C with one or no cutvertex and the block is a cycle. Different blocks overlap in at most one vertex, which is then a cutvertex of the original graph (Diestel 2000). Consider the *block graph* (Diestel 2000) of ∂C: let A denote the set of ∂C's cutvertices and B the set of ∂C's blocks; the block graph is a bipartite graph on $A \cup B$ formed by the edges (a, B) where $a \in B$. The block graph is a tree (the block graph of a connected

graph is a tree (Diestel 2000)). Therefore, any leaf of the tree is indeed a block of ∂C with one or no cutvertex. The block represented by the leaf cannot be a bridge or an isolated vertex here because we have deleted all the vertices of degree 1 before executing Line 9. Therefore, the block must be a 2-connected subgraph, which in turn is a cycle here by applying Proposition 3.1.3. of Diestel (2000).

According to the paragraph elaborating Line 9, we can get two distinct, non-empty sequences of sites to be disassembled from the block. Because the block contains less than two cutvertices of ∂C, one of the sequences can always be returned as the output of Line 9. □

Lemma 4 ensures that the number of vertices of C keeps decreasing as we iterate the process of Lines 4 – 11; thus, Algorithm 4 terminates.

Next, we show that the following property holds for Algorithm 4 as a loop invariant.

Lemma 5. *At the start of each iteration of the* ***while*** *loop in Algorithm 4, C satisfies the assumptions stated in Section 9.2: C is connected and free of narrow corridors.*

Proof. We must verify the following two items to see the property holds as a loop invariant: (1) **Initialization** (the property is true prior to the first iteration); (2) **Maintenance** (if the property is true before an iteration, it remains true before the next iteration).

Initialization: Before the first loop iteration, C satisfies the property by our assumption on target structures.

Maintenance: Next, we show that each iteration maintains the property. On the one hand, in Lines 6 – 7, disassembling a site of degree 1 does not disconnect C and the frontiers of the faces of the resultant ∂C remain reachable by the rhombus (recall Figure 9.3c) as can be seen in Figure 11.5a, where deleting the site, s_i, just opens up the short cut colored green. On the other hand, in Lines 9 – 11, we disassemble either one (s_i) or three sites (s_i, s_j, and s_k). First, disassembling s_i alone does not disconnect C because s_i is disassembled from a cycle, which is 2-connected, and there is no site adjacent to s_i (in terms of C) inside the cycle (see Figure 11.4d). The frontiers of the faces of the resultant ∂C remain reachable by the rhombus as can be seen in Figure 11.5b, where removing s_i just opens up the new detour colored green. Second, disassembling s_i, s_j, and s_k does not disconnect C because they are contiguous in a cycle and there is no site adjacent to them (in terms of C) inside the cycle (see Figures 11.4b and 11.4c). The frontiers of the faces of the resultant ∂C remain reachable by the rhombus because disassembling the three contiguous sites opens a gap wide enough for the rhombus to enter a new area that used to be an inner face of C (Figure 11.5c). □

When Algorithm 4 terminates, C is a null graph. Therefore, Lemma 5 guarantees that when we assemble a structure according to the computed assembly sequence, the structure grows from a single seed robot as a single connected component always

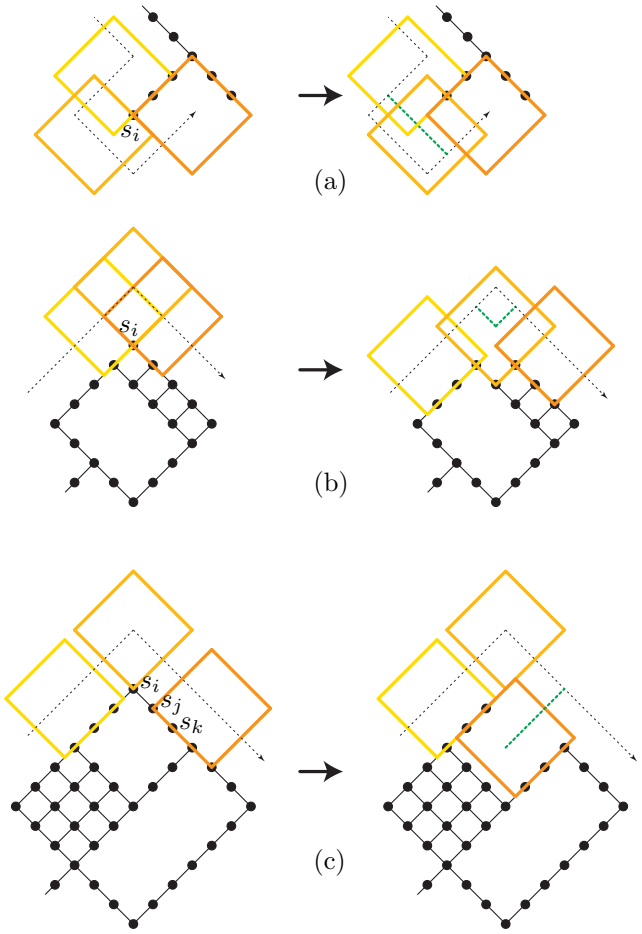


Figure 11.5: Each panel is zooming in on some part of C shown in Figure 11.2 along with the rhombus, as appeared in Figure 9.3c, translating along the frontier of ∂C; the centroid of the rhombus follows the dotted lines. The panels (a), (b), and (c) show what happens to the path of the rhombus after sites are removed from C according to Algorithm 4.

satisfying the assumptions of Section 9.2.

Because assembly happens only on ∂C, a robot can reach a site open for occupancy without being blocked by a physical wall or a narrow corridor. In addition, our last lemma ensures that a site open for occupancy is not flanked between two physical robots already docked in the structure (recall Figure 9.4).

Lemma 6. *In each iteration of the* ***while*** *loop in Algorithm 4, each disassembly event can be performed in such a manner that the site to be disassembled is not located between two other sites that form a gap as large as a side of a site.*

Proof. If a site to be disassembled is a vertex of degree 1 in ∂C (Lines 6 – 7), the site has at most one neighbor. Thus, the site is not located between two other sites. In Lines 9 – 11, if we disassemble s_i, s_j, and s_k one at a time in the order enumerated, each of them is not flanked by two other sites that form a gap as large as a side of a site. □

Chapter 12

Implementation and Experiment

In this chapter, we discuss the implementation of our assembly planning algorithms. Section 12.1 introduces our software implementation. Sections 12.2 and 12.3 present a set of simulations and experiments.

12.1 Assembly Planner

We wrote a software package[12] that implements Algorithms 3 and 4, written in C++. The software, which we call the *Assembly Planner*, parses a blueprint for a target structure and returns an assembly plan that specifies a partial/total order for filling the sites of the target structure. Given the shape of the current structure, the software is capable of returning open site(s) that can be occupied simultaneously around the current structure.

[12] Available at `http://www.seas.upenn.edu/~juse`

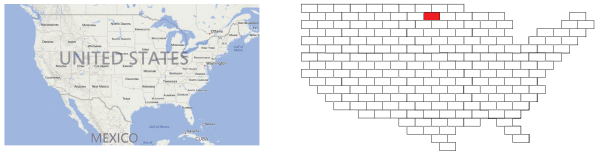
The Assembly Planner is a part of a software suite that operates the robotic boats introduced in Section 9.1. At the highest level, the suite is composed of the Coordinator, the Assembly Planner, the Trajectory Planner, and the Docking Routine. The Coordinator is a state machine that handles the operation of the other components. We addressed the Assembly Planner in the previous paragraph. The Trajectory Planner generates feasible paths for robots to reach sites open for occupancy. The Docking Routine enables a robot to perform a sequence of actions for docking with other robot. The software components were built on top of a middleware platform, ROS[13]; For more details on software design, see O'Hara et al. (2014).

12.2 Experiment 1: Assembly Planning

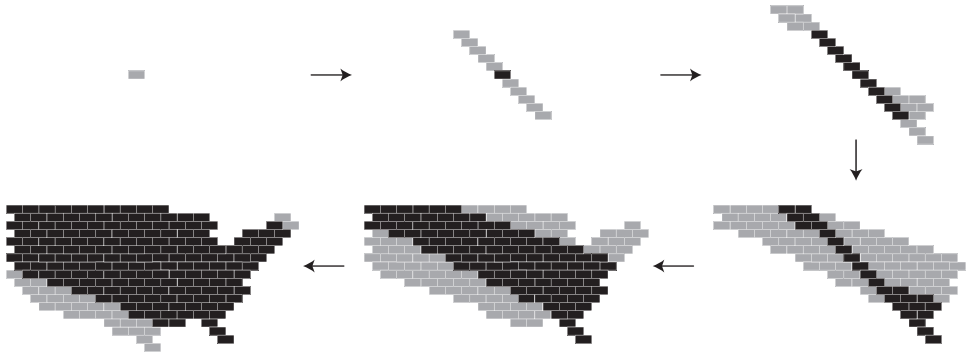
We first ran computer simulations of how Algorithm 3 works for target structures without holes. Figure 12.1a shows a target structure composed of 207 member sites that looks like the continental United States. Given an arbitrary choice of the seed, denoted as $\mathfrak{s}$ in Line 3 of Algorithm 3, it took 0.007 second on average for the Assembly Planner running on a 4GB, 2.53 GHz machine to compute G_A and its topological sort. Figure 12.2a shows a target structure composed of 435 member sites that looks like a row of piers in a harbor. It took 0.015 second on average to compute an individual assembly plan. Figures 12.1b and 12.2b show

[13] `http://www.ros.org`

some intermediate snapshots on the way to the target structures.



(a)

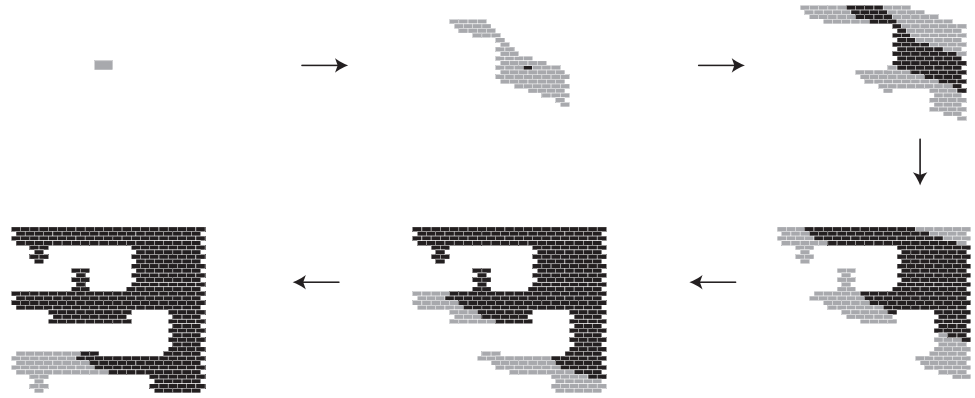


(b)

Figure 12.1: (a) The right panel shows a target structure composed of 207 member sites, which looks like the continental United States. Choosing the red site as the seed can minimize the assembly time by maximizing parallelism. (b) Six snapshots showing how the given seed grows into the target structure. In each panel, the gray sites are to be occupied in the next snapshot around the current structure colored black.

Applying Algorithm 3, we can find the best seed that results in the minimum *height* of the resultant directed acyclic graph, where the height of the graph is the height of its root, that is, the length of the longest path between the root and a leaf. By minimizing the height, we can make the most of the parallelism of a swarm of the robotic boats, which can result in the fastest assembly time. Given a target structure with m member sites, it takes $O(m^2)$ time to compute the optimal assembly plan starting from the best seed: designating each member site as the

(a)



(b)

Figure 12.2: (a) The right panel shows a target structure composed of 435 member sites, which looks like the harbor on the left panel. Choosing the red site as the seed can minimize the assembly time by maximizing parallelism. (b) Six snapshots showing how the given seed grows into the target structure. In each panel, the gray sites are to be occupied in the next snapshot around the current structure colored black.

seed, we repeatedly apply Algorithm 3, running in $O(m)$ time; finding the height of the resultant graph can be done in $O(m)$ time using a breadth-first search.

In Figures 12.1a and 12.2a, the seeds that can derive the "shortest" assembly plans are colored red. Note that the snapshots of Figures 12.1b and 12.2b do not show the progress of the optimal plan. Indeed, following an assembly plan other than the optimal one can be a better idea in some cases. For example, although the robots are geometrically identical, they may provide different functions; then we may want

to grow the structure from the site that accommodates "the most important" robot. For the USA example (Figure 12.1a), the minimum height of the resultant graph was 31. Suppose that sites are occupied immediately as soon as they are discovered by the Assembly Planner. We could then assemble maximally 11 robots at a time, by following the optimal plan. For the harbor example (Figure 12.2a), the minimum height and the maximum growth rate were 57 and 13, respectively. The results of the above experiments are summarized in Table 12.1.

		Computation time			
Target	Sites	Individual (avg.)	Optimal	Min. height	Max. growth rate
USA (Figure 12.1)	207	0.007s	1.68s	31	11 robots
harbor (Figure 12.2)	435	0.015s	10.75s	57	13 robots
100-by-100 square	10,000	0.607s	7604.26s	248	84 robots

Table 12.1: The results of assembly planning experiments. For the optimal plans, we iterated over all member sites for the USA and harbor examples; we sampled 500 sites for the 100-by-100 square.

The minimum height can also be interpreted as the complexity of a target structure itself in terms of parallel assembly with a swarm of robots. For example, although the harbor example (Figure 12.2a) has more than twice the number of sites that the USA example (Figure 12.1a) has, the time to assemble the harbor example can be less than twice the time it takes to assemble the USA example (compare the minimum heights). Another example is shown in Figure 12.3. The two structures of Figure 12.3 have the same number of member sites; however, the structure of Figure 12.3a can be assembled faster by taking advantage of parallel assembly.

Figure 12.4 shows some snapshots that illustrate the assembly plans for the three

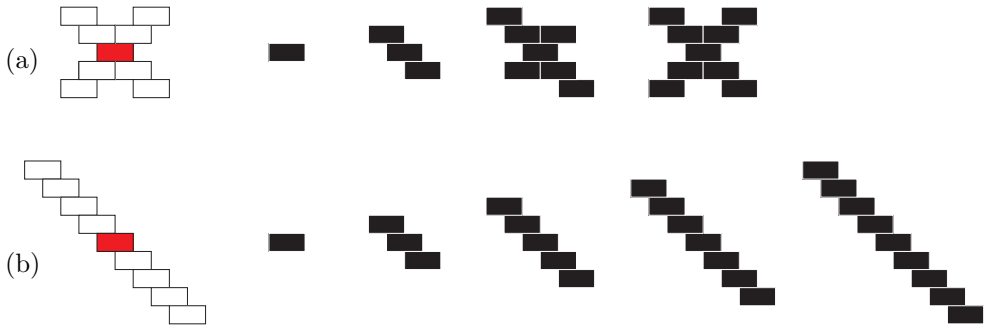


Figure 12.3: The Assembly Planner computed the optimal assembly plans for the structures, starting from the seeds colored red. The minimum heights of (a) and (b) were 3 and 4, respectively.

target structures with holes, letters 'R', 'O', and 'B', returned by the Assembly Planner running Algorithm 4. Each letter is composed of 46 sites and it took 0.03s to compute each assembly plan.

12.3 Experiment 2: Assembling Robotic Boats

We built real structures with our hardware/software implementation; see O'Hara et al. (2014) for details. Here we summarize the result by showing two structures autonomously built with the robots. Figure 12.5a shows a structure of six robots that was used as a landing platform for the quadrotor. Figure 12.5b shows a bridge that spanned a corner of the pool. Figure 12.5c shows some snapshots that illustrate the computed assembly plan for the bridge. The assembly begins with occupying the site colored red. Note that the seed does not result in the optimal assembly plan; however, it is reasonable to start constructing from one end of the bridge.

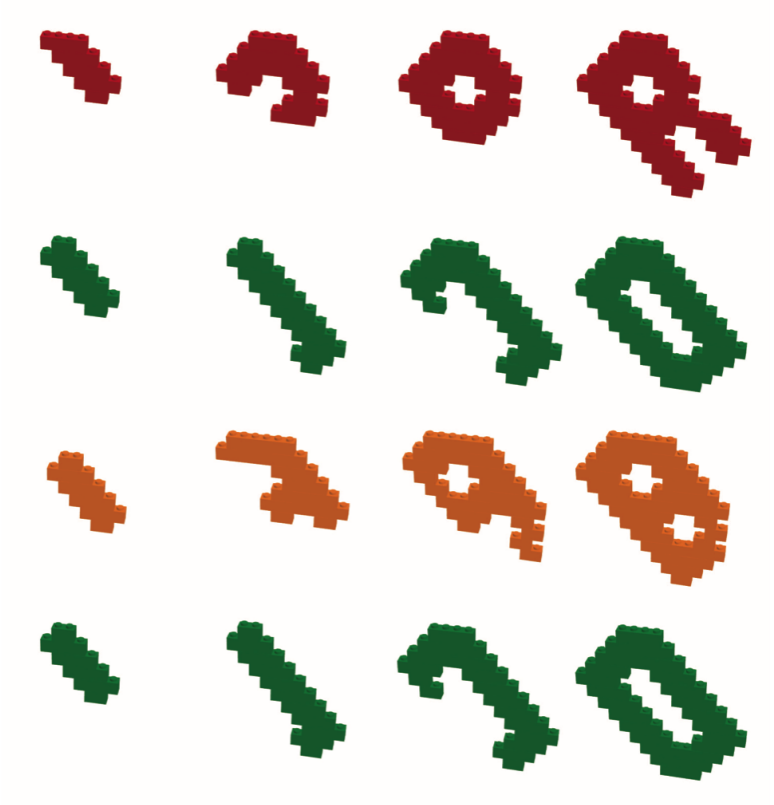
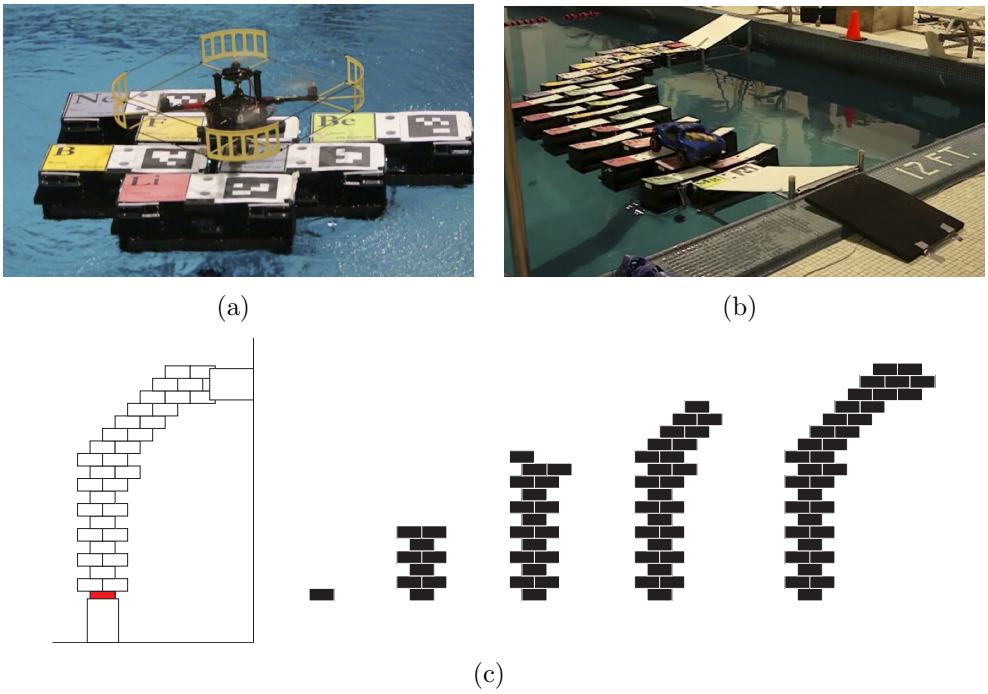


Figure 12.4: Assembling congruent lego blocks into the four letters with holes, ‘R’, ‘O’, ‘B’, and ‘O’.



(a) (b)

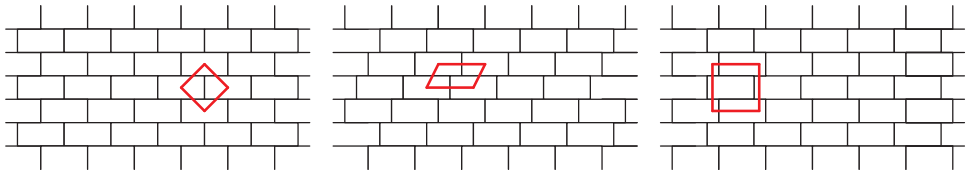
(c)

Figure 12.5: Structures autonomously constructed with our robotic boats that are 0.5m in length. (a) A landing platform for an aerial vehicle. (b) A bridge for a ground vehicle. (c) For the bridge, we had the Assembly Planner compute a plan starting from one end (the site colored red). The snapshots show how the bridge grows.

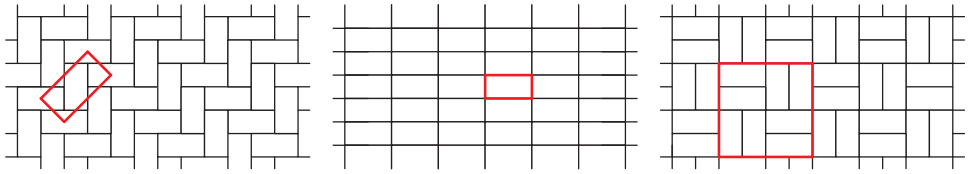
Chapter 13

Extension to Other Patterns

In this chapter, we investigate how to extend our assembly planning algorithms to other patterns formed by congruent rectangles.



(a) ***cmm*** (b) ***p2*** (c) ***pmg***



(d) ***pgg*** (e) ***pmm*** (f) ***p4g***

Figure 13.1: Six symmetric patterns that can be generated by congruent rectangles. The red polygons are the lattice units of the patterns.

On the plane, there are 17 distinct groups that classify repetitive patterns based on their symmetries; the groups are called *plane symmetry groups*, *plane crystallo-*

graphic groups, or *wallpaper groups*. Coxeter (1961) showed how six of the 17 groups can be produced by tessellating the plane with congruent rectangles; see Figure 13.1. So far, we have been addressing the common brick wall pattern that is classified into one of the groups called ***cmm*** (see Schattschneider (1978) for the nomenclature). Among the six patterns shown in Figure 13.1, Algorithms 3 and 4 can directly be applied to constructing structures of type not only ***cmm*** (Figure 13.1a) but also ***p2*** (Figure 13.1b), ***pmg*** (Figure 13.1c), ***pgg*** (Figure 13.1d), and ***pmm*** (Figure 13.1e).

Consider a target structure of type ***p2***, ***pmg***, ***pgg***, or ***pmm***. The target structure can unambiguously be transformed into a structure of type ***cmm***, with preserving geometric adjacency and/or mechanical connectivity (Figure 13.2). On the one hand, types ***p2***, ***pmg***, and ***pgg*** are topologically identical to type ***cmm*** in the sense that every rectangle is surrounded by six other rectangles. In the structures of types ***p2***, ***pmg***, and ***pgg*** shown in Figure 13.2, it can be seen that the rectangles with the same coordinate have the identical set of neighbors. Moreover, by assuming again that two robots dock only along their long sides for types ***p2*** and ***pmg*** (only longside-shortside docking for type ***pgg***), the graph C representing the mechanical connectivity remains invariant under the transformation. For example, the robot at $(0, 0)$ is mechanically connected only to the robots at $(0, 1)$, $(-1, 0)$, $(0, -1)$, and $(1, 0)$ in all the structures of types ***cmm***, ***p2***, ***pmg***, and ***pgg*** shown in Figure 13.2. On the other hand, type ***pmm*** is not topologically identical to type ***cmm***. If robots can dock along all of their sides, however, it is possible to transform

a target structure of type ***pmm*** into one of type ***cmm*** preserving the mechanical connectivity; see Figure 13.2.

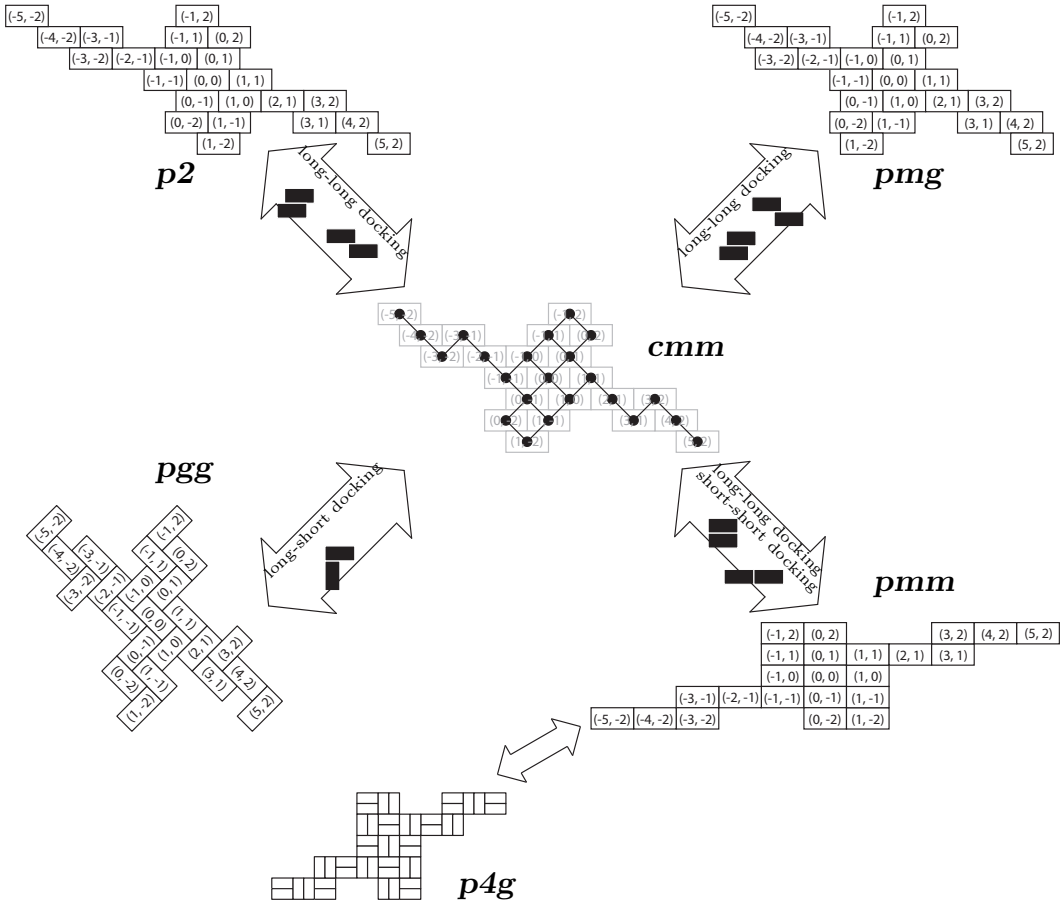


Figure 13.2: A target structure of type ***cmm*** with the graph C (in the center) and the equivalent ones of other types in the sense of geometric adjacency/mechanical connectivity. Also shown in the block arrows are modes of docking between two robots sufficient to assemble the structures (symmetric cases are omitted); then, the structures of types ***cmm***, ***p2***, ***pmg***, ***pgg***, and ***pmm*** have the same mechanical connectivity represented as C. The structure of type ***p4g*** is composed of the meta modules, each of which is composed of two rectangles.

Suppose that the transformed target structure, tessellated in type ***cmm***, is consistent with the constraints discussed in Section 9.2; then, the assembly plan obtained by applying Algorithm 3 or 4 to the structure of type ***cmm*** can also be

applied to correctly assembling the original target structure, tessellated in type ***p2***, ***pmg***, ***pgg***, or ***pmm***, as will be shown in the following theorem.

Remark : Note that for structures of type ***pgg*** tessellated as shown in Figure 13.2, the algorithms have to be modified as follows. First, in the rule presented in Line 3 of Algorithm 3, switch the places of the two words 'largest' and 'least.' Second, in Lines 9 - 11 of Algorithm 4, s_i should be the site with the largest (smallest) x-coordinate on the row of the largest (smallest) y-coordinate.

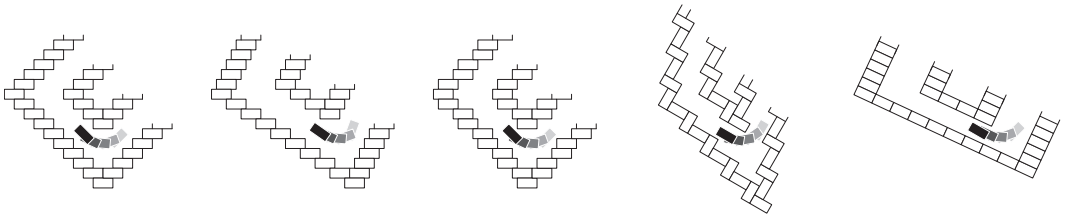
Theorem 4. *Algorithms 3 and 4 can also be applied to constructing target structures of types* ***p2****,* ***pmg****,* ***pgg****, and* ***pmm*** *formed by congruent rectangular robots in such a manner that the robots do not have to pass through a gap that is as large as a side of a robot formed between two robots docked in the structure.*

Proof. To see that this theorem holds, it is sufficient to show that Lemmas 1 to 6, proved for type ***cmm***, also hold for types ***p2***, ***pmg***, ***pgg***, and ***pmm***.

Lemmas 1, 2, 4, and 5 hold for target structures of types ***p2***, ***pmg***, ***pgg***, and ***pmm*** because the lemmas depend only on the properties of the graph C, representing the mechanical connectivity, which is invariant under the transformation between type ***cmm*** and the four types (***p2***, ***pmg***, ***pgg***, and ***pmm***), as shown in Figure 13.2. Note that for Lemma 5, we need to assume a particular embedding for C, that is, the embedding shown in Figure 9.3c.

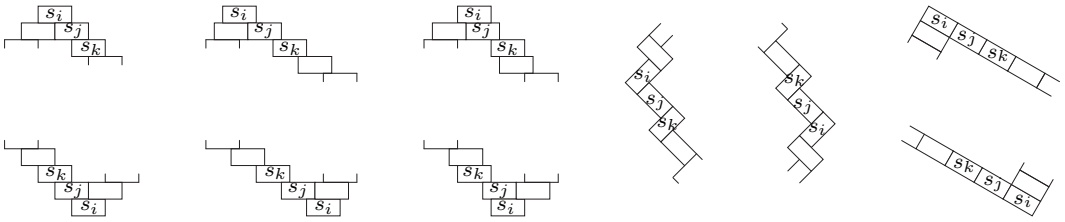
To see Lemma 3 holds, first recall Figure 10.2 that shows a corridor as wide

as two contiguous cells in type ***cmm*** is navigable for a robot; then, it suffices to verify that such a corridor remains navigable under the transformation between type ***cmm*** and the others. Figure 13.3a shows a navigable corridor in type ***cmm***; Figures 13.3b, 13.3c, 13.3d, and 13.3e show the structures of the other types in one-to-one correspondence with the structure of Figure 13.3a. From the figures, it can be verified that a robot can also navigate through the corridors of the other types at least by omnidirectional translation.



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

Figure 13.3: Corridors as wide as two contiguous cells are navigable for all the patterns: (a) ***cmm***, (b) ***p2***, (c) ***pmg***, (d) ***pgg***, and (e) ***pmm***. It can be seen that the robot, colored black, can navigate the corridors at least by omnidirectional translation.



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

Figure 13.4: Disassembling s_i, s_j, and s_k one at a time in the order enumerated guarantees easy accessibility (a robot does not have to pass through a gap as wide as its side) for not only (a) type ***cmm*** but also (b) types ***p2***, (c) ***pmg***, (d) ***pgg***, and (e) ***pmm***.

Finally, we show that Lemma 6 holds for types ***p2***, ***pmg***, ***pgg***, and ***pmm***. First, it is straightforward to see that if a site to be disassembled is a vertex of

degree 1 in ∂C (Lines 6 - 7 of Algorithm 4), the site is not located between two other sites because the site has at most one neighbor. Figure 13.4a reminds us that the disassembly involved with Lines 9 - 11 of Algorithm 4 can be done without having a robot pass through a gap as wide as its side if we disassemble s_i, s_j, and s_k in the order enumerated (the upper (lower) panel is involved with Figure 11.4b (11.4c)). Figures 13.4b, 13.4c, 13.4d, and, 13.4e show the structures of the other types in one-to-one correspondence with the structure of Figure 13.4a. It can also be seen that disassembling s_i, s_j, and s_k in the order enumerated guarantees easy accessibility. □

Algorithms 3 and 4 can also be applied to target structures of type ***p4g***, but in a less direct manner. We first propose to subassemble meta modules, each of which is composed of two rectangles. Then it is possible to transform a structure of type ***p4g*** into one of type ***pmm***, which can then be transformed into one of type ***cmm*** as discussed earlier; see Figure 13.2.

Our algorithms and their generalization discussed so far have been based on conservative assumptions in terms of robots' docking capability; for example, we assumed that only longside-longside docking is available for robots to form structures of types ***cmm***, ***p2***, or ***pmg*** (longside-shortside docking for ***pgg***). Our algorithms can also be directly applied to the cases where the robots are outfitted with more capable docking hardware (for example, suppose that longside-shortside docking is additionally available for robots to form type ***cmm***) by virtue of the

conservativeness.

Part IV

Conclusion

Chapter 14

Conclusion

In this chapter, we conclude the thesis with summarizing main contributions in Section 14.1 and discussing future research directions in Sections 14.2 and 14.3.

14.1 Summary of Contributions

The contributions of the thesis are twofold.

A theoretical framework for grasping objects with curved effectors and its application to a modular robot system: We have presented three types of immobilizing grasps and cages and showed that they can be applied to a wide range of objects including polyhedra. Each of the grasps or cages is formed by at most three appropriately curved effectors. The immobilizing grasps depend on the local curvature properties of the effectors, whereas the cages are formulated by

global parameters, such as the distance between the effectors. A stable grasp can be obtained from any of our cages by a squeezing motion. The collection of grasps and cages was extended to include more types of grasps and cages for two- or three-dimensional objects. Based on the theory, we developed hardware and software implementing our modular approach to grasping objects using end-effectors with curved surfaces.

Assembly planning algorithms that can be applied to the collective construction of planar structures with rectangular modules: We have presented two provably correct and complete assembly planning algorithms for constructing arbitrary planar target structures, possibly with internal holes, with congruent, rectangular mobile units. The algorithms build on graph theory and return an assembly plan, represented as a partial or total order, in linear time. The resultant assembly plan guarantees easy accessibility in the sense that a free robot does not have to pass through a narrow gap while approaching its target position on the boundary of the growing structure. For target structures without holes, the resultant assembly plan allows parallel assembly and can be executed in a decentralized manner. The algorithms were initially designed to address the common brick wall pattern, but can also address other symmetric patterns that can be formed by the collection of congruent rectangles.

14.2 Future Work

We here describe our future work under three themes: extending theory, enhancing autonomy, and more applications.

Extending theory: Our theoretical framework can be extended in a number of directions. Future work for the grasping work presented in Part II includes the following. In order to guarantee performance under frictional, compliant contact, the effects of friction and compliance should be treated in a more quantitative manner. Approaches may include incorporating accurate and tractable models of finger-object contact, contact friction, and the compliance of the robot joints and links. For the assembly work presented in Part III, two avenues of future work include the following. By incorporating dynamic stability analysis, our assembly planning algorithms may direct the growth of the structure in such a way that is the most robust to external disturbances due to, for example, waves and tides. In addition, it may be fruitful to extend the algorithms to address three-dimensional structures; one approach is to generalize Chapter 13 into *space groups* (also known as *crystallographic groups*), the symmetry groups of a configuration in three-dimensional space.

Enhancing autonomy: There are many great benefits to enhanced autonomy. For example, autonomous robotic systems can eliminate human errors and work safely in dangerous environments such as outer space or the ocean floor. Our objec-

tives for enhancing autonomy include incorporating more sensing capabilities; for example, visual sensing will be beneficial to not only perceiving objects to grasp but also making the docking procedure more robust. The level of autonomy can also be greatly increased by enabling proper interaction with the environment, which will facilitate grasping an object in clutter or assembling an adaptive structure on water around an island. Another interesting direction is to develop hardware and software supporting efficient self-reconfiguration: such a system can truly perform the best reconfiguration plan to replace a module that is not working normally in an assembly or adapt itself to grasp an arbitrary object by autonomously exchanging end-effector or attaching more arm links.

More applications: Effectors with curved contact surfaces can be used as not only "hands" but also "feet" for walking or running. RHex (Saranli et al. 2001, Johnson and Koditschek 2013) showed the versatility of single-bodied legs without any internal degrees of freedom. Leg stiffness will be an important factor as discussed by Galloway et al. (2011). Curved effectors can also be useful in manufacturing, eliminating the need of redesigning fixtures according to objects. By considering static stability in the presence of external forces such as gravity, our assembly planning algorithms can be applied to constructing vertical walls and buildings. The scenario can be made more practical by considering a bipartite system composed of passive bricks and mobile robots manipulating the bricks, which can be a modular manipulator discussed in Part II of the thesis. We expect that such a capability can

facilitate challenges commonly encountered in space exploration, disaster response, or assisted living.

14.3 Milestones for Our Vision

Considering the vignettes presented in Section 1.3, we finally propose possible 5-, 10-, and 15-year milestones toward the vision.

Autonomous grasping : We ultimately want robots to safely grasp a large class of objects with different material properties, even in a cluttered environment. It is necessary to combine our model-based approach presented in the thesis with extensive visual and/or tactile perception and machine learning technologies. Our roadmap includes the following goals:

- 5 years: Achieve the ability to safely grasp a group of objects with known mechanical properties.
- 10 years: Achieve the ability to safely grasp objects with unknown mechanical properties.
- 15 years: Achieve the ability to safely grasp objects with unknown mechanical properties while properly interacting with the environment.

Adaptable self-assembly planning : The thesis presented planning algorithms for assembling planar, horizontal structures. Extending the algorithms to assemble

a wider range of structures (for example, vertical walls and three-dimensional structures on the ground) will require achievements including the following milestones:

- 5 years: Assembly planning algorithms incorporating static/dynamic stability analysis for two-dimensional target structures.
- 10 years: The algorithms will be capable of addressing environmental constraints.
- 15 years: The algorithms will be extended to assemble three-dimensional target structures.

Other critical capabilities that will realize the vision include *self-reconfigurability* and *dexterous manipulation*.

Bibliography

Oswin Aichholzer, Erik D. Demaine, Jeff Erickson, Ferran Hurtado, Mark Overmars, Michael A. Soss, and Godfried T. Toussaint. Reconfiguring convex polygons. *Computational Geometry: Theory and Applications*, 20(1–2), October 2001.

Thomas Allen, Joel Burdick, and Elon Rimon. Two-fingered caging of polygons via contact-space graph search. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4183–4189. IEEE, 2012.

Esther M Arkin, Robert Connelly, and JS Mitchell. On monotone paths among obstacles with applications to planning assemblies. In *Proceedings of the fifth annual symposium on Computational geometry*, pages 334–343. ACM, 1989.

Spring Berman, Quentin Lindsey, Mahmut Selman Sakar, Vijay Kumar, and Stephen C Pratt. Experimental study and modeling of group retrieval in ants as an approach to collective transport in swarm robotic systems. *Proceedings of the IEEE*, 99(9):1470–1481, 2011.

Markus Bernard, Konstantin Kondak, Ivan Maza, and Anibal Ollero. Autonomous

transportation and deployment with aerial robots for search and rescue missions. *Journal of Field Robotics*, 28(6):914–931, 2011.

A. Bicchi and V. Kumar. Robotic grasping and contact: A review. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2000.

J Bishop, Samuel Burden, Eric Klavins, R Kreisberg, W Malone, Nils Napp, and T Nguyen. Programmable parts: A demonstration of the grammatical approach to self-organization. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3684–3691. IEEE, 2005.

Christoph Borst, Max Fischer, and Gerd Hirzinger. A fast and robust grasp planner for arbitrary 3d objects. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 3, pages 1890–1896. IEEE, 1999.

Prosenjit Bose, David Bremner, and Godfried T. Toussaint. All convex polyhedra can be clamped with parallel jaw grippers. *Computational Geometry: Theory and Applications*, 6:291 – 302, 1996.

TGR Bower. Demonstration of intention in the reaching behaviour of neonate humans. *Nature*, 228:679–681, 1970.

David L Bowers and Ron Lumia. Manipulation of unmodeled objects using intelligent grasping schemes. *Fuzzy Systems, IEEE Transactions on*, 11(3):320–330, 2003.

Randy C Brost and Kenneth Y Goldberg. A complete algorithm for designing planar fixtures using modular components. *Robotics and Automation, IEEE Transactions on*, 12(1):31–46, 1996.

Kevin Q Brown. *Geometric transforms for fast geometric algorithms*. PhD thesis, Carnegie-Mellon University Pittsburgh, PA, 1979.

Joel W Burdick, Jim Radford, and Gregory S Chirikjian. A sidewinding locomotion gait for hyper-redundant robots. *Advanced Robotics*, 9(3):195–216, 1994.

Zack Butler, Keith Kotay, Daniela Rus, and Kohji Tomita. Generic decentralized control for lattice-based self-reconfigurable robots. *The International Journal of Robotics Research*, 23(9):919–937, 2004.

David J Cappelleri, Michael Fatovic, and Zhenbo Fu. Caging grasps for micromanipulation & microassembly. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 925–930. IEEE, 2011a.

David J Cappelleri, Michael Fatovic, and Utsav Shah. Caging micromanipulation for automated microassembly. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3145–3150. IEEE, 2011b.

Peng Cheng, Jonathan Fink, Vijay Kumar, and Jong-Shi Pang. Cooperative towing with multiple robots. *Journal of mechanisms and robotics*, 1(1):011008, 2009.

R. Connelly, E.D. Demaine, and G. Rote. Straightening polygonal arcs and convexifying polygonal cycles. *Discrete Comput Geom*, 30:205–239, 2003.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 2001.

Harold Scott Macdonald Coxeter. *Introduction to geometry*. New York, London, 1961.

Mark R Cutkosky and Imin Kao. Computing and controlling compliance of a robotic hand. *Robotics and Automation, IEEE Transactions on*, 5(2):151–165, 1989.

Jurek Czyzowicz, Ivan Stojmenovic, and Jorge Urrutia. Immobilizing a polytope. In *Algorithms and data structures*, pages 214–227. Springer, 1991.

C. Davidson and A. Blake. Caging planar objects with a three-finger one-parameter gripper. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2722–2727. IEEE, 1998a.

Colin Davidson and Andrew Blake. Error-tolerant visual planning of planar grasp. In *Computer Vision, 1998. Sixth International Conference on*, pages 911–916. IEEE, 1998b.

Mark de Berg, Marc Van Kreveld, Mark Overmars, and Otfried Cheong Schwarzkopf. *Computational geometry*. Springer, 2000.

Thomas De Fazio and Daniel E Whitney. Simplified generation of all mechanical assembly sequences. *Robotics and Automation, IEEE Journal of*, 3(6):640–658, 1987.

Rosen Diankov, Siddhartha S Srinivasa, Dave Ferguson, and James Kuffner. Manipulation planning with caging grasps. In *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, pages 285–292. IEEE, 2008.

Reinhard Diestel. *Graph Theory {Graduate Texts in Mathematics; 173}*. Springer-Verlag Berlin and Heidelberg GmbH & Company KG, 2000.

Mehmet Dogar and Siddhartha Srinivasa. A framework for push-grasping in clutter. *Robotics: Science and Systems VII*, 2011.

Aaron M Dollar and Robert D Howe. The highly adaptive sdm hand: Design and performance evaluation. *The international journal of robotics research*, 29(5): 585–597, 2010.

Etienne Ferre and Jean-Paul Laumond. An iterative diffusion algorithm for part disassembly. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 3149–3154. IEEE; 1999, 2004.

Jonathan Fink, M Ani Hsieh, and Vijay Kumar. Multi-robot manipulation via caging in environments with obstacles. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1471–1476. IEEE, 2008.

Jonathan Fink, Nathan Michael, Soonkyum Kim, and Vijay Kumar. Planning and control for cooperative manipulation and transportation with aerial robots. *The International Journal of Robotics Research*, 30(3):324–334, 2011.

Augustin Fruchard. Small holding circles. *Geometriae Dedicata*, 157(1):397–405, 2012.

Kevin C Galloway, Jonathan E Clark, Mark Yim, and Daniel E Koditschek. Experimental investigations into the role of passive variable compliant legs for dynamic robotic locomotion. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1243–1249. IEEE, 2011.

Kyle Gilpin, Keith Kotay, Daniela Rus, and Iuliu Vasilescu. Miche: Modular shape formation by self-disassembly. *The International Journal of Robotics Research*, 27(3-4):345–372, 2008.

K Gopal Gopalakrishnan and Ken Goldberg. Gripping parts at concave vertices. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1590–1596. IEEE, 2002.

Dan Halperin, J-C Latombe, and Randall H Wilson. A general framework for assembly planning: The motion space approach. *Algorithmica*, 26(3-4):577–601, 2000.

Hideo Hanafusa, Haruhiko Asada, et al. Stable prehension by a robot hand with elastic fingers. In *Proc. 7th Int. Symp. Industrial Robots*, pages 361–368, 1977.

Ross L Hatton and Howie Choset. Generating gaits for snake robots: annealed chain fitting and keyframe wave extraction. *Autonomous Robots*, 28(3):271–281, 2010.

Luiz S Homem de Mello and Arthur C Sanderson. And/or graph representation of assembly plans. *Robotics and Automation, IEEE Transactions on*, 6(2):188–199, 1990.

Kazuo Hosokawa, Takehito Tsujimori, Teruo Fujii, Hayato Kaetsu, Hajime Asama, Yoji Kuroda, and Isao Endo. Self-organizing collective robots with morphogenesis in a vertical plane. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 4, pages 2858–2863. IEEE, 1998.

W Stamps Howard and Vijay Kumar. On the stability of grasped objects. *Robotics and Automation, IEEE Transactions on*, 12(6):904–917, 1996.

K. Hsiao and T. Lozano-Perez. Imitation learning of whole-body grasps. *IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS)*, 2006.

Nicolas Hudson, Thomas Howard, Jeremy Ma, Abhinandan Jain, Max Bajracharya, Steven Myint, Calvin Kuo, Larry Matthies, Paul Backes, Paul Hebert, et al. End-to-end dexterous manipulation with deliberate interactive estimation. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2371–2378. IEEE, 2012.

H.N. Iben, J.F. O'Brien, and E.D. Demaine. Refolding planar polygons. *Discrete & Computational Geometry*, 41(3):444–460, 2009.

SC Jacobsen, EK Iversen, D Knutti, R Johnson, and K Biggers. Design of the utah/mit dextrous hand. In *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, volume 3, pages 1520–1532. IEEE, 1986.

Aaron M Johnson and Daniel E Koditschek. Toward a vocabulary of legged leaping. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2568–2575. IEEE, 2013.

Lydia E Kavraki and Mihail N Kolountzakis. Partitioning a planar assembly into two connected parts is np-complete. *Information Processing Letters*, 55(3):159–165, 1995.

Heedong Ko and Kunwoo Lee. Automatic assembling procedure generation from mating conditions. *Computer-Aided Design*, 19(1):3–10, 1987.

Kazuhiro Kosuge, Yasuhisa Hirata, Hajime Asama, Hayato Kaetsu, and Kuniaki Kawabata. Motion control of multiple autonomous mobile robots handling a large object in coordination. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 4, pages 2666–2673. IEEE, 1999.

Keith Kotay, Daniela Rus, Marsette Vona, and Craig McGray. The self-reconfiguring robotic molecule: Design and control algorithms. In *Workshop on Algorithmic Foundations of Robotics*, pages 376–386. Citeseer, 1998.

G Kragten, Mathieu Baril, Clement Gosselin, and J Herder. Stable precision grasps by underactuated grippers. *Robotics, IEEE Transactions on*, 27(6):1056–1066, 2011.

C Ronald Kube and Eric Bonabeau. Cooperative transport by ants and robots. *Robotics and autonomous systems*, 30(1):85–101, 2000.

Haruhisa Kurokawa, Satoshi Murata, Eiichi Yoshida, Kohji Tomita, and Shigeru Kokaji. A 3-d self-reconfigurable structure and experiments. In *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, volume 2, pages 860–865. IEEE, 1998.

K Lakshminarayana. *Mechanics of form closure*. 1978.

J-C Latombe, RH Wilson, and F Cazals. Assembly sequencing with toleranced parts. *Computer-Aided Design*, 29(2):159–174, 1997.

S.M. LaValle. *Planning algorithms*. Cambridge university press, 2006.

Duc Thanh Le, Juan Cortés, and Thierry Siméon. A path planning approach to (dis) assembly sequencing. In *Automation Science and Engineering, 2009. CASE 2009. IEEE International Conference on*, pages 286–291. IEEE, 2009.

Kevin Lipkin, Isaac Brown, Aaron Peck, Howie Choset, Justine Rembisz, Philip Gianfortoni, and Allison Naaktgeboren. Differentiable and piecewise differentiable

gaits for snake robots. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1864–1869. IEEE, 2007.

Tomas Lozano-Perez. The design of a mechanical assembly system. Technical report, Massachusetts Institute of Technology, 1976.

Hiroshi Maehara. On the diameter of a circle to hold a cube. In *Computational Geometry, Graphs and Applications*, pages 147–153. Springer, 2011.

X. Markenscoff, L. Ni, and C.H. Papadimitriou. The geometry of grasping. *The International Journal of Robotics Research*, 9(1):61–74, 1990.

Matthew T Mason. *Mechanics of robotic manipulation*. MIT press, 2001.

Matthew T. Mason and J. Kenneth Salisbury. *Robot Hands and the Mechanics of Manipulation*. MIT Press, Cambridge, MA, 1985.

Matthew T Mason, Alberto Rodriguez, Siddhartha S Srinivasa, and Andrés S Vazquez. Autonomous manipulation with a general-purpose simple hand. *The International Journal of Robotics Research*, 31(5):688–703, 2012.

Margaret A McDowell, National Center for Health Statistics (US), et al. *Anthropometric reference data for children and adults: United States, 2003-2006*. US Department of Health and Human Services, Centers for Disease Control and Prevention, National Center for Health Statistics, 2008.

Andrew T Miller, Steffen Knoop, Henrik I Christensen, and Peter K Allen. Automatic grasp planning using shape primitives. In *Robotics and Automation, 2003. Proceedings. IEEE International Conference on*, volume 2, pages 1824–1829, 2003.

Gustavo Montemayor and John T Wen. Decentralized collaborative load transport by multiple robots. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 372–377. IEEE, 2005.

Antonio Morales, Pedro J Sanz, and Angel P Del Pobil. Vision-based computation of three-finger grasps on unknown planar objects. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 2, pages 1711–1716. IEEE, 2002.

Heiko Mosemann, F Rohrdanz, and F Wahl. Assembly stability as a constraint for assembly sequence planning. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 1, pages 233–238. IEEE, 1998.

Satoshi Murata, Haruhisa Kurokawa, and Shigeru Kokaji. Self-assembling machine. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 441–448. IEEE, 1994.

Richard M Murray. Trajectory generation for a towed cable system using differential flatness. In *IFAC world congress*, pages 395–400, 1996.

Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC, 1994.

J R Napier. The prehensile movements of the human hand. *The Journal of Bone and Joint Surgery*, 38-B(4):902–913, 1956.

Balas K Natarajan. On planning assemblies. In *Proceedings of the fourth annual symposium on Computational geometry*, pages 299–308. ACM, 1988.

National Concrete Masonry Association. Concrete masonry bond patterns. *TEK 14-6*, 2004.

David Navarro-Alarcon, Yun-hui Liu, Jose Guadalupe Romero, and Peng Li. On the visual deformation servoing of compliant objects: Uncalibrated control methods and experiments. *The International Journal of Robotics Research*, 2014.

Van-Duc Nguyen. Constructing force-closure grasps. *The International Journal of Robotics Research*, 7(3):3–16, 1988.

Van-Due Nguyen. Constructing stable grasps. *The International Journal of Robotics Research*, 8(1):26–37, 1989.

Ian O'Hara, James Paulos, Jay Davey, Nick Eckenstein, Neel Doshi, Tarik Tosun, Jonathan Greco, Jungwon Seo, Matt Turpin, Vijay Kumar, and Mark Yim. Self-assembly of a swarm of autonomous boats into floating structures. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, June 2014.

Yaron Ostrovsky-Berman and Leo Joskowicz. Relative position computation for assembly planning with planar toleranced parts. *The International Journal of Robotics Research*, 25(2):147–170, 2006.

Guilherme AS Pereira, Mario FM Campos, and Vijay Kumar. Decentralized algorithms for multi-robot manipulation via caging. *The International Journal of Robotics Research*, 23(7-8):783–795, 2004.

M Peshkin and A Sanderson. Reachable grasps on a polygon: the convex rope algorithm. *Robotics and Automation, IEEE Journal of*, 2(1):53–58, 1986.

Kirstin Petersen, Radhika Nagpal, and Justin Werfel. Termes: An autonomous robotic system for three-dimensional collective construction. *Proc. Robotics: Science & Systems VII*, 2011.

Nancy S Pollard. Closure and quality equivalence for efficient synthesis of grasps from examples. *The International Journal of Robotics Research*, 23(6):595–613, 2004.

Jean Ponce. On planning immobilizing fixtures for three-dimensional polyhedral parts. In *IEEE International Conference on Robotics and Automation*, pages 509–514. Citeseer, 1996.

Jean Ponce, Steve Sullivan, Attawith Sudsang, Jean-Daniel Boissonnat, and Jean-Pierre Merlet. On computing four-finger equilibrium and force-closure grasps of

polyhedral objects. *The International Journal of Robotics Research*, 16(1):11–35, 1997.

Sourav Rakshit and Srinivas Akella. The influence of motion paths and assembly sequences on the stability of assemblies. *IEEE Transactions on Automation Science and Engineering*, PP, 2014.

E. Rimon and A. Blake. Caging planar bodies by one-parameter two-fingered gripping systems. *The International Journal of Robotics Research*, 18(3):299–318, 1999.

E. Rimon and J. W. Burdick. Mobility of bodies in contact - part I: A 2nd-order mobility index for multiple-finger grasps. *Robotics and Automation, IEEE Transactions on*, 14:696 – 708, 1998a.

E. Rimon and J. W. Burdick. Mobility of bodies in contact - Part II: How forces are generated by curvature effects. *Robotics and Automation, IEEE Transactions on*, 14:709 – 717, 1998b.

A. Rodriguez and M. Mason. Two finger caging: squeezing and stretching. *Algorithmic Foundation of Robotics VIII*, pages 119–133, 2009.

A. Rodriguez, M.T. Mason, and S. Ferry. From caging to grasping. *The International Journal of Robotics Research*, 31(7):886–900, 2012.

John W Romanishin, Kyle Gilpin, and Daniela Rus. M-blocks: Momentum-

driven, magnetic modular robots. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4288–4295. IEEE, 2013.

Bruce Romney. Atlas: An automatic assembly sequencing and fixturing system. In *Geometric Modeling: Theory and Practice*, pages 397–415. Springer, 1997.

Daniela Rus and Marsette Vona. Crystalline robots: Self-reconfiguration with compressible unit modules. *Autonomous Robots*, 10(1):107–124, 2001.

Uluc Saranli, Martin Buehler, and Daniel E Koditschek. Rhex: A simple and highly mobile hexapod robot. *The International Journal of Robotics Research*, 20(7): 616–631, 2001.

Jimmy Sastra, Sachin Chitta, and Mark Yim. Dynamic rolling for a modular loop robot. *The International Journal of Robotics Research*, 28(6):758–773, 2009.

Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, 27(2):157–173, 2008.

Doris Schattschneider. The plane symmetry groups: Their recognition and notation. *The American Mathematical Monthly*, 85(6):pp. 439–450, 1978. ISSN 00029890. URL `http://www.jstor.org/stable/2320063`.

J. Seo and V. Kumar. Spatial, bimanual, whole-arm grasping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

Jungwon Seo, Soonkyum Kim, and Vijay Kumar. Planar, bimanual, whole-arm grasping. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

Jungwon Seo, Mark Yim, and Vijay Kumar. Assembly planning for planar structures of a brick wall pattern with rectangular modular robots. In *IEEE International Conference on Automation Science and Engineering (CASE)*, 2013a.

Jungwon Seo, Mark Yim, and Vijay Kumar. Restraining objects with curved effectors and its application to whole-arm grasping. In *International Symposium of Robotics Research (ISRR)*, 2013b.

Michael Sfakiotakis and Dimitris P Tsakiris. Biomimetic centering for undulatory robots. *The International Journal of Robotics Research*, 26(11-12):1267–1282, 2007.

Koushil Sreenath and Vijay Kumar. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. In *Robotics: Science and Systems (RSS)*, 2013. This paper won the RSS Best Paper Award 2013.

Thomas Sugar and Vijay Kumar. Multiple cooperating mobile manipulators. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 2, pages 1538–1543. IEEE, 1999.

Sujay Sundaram, Ian Remmler, and Nancy M Amato. Disassembly sequencing using a motion planning approach. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1475–1480. IEEE, 2001.

Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, 1972.

Ulrike Thomas, Mark Barrenscheen, and Friedrich M Wahl. Efficient assembly sequence planning using stereographical projections of c-space obstacles. In *Assembly and Task Planning, 2003. Proceedings of the IEEE International Symposium on*, pages 96–102. IEEE, 2003.

J.C. Trinkle, J. M. Abel, and R. P. Paul. An investigation of frictionless enveloping grasping in the plane. *International Journal of Robotics Research*, 7(3):33–51, June 1988.

Jeffrey C Trinkle. On the stability and instantaneous velocity of grasped frictionless objects. *Robotics and Automation, IEEE Transactions on*, 8(5):560–572, 1992.

Nathan Ulrich, Richard Paul, and Ruzena Bajcsy. A medium-complexity compliant end effector. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 434–436. IEEE, 1988.

M. Vahedi and A.F. van der Stappen. Caging polygons with two and three fingers. *The International Journal of Robotics Research*, 27(11-12):1308–1324, 2008a.

M. Vahedi and A.F. van der Stappen. On the complexity of the set of three-finger caging grasps of convex polygons. In *Proceedings of Robotics: Science and Systems*, 2009.

Mostafa Vahedi and A Frank van der Stappen. Towards output-sensitive computation of two-finger caging grasps. In *Automation Science and Engineering, 2008. CASE 2008. IEEE International Conference on*, pages 73–78. IEEE, 2008b.

Mostafa Vahedi and A Frank van der Stappen. Caging convex polygons with three fingers. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1777–1783. IEEE, 2008c.

A Frank Van der Stappen, Chantal Wentink, and Mark H Overmars. Computing form-closure configurations. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 3, pages 1837–1842. IEEE, 1999.

Serguei Vassilvitskii, Mark Yim, and John Suh. A complete, local and parallel reconfiguration algorithm for cube style modular robots. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 1, pages 117–122. IEEE, 2002.

Weiwei Wan, Rui Fukui, Masamichi Shimosaka, Tomomasa Sato, and Yasuo Kuniyoshi. On the caging region of a third finger with object boundary clouds and two given contact positions. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4154–4161. IEEE, 2012.

Justin Werfel and Radhika Nagpal. Three-dimensional construction with mobile robots and modular blocks. *The International Journal of Robotics Research*, 27 (3-4):463–479, 2008.

Paul J White, Chris E Thorne, and Mark Yim. Right angle tetrahedron chain externally-actuated testbed (ratchet): A shape changing system. In *ASME International Design Engineering Technical Conference and Computers and Information in Engineering Conference (IDETC/CIE)*, volume 7, pages 807–817, 2009.

PJ White, Kris Kopanski, and Hod Lipson. Stochastic self-reconfigurable cellular robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

Randall H Wilson and Jean-Claude Latombe. Geometric reasoning about mechanical assembly. *Artificial Intelligence*, 71(2):371–396, 1994.

Randall H Wilson and J-F Rit. Maintaining geometric dependencies in an assembly planner. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 890–895. IEEE, 1990.

M Yim, B Shirmohammadi, J Sastra, M Park, M Dugan, and CJ Taylor. Towards robotic self-reassembly after explosion. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 2767–2772. IEEE, 2007a.

Mark Yim. *Locomotion with a Unit-Modular Reconfigurable Robot*. PhD thesis, Stanford University, 1994.

Mark Yim, David G Duff, and Kimon D Roufas. Polybot: a modular reconfigurable robot. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 1, pages 514–520. IEEE, 2000.

Mark Yim, Ying Zhang, John Lamping, and Eric Mao. Distributed control for 3d metamorphosis. *Autonomous Robots*, 10(1):41–56, 2001.

Mark Yim, Wei-Min Shen, Behnam Salemi, Daniela Rus, Mark Moll, Hod Lipson, Eric Klavins, and Gregory S Chirikjian. Modular self-reconfigurable robot systems [grand challenges of robotics]. *Robotics & Automation Magazine, IEEE*, 14 (1):43–52, 2007b.

Mark Yim, Paul J. White, Michael Park, and Jimmy Sastra. Modular self-reconfigurable robots. In Robert A. Meyers, editor, *Encyclopedia of Complexity and Systems Science*, pages 5618–5631. Springer, 2009. ISBN 978-0-387-75888-6.

Ryo Yokoi, Tatsuya Kobayashi, and Yusuke Maeda. 2d caging manipulation by robots and walls. In *Assembly and Manufacturing, 2009. ISAM 2009. IEEE International Symposium on*, pages 16–21. IEEE, 2009.

T Zamfirescu. How to hold a convex body? *Geometriae Dedicata*, 54(3):313–316, 1995.