INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®
RICE UNIVERSITY

Part Assembly
Using Static and Dynamic Force Fields

by

Jiangchun Luo

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Science

APPROVED, THESIS COMMITTEE:

[Signatures and names]

Dr. Lydia Kavraki
Professor of Computer Science

Dr. Ronald Goldman
Professor of Computer Science

Dr. Liliana Borcea
Professor of Computational and Applied Mathematics

Houston, Texas

June, 2000
Abstract

Part assembly is an important goal of part manipulation. Among other techniques, programmable force fields have been introduced for part manipulation. Many different force fields have been proposed to manipulate a part, such as the squeeze and the elliptical fields. For part assembly, more than one part needs to be manipulated. This can create problems since there can be interactions between the parts such as impact and friction. With technology developing, certain fields can be implemented in the micro-scale with MEMS actuator arrays and in the macro-scale with arrays of directed air jets or small motors. Modern technology is beginning to provide the means to control the magnitude and frequency of each actuator of the implemented force field. Thus dynamic and localized force fields can be used for part manipulation.

This thesis presents a novel strategy for assembling two parts with a sequence of static and dynamic programmable force fields. The strategy involves some initial sensing. The choices of the force fields are discussed extensively. Uncertainties occurring in the motion of the parts are taken into account to make the proposed strategy more robust. This process does not make any assumption about the shape of the assembled parts, thus any pair of parts can be assembled by our strategy. We have implemented two simulators: one mimics the motion of a part under a static force field and the other mimics the part assembly process using static and dynamic force fields.
Acknowledgments

It has been my great pleasure to have Professor Lydia Kavraki be my advisor. I could not have finished this research without her thoughtful advice and tireless help. I learned a lot from her erudite knowledge, creativity, integrity and strictness. Her sharp feeling about new topics always impresses me. I wish to express my special thanks to Professor Lydia Kavraki.

It has been a lot of fun to work in the Physical Computing Group. We have had a “cookie-meeting” every Tuesday. People exchange ideas and actively discuss diverse issues during these meetings. I still can remember some of our discussions, which help me both in study and in life. Thanks to all members in the physical computing group, in particular, Dr. Oliver Brock, Rusty Holleman, Tim Danner, Elliot Anshelevich, and Miguel Teodoro.

I would like to thank Professor Karl Böhringer, Professor Bruce Donald, Professor Jean-Claude Latombe, Dr. Florent Lamiraux and Dr. Attawith Sudsang for their fruitful discussions. Also, thanks to Scott Owens with whom I collaborated in developing a simulator of static force fields. Dr. Jan Hewitt was my English teacher and she offered a lot of help for my English writing. Thanks to Chen Ding, Ming Zhang and Zoran Budimlic for their kind help with my thesis writing.

Thanks to my thesis committee, Professor Ronald Goldman and Professor Liliana Borcea for their helpful comments.

Finally, I thank my family for being a constant source of strength, love and support. My girlfriend, Yingqin Jing, always encourages me whenever I feel frustrated by the difficulties of research.

Most of all I thank my Mom and Dad, who have given me everything and to whom I owe everything. Mom, this is for you!

This work is supported by NSF IRI-970228, NSF CISE SA1728-21122N and a Sloan Fellowship to Lydia Kavraki.
Contents

List of Illustrations vii

1 Introduction 1

1.1 Related Work 1

1.1.1 Assembly Sequencing 1

1.1.2 Mechanical Devices Plus Algorithms 2

1.1.3 Part Manipulation Using Programmable Force Fields 3

1.2 Problem Description 6

1.3 Brief Outline of Our Approach 8

1.3.1 Basic Strategy to Assemble Two Parts 8

1.4 Thesis Outline 9

2 Basic Dynamics and Damping Model 11

2.1 Basic Dynamics for Planar Motion 11

2.2 Damping Force and Torque 12

2.2.1 Unsuitability of Coulomb Friction 13

2.2.2 Viscous Damping 15

3 Part Assembly with Force Fields 17

3.1 Centering Parts Using Radial Fields 18

3.1.1 Part Translation under Radial Fields 18

3.1.2 Lack of Rotation under Radial Fields 19

3.2 Rotating Parts Using Dynamic Curl Fields 21

3.2.1 Why Use Dynamic Curl Fields 21
3.2.2 Lack of Translation under Curl Fields ........................................ 22
3.2.3 Rotation Control Using Curl Fields ........................................... 23
3.2.4 Flexibility of Curl Fields .......................................................... 25
3.3 Assembling Parts Using Dynamic Push Fields ................................. 26
3.3.1 Translation Control Using Push Fields ...................................... 27
3.4 Fine Control Relaxation ............................................................... 30

4 Uncertainties in Part Assembly ....................................................... 32
  4.1 Related Background From Robotics ............................................. 32
  4.2 Involved Uncertainties .............................................................. 33
  4.3 Assembly Handling Uncertainties .............................................. 35
    4.3.1 C-obstacle Calculation ....................................................... 35
    4.3.2 Goal Kernel and Preimage Calculation ................................ 36

5 Programmable Force Field Simulators and Experimental Results ......... 40
  5.1 A Simulator for Static Force Fields ........................................... 40
    5.1.1 Introduction ........................................................................... 40
    5.1.2 User Interface ....................................................................... 41
    5.1.3 Force and Torque Calculation .............................................. 42
    5.1.4 Coordinate Frames ............................................................... 43
    5.1.5 Features ................................................................................ 44
  5.2 A Simulator for Part Assembly .................................................... 45
  5.3 Experimental Results ................................................................. 46
    5.3.1 Simulation Results of Static Force Fields ................................. 46
    5.3.2 Simulation Results of Part Assembly .................................... 48

6 Conclusion and Future Work .......................................................... 49
A Implementation Issues Concerning the Simulators 52

A.1 Force Fields .................................................. 52
A.2 Parts ............................................................ 52
  A.2.1 File Format for Polygonal Parts ......................... 52
  A.2.2 Algorithm of Polygon Filling ............................. 53
  A.2.3 class Part .................................................. 54
A.3 Other ............................................................. 54

Bibliography 55
Illustrations

1.1 (a) is an elliptical force field with $\alpha = 1$ and $\beta = 2$. (b) is the corresponding elliptical potential field. ........................................... 6

1.2 Snapshots along a part assembly simulation. (a) shows the initial positions and orientations of two parts before assembly. (b) shows the parts after centering with radial fields. (c) shows curl force fields applying to the parts. (d) shows the parts with the assembled orientations after rotation. (e) shows a push force field applied to one part. (f) shows the assembled parts. ........................................... 7

1.3 Algorithm of part assembly. ........................................... 9

2.1 $\mathbf{G}$ is the gravity force of $P$, $\mathbf{N}$ is the normal force, $\mathbf{F}$ is the external force and $\mathbf{F}_d$ is the damping force. ........................................... 11

2.2 Coulomb friction changes with the external driving force. ........ 13

3.1 A radial force field with $\beta = 1$. ........................................... 18

3.2 Distance of $P$ vs. time ........................................... 20

3.3 A curl force field with $\beta = 1$. ........................................... 22

3.4 (a) is a symmetric pair of parts while (b) is an asymmetric pair of parts. 26

3.5 A push field with $\phi = -\frac{\pi}{4}$ and $\gamma = 1$. ........................................... 27

4.1 Three kinds of uncertainties: (a) shows position uncertainty. (b) shows the orientation uncertainty. (c) shows the velocity uncertainty. 34
4.2 An algorithm that deals with uncertainties in part assembly. .......... 35
4.3 C-obstacle computation algorithm. $\text{vert}(\mathcal{A})$ is the vertex set of $\mathcal{A}$ and
$\ominus$ is Minkowski difference. ............................................. 36
4.4 A method to calculate the box containing the robots with all the
possible orientations when the orientation uncertainty is $\phi$. .......... 37
4.5 (a) is a rectangular robot. (b) shows a box to contain the robot with
all possible orientations when the uncertainty $\phi = 0.1$. ............... 37
4.6 The goal region $\mathcal{T}$ is a rectangle in a configuration space. The goal
kernel $\mathcal{K}$ is obtained by shrinking $\mathcal{T}$ by the radius of the uncertainty
disk. ................................................................. 38
4.7 Backprojection calculation algorithm. ................................. 38
4.8 (a) is the same box as shown in Figure 4.5 (b). (b) is the inserted
part(obstacle in this case). (c) shows the C-obstacle(the light line),
the goal region(the area below the horizontal black solid line), the
kernel of the goal region(the area below the dotted line) and the
preimage of the goal kernel(the thick line). ................................ 39

5.1 Two different parts reaching equilibrium in the elliptic force field
$f(x, y) = (-4x, -2y)$ using the quasi-static model. ...................... 41

A.1 A file for a polygonal part. ............................................. 53
Notation

$P$  
a part
$W$  
a force field
$COM$  
the center of mass of $P$
$\rho$  
the mass density of $P$
$M$  
mass of $P$
$S$  
the area of $P$
$I_z$  
the moment of inertia of $P$ about the $z$ axis
$xoy$  
the coordinate frame for a force field
$XOY$  
the local coordinate frame with the $COM$ of $P$ as the origin
$r = (X, Y)^T$  
the vector from the $COM$ to a point $(X, Y)^T$
$(x_c, y_c)^T$  
the coordinates of the $COM$ in $xoy$ frame
$f(x, y)$  
the force at a point $(x, y)^T$ from $W$
$F = \int_S f ds$  
the external force applied on $P$ from $W$
$T = \int_S r \times f ds$  
the external torque applied on $P$ from $W$
$f_d$  
the damping force at a point
$F_d = \int_S f_d ds$  
the damping force applied on $P$
$T_d = \int_S r \times f_d ds$  
the damping torque applied on $P$
$v$  
the translational velocity of $P$
$a$  
the translational acceleration of $P$
$\omega$  
the angular velocity of $P$
$\alpha$  
the angular acceleration of $P$
$\mathbf{v}_c$  
the translational velocity of the $COM$ of $P$
$\mathbf{a}_c$  
the translational acceleration of the $COM$ of $P$
$\beta$  
the magnitude of a curl force field
$\gamma$  
the magnitude of a push force field
$\delta$  
the size difference rate
$A$  
a robot
$B$  
a obstacle
$CB$  
a C-obstacle
Chapter 1

Introduction

Part assembly is used to construct a product from its individual parts. This task is critical in industrial automation since it strongly affects the efficiency of product manufacturing. An assembly is described by a geometric model of its parts and their relative placement [24]. To assemble two parts, first we need to understand how to manipulate each part individually. Several techniques and devices have been proposed for part manipulation, some of them involving sensing (e.g., [2]) and others requiring no sensing (e.g., [15]). In modern manufacturing, parts are massively produced in parallel and many of them are small and fragile. The use of programmable force fields is a new but rapidly developing solution for part manipulation. This research concentrates on methods that use programmable force fields and proposes a novel strategy to assemble two parts, which requires an initial sensing operation.

1.1 Related Work

1.1.1 Assembly Sequencing

Assembly sequencing specifies a partial ordering of operations to construct a product from its individual parts [24]. Each operation generates a new subassembly by merging individual parts and/or subassemblies constructed by previous operations. Extensive research on assembly sequencing has been done during the last ten years [4, 13, 16, 17, 18, 22, 24, 25, 32, 33, 34, 35]. Early assembly sequencers were mainly interactive sequence editors; geometric reasoning was supplied by a human who answered questions asked by the system [13]. Automated geometric reasoning was then added to answer some of these questions [3, 4, 18, 32, 35]. This develop-
ment first resulted in generate-and-test sequencers, with a module guessing candidate sequences and geometric reasoning modules checking their feasibility [18, 34]. More efficient techniques were later proposed to replace time-consuming generate-and-test [3, 32]. The later techniques used computational geometry and clever data structures to automatically decide feasible assembly sequences. The automatic generation of an assembly sequence has been shown to be a PSPACE-hard problem [30, 35]. Even if every operation merges exactly two subassemblies/parts into their final relative configurations, the problem is still NP-complete [21].

1.1.2 Mechanical Devices Plus Algorithms

When an assembly sequence is implemented in real manufacturing, at each operation of the sequence people need devices (e.g., a vibratory bowl feeder) to manipulate the assembled parts. In order to reach an overall high efficiency and flexibility, each operation has to be efficient and flexible to manipulate different parts. Traditional part manipulation devices such as vibratory bowl feeders are usually designed for manipulating a single part and need to be redesigned if the shape of the part changes. It is time-consuming to redesign a bowl feeder. Recent work has explored more efficient and flexible techniques to manipulate parts, e.g., [1, 6, 8, 9, 15, 27]. Programmable part feeders offer high flexibility and can be adapted easily to different parts [1, 15]. It is also desirable to use programmable feeders that are robust and easy to implement. Techniques that require no sensing are favored [6, 15]. A few different approaches to programmable part feeders are listed below.

Goldberg proposed an algorithm which can orient a polygonal part by a sequence of squeezes performed by a pair of parallel jaw grippers [15]. A part is polygonal if the part's convex hull is a polygon. Given a list of $n$ vertices of a polygonal part whose initial orientation is unknown, this algorithm can find the shortest sequence of squeezes that uniquely orients the part up to symmetry in its convex hull in $O(n^2 \log n)$ time. This algorithm requires no sensing.
Akella et al. presented a method for manipulating a planar rigid part on a conveyor belt using a robot with one joint [1]. For a given sequence of edges of a polygonal part, the planner can compute a feasible sequence of jogs and turns to feed the part by resolving a nonlinear programming problem. Any polygonal part is controllable from a broad range of initial configurations to any goal chosen from a broad range of goal configurations. If the fence and the conveyor are large enough, such a feasible sequence always exists.

Böhringer et al. manipulate planar parts by vibrating a contact plane [6]. Depending on the frequency of vibration and the boundary condition, a part is moved towards one of serial stable configurations in the contact plane. The underlying dynamics that moves the part give rise to an effective force field. This manipulation does not use any sensing.

1.1.3 Part Manipulation Using Programmable Force Fields

Some programmable part feeders have high flexibility, but they are generally designed to manipulate large and rigid parts. With the development of modern manufacturing, parts become small and fragile. The use of programmable force fields is a new technique suitable for manipulating such small and fragile parts. The basic idea is the following: the force field is realized on a planar surface on which the part is placed; the force and torque applied on the part translate and rotate it. The use of programmable force fields becomes an interesting research issue for the following reasons. First, programmable force fields can be used for a wide variety of parts and require little or no sensing. They employ simple devices and are rather robust. Second, for some small parts, it is really difficult to build a conventional mechanical device to control them. Last, unlike mechanical devices such as grippers, force fields do not damage the parts. When using a mechanical gripper, one often needs to consider mechanical failures (such as stress concentration, wreck and fatigue) of the parts during manipulation.
1.1.3.1 Force Field Implementations

Leading researchers have started implementing certain force fields with different technologies. Böhringer et al. have implemented the force fields with MEMS (Micro Electrical Mechanical System) actuator arrays in micro scale [7, 9]. Their devices exploit arrayed massively-parallel micro fabricated motion pixels. By controlling the motion of the pixels, each pixel can apply a small force on the part above it. Thus a discretized programmable force fields can be realized on the surface of the arrayed pixels.

Yim and Berlin have implemented contactless manipulation using arrays of directed air jets [36]. The air jet system realizes macro-scale distributed manipulation by using the idea of micro actuation. Directed air jets can levitate and transport planar parts without contacting them. By controlling the magnitude and frequency of each air jet, the whole arrays of air jets can generate a discretized force field for the purpose of part manipulation.

Luntz et al. manipulate parts on the plane using a macroscopic actuator array, MDMS (Modular Distributed Manipulator System) [26, 27]. Each actuator cell is composed of a pair of orthogonally oriented motorized roller wheels. By rotating the two wheels, if a part contacts with the two wheels at a moment, a pair of orthogonal friction forces will apply on the part. The discretized force fields generated by MDMS can be controlled by selecting the appropriate rotational velocity of each wheel.

Reznik and Canny use vibrating surfaces to manipulate parts in the plane by displaying arbitrary forces at a number of points [31]. Under a rigid and closed motion of the surface, the desired forces at some points and/or the curl about some points on the plane can be generated basing on the time-averaging of Coulomb friction.

1.1.3.2 Algorithmic Study of Programmable Force Fields

Extensive algorithmic studies have been done for manipulating a single part using force fields in the past few years. Donald and Böhringer pioneered this area [6, 7, 9].
In recent papers many kinds of force fields have been proposed. One interesting class of force fields are conservative force fields. A force is conservative if the work the force does on a particle that moves through a round trip is zero. A potential energy distribution can be associated with a conservative force field. We also call the conservative force fields the potential fields. Böhringer et al. use a sequence of squeeze force fields to uniquely orient a polygonal part up to symmetry in its convex hull [8, 9, 27]. Kavraki presents an elliptical force field which can uniquely pose a part and orient an asymmetric part to one of its two stable orientations [8, 19, 27]. Lamiraux and Kavraki show that a radial plus gravity force field can uniquely pose and orient a part whose center of mass is different from its pivot point [8, 20].

**An Example of Potential Fields** Consider a force field of the type [19]

\[
f(x, y) = (-\alpha x, -\beta y),
\]

(1.1)

where \(\alpha\) and \(\beta\) are two distinct positive constants. This is a called elliptical force field. Figure 1.1 (a) shows an elliptical force field with \(\alpha = 1\) and \(\beta = 2\). When moving a particle from the origin of the field to a point \((x, y)^T\), the work done by the force field is \(\frac{\alpha}{2} x^2 + \frac{\beta}{2} y^2\). If the origin is selected as the zero potential point, the potential energy at \((x, y)^T\) is \(\frac{\alpha}{2} x^2 + \frac{\beta}{2} y^2\). Figure 1.1 (b) shows the potential distribution over the elliptical force field shown in 1.1(a).

**Equilibrium Configurations under Potential Fields** An important merit of potential fields is that these fields can translate and rotate a part to stable equilibrium configurations. For a part to be in equilibrium, the force applied on the part is required to be zero and the torque about, say, the COM (center of mass) of the part must be zero. Suppose the force distribution at a point \((x, y)^T\) is \(f(x, y)\). The equilibrium conditions are

\[
\int_s f(x, y) ds = 0,
\]

(1.2)
Figure 1.1: (a) is an elliptical force field with $\alpha = 1$ and $\beta = 2$. (b) is the corresponding elliptical potential field.

$$\int_S \mathbf{r} \times f(x, y) \, ds = 0, \quad (1.3)$$

where $S$ is the area occupied by the part and $\mathbf{r}$ is the vector from the COM to the point $(x, y)^T$. For example, for an elliptical potential field, the equilibrium position of any part is the origin of the field [19].

The application of force fields is becoming more and more popular. Most of the current research concentrates on manipulating one part. This thesis shows how to use force fields to assemble parts.

1.2 Problem Description

This thesis focuses on the following problem. Consider two planar parts on a plane. The problem is to find a sequence of force fields to assemble them, see Figure 1.2. There are three basic assumptions in our work:

- Assumption 1: one part lies in the left half plane and the other lies in the right half plane, as shown in Figure 1.2 (a). This assumption avoids the part separation problem, which could be as difficult as the part assembly problem.
- Assumption 2: we know the initial orientations of the two parts but do not know
their initial positions.

- Assumption 3: the assembly positions and orientations of the two parts are known.

Our work makes some limiting assumptions but to our knowledge this is the first step towards part assembly using programmable force fields. Assumption 2 implies that the process needs sensing to find the initial orientations of the two parts. After that, the remaining process does not require any sensing. However, sensing, if available, can be used to compensate for errors.
This study focuses on the case where one part has a convex section and the other part has a complementary concave section. We call the part with the convex section the "inserting part" $P_1$ and the other part the "inserted part" $P_2$. For example, in Figure 1.2 (a), the rectangular part is the inserting part and the concave part is the inserted part. In some assembly scenarios when both parts are convex, e.g., two rectangular parts, it makes no difference which part is the inserting part. For convenience, the smaller part is called the inserting part.

1.3 Brief Outline of Our Approach

All current force fields handle only a single part. When placing two parts in the same force field, the problem becomes much more complicated because there could be impact and friction between the parts. For the purpose of this research, we initially put one part in each half plane and apply force fields on each half plane individually to avoid interactions between the parts. One contribution of our work is the use of dynamic force fields. Dynamic force fields are time-varying; i.e., the force (both in magnitude and direction) at each point can change with the time. In certain force field realizations (e.g., air jets), the magnitude and frequency of each pixel (e.g., each air jet) can be controlled individually [36]. This inspired us to think about using dynamic force fields. All previous work has considered static fields. For example, the squeeze, elliptical, and radial plus gravity force fields described in [9, 19, 20] are all static.

1.3.1 Basic Strategy to Assemble Two Parts

In order to assemble two parts, first, they should be positioned with their specified orientations which we call assembly orientations. Then the parts are moved to their desired positions to construct the product. Figure 1.3 shows our algorithm assembling two parts using force fields.
(1) Center the two parts in their half planes with two separate radial fields; see Figure 1.2 (a) and (b).

(2) Rotate the parts to their assembly orientations with two dynamic curl fields; see Figure 1.2 (c) and (d).

(3) Translate the inserting part to complete the assembly with one or two dynamic push fields; see Figure 1.2 (e) and (f).

Figure 1.3 : Algorithm of part assembly.

Rotation and translation control are critical in the strategy given in Figure 1.3. The parts may start with any initial orientations, so the rotational force field has to be able to rotate each part to its assembly orientation. Section 3.2 will explain why none of the existing force fields that can orient a part can be used in our assembly process. Thus, we invented the dynamic curl field used in stage (2) of Figure 1.3. This study also presents a pure radial field and a dynamic push field to control part translation. Chapter 3 will show that these two force fields can translate a part without rotating it.

1.4 Thesis Outline

This thesis studies the two dimensional part assembly problem. Chapter 2 discusses the dynamics used for planar motion and the damping model used in this research. Chapter 3 explains our part assembly strategy and talks about the properties of the radial, dynamic curl and dynamic push force fields. Rotation and translation control are also discussed in this chapter. Chapter 4 discusses uncertainties involved
in part assembly and presents an algorithm to handle those uncertainties. Chapter 5 describes two implemented simulators. One is used for describing the motion of a part under a static force field; the other is used to verify the correctness of our assembly strategy. Chapter 6 concludes this thesis and discusses future work.
Chapter 2

Basic Dynamics and Damping Model

The first section of this chapter describes the dynamics used to control the motion of a part on a plane. The second section discusses the damping model used in our assembly strategy.

2.1 Basic Dynamics for Planar Motion

Suppose there is a part $P$ on a contact plane which implements a force field $W$. We assume that four kinds of forces apply on $P$. They are the gravity $G$ of $P$, the normal force $N$ to support $P$ on the plane, the external force $F$ from $W$, and the damping force $F_d$; see Figure 2.1.

![Figure 2.1](image)

Figure 2.1: $G$ is the gravity force of $P$, $N$ is the normal force, $F$ is the external force and $F_d$ is the damping force.

Suppose the force distribution of $W$ is $f(x, y)$ over the contact plane, the external force applied on $P$ can be calculated as

$$F = \int \int_S f(x, y) ds,$$

where $S$ is the area occupied by $P$. In general, the motion of $P$ involves both translation and rotation. From classical dynamics, the force applied on $P$ translates $P$, while the torque applied on $P$ rotates $P$. The torque about the COM of $P$ involves
only the torque in the z direction, which rotates P on the plane. Since both G and N are in the z direction, they apply no torque in the z direction. The external torque is calculated as

\[ \mathbf{T} = \int_S \mathbf{r} \times f(x, y) \, ds, \]  

(2.2)

where \( \mathbf{r} \) is the vector from the COM of P to the point \((x, y)^T\). The damping force and torque will be discussed in section 2.2.

This research focuses on a scenario in which \( \mathbf{G} + \mathbf{N} = 0 \); i.e., the motion of P under W is a standard planar motion. Any planar motion of P can be decomposed into a pure translation of its COM and a pure rotation about its COM. Thus at each point of P, its instaneous velocity is

\[ \mathbf{v} = \mathbf{v}_c + \omega \times \mathbf{r}, \]  

(2.3)

where \( \mathbf{v}_c \) is the translational velocity of the COM of P and \( \omega \) is the angular velocity of P. For the translation, using Newton's Second law, we get

\[ \mathbf{F} + \mathbf{F}_d = M \, \mathbf{a}, \]  

(2.4)

where M is the mass of P and \( \mathbf{a} \) is the translational acceleration.

For the rotation, there is a similar equation in dynamics:

\[ \mathbf{T} + \mathbf{T}_d = I_z \, \alpha, \]  

(2.5)

where \( \mathbf{T}_d \) is the damping torque, \( I_z \) is the moment of inertia about the z axis, and \( \alpha \) is the angular acceleration of P about its COM.

Using these two fundamental equations of dynamics, we can develop the differential equations for translation and rotation by substituting specific values for \( \mathbf{F}, \mathbf{F}_d, \mathbf{T}, \) and \( \mathbf{T}_d \). Then the trajectory and orientation function of P can be calculated.

### 2.2 Damping Force and Torque

There are few papers that discuss the damping model of programmable force fields. Without damping, no potential field can orient and pose a part: since the potential
energy is associated with the conservative force, without damping, the sum of the part's potential and kinematic energy will never decrease, which means the part will never stop if its initial energy is not minimum. Damping plays a critical role in our part assembly method. A detailed model of damping depends on the physical implementation of the force field. Due to lack of availability of specific new models, our analysis is based on existing damping models. There are three kinds of damping in physical systems [29]:

(1) Coulomb or dry friction damping,
(2) Viscous or fluid damping, and
(3) Internal or molecular damping.

We can't use the third kind of damping, since part assembly is done in the macro-scale and no molecular damping needs to be considered. The rest of this chapter will examine both Coulomb friction and viscous damping.

2.2.1 Unsuitability of Coulomb Friction

Coulomb friction is a common model for dry contact in physical systems. Figure 2.2 shows how the Coulomb friction force changes with the external driving force. The

![Figure 2.2: Coulomb friction changes with the external driving force.](image)
magnitude of the maximum static friction force applied on $P$ is of the form $F_{dm} = \mu N$, where $\mu$ is the Coulomb friction coefficient and $N$ is the magnitude of the normal force, which is equal to the gravity of $P$. Whenever the driving force is smaller than $F_{dm}$, $P$ can’t be moved. Once the driving force is larger than $F_{dm}$, $P$ starts to move and the friction force changes from static friction to dynamic friction. The magnitude of Coulomb dynamic friction obeys the same equation as $F_{dm}$ and the dynamic friction coefficient is slightly smaller than the Coulomb static friction coefficient. Because both $\mu$ and $N$ are constants, $F_{dm}$ of $P$ is constant and hence defines a threshold value for translating $P$. Any external force less than $F_{dm}$ can’t translate $P$. For example, suppose we have a part under an elliptical force field (see section 1.1.3). If the external force applied on the part is smaller than $F_{dm}$ at the part’s initial position, the part will remain at its initial position without being moved by the force field. In that case the part can’t be moved to its equilibrium position, the origin of the elliptical field.

Further consideration can make the above discussion clearer. Under a potential field there is a region $S_d$ such that the external force on $P$ is smaller than $F_{dm}$ at any point inside $S_d$. $S_d$ can be calculated from the definition of the external force. Any planar part under a given $W$ has a specific shape for $S_d$. This can be proved in the following way. Both the x-component and y-component of the external force are functions of the coordinates of the COM of $P$. So the boundary of $S_d$ can be given by the following equation:

$$\sqrt{F_x^2(x_c, y_c) + F_y^2(x_c, y_c)} = F_{dm}. \quad (2.6)$$

The above equation describes a two-dimensional curve. For example, under a radial field $f(x, y) = (-x, -y)^T$, the external force is

$$\mathbf{F}(x, y) = (-x_cS, -y_cS)^T. \quad (2.7)$$

Letting $\mathbf{F}$’s magnitude be equal to $F_{dm}$, we get

$$S \sqrt{x_c^2 + y_c^2} = F_{dm}. \quad (2.8)$$
This is a circle, whose center is the origin of the force field and radius is $\frac{F_{dm}}{S}$.

Under a potential field, with Coulomb friction, a part may stop at any position inside its $S_d$ and the final position is decided by the initial position and initial velocity of the part. Inside $S_d$, the external force at any point is smaller than $F_{dm}$. So if the instantaneous velocity of $P$ is zero at some position inside $S_d$, $P$ will remain at that position. During the motion of $P$, there is mechanical energy transfer between potential energy and kinematic energy and the energy transfer can occur in either direction. Because the friction force always does negative work, the mechanical energy of $P$ (sum of potential energy and kinematic energy) decreases with $P$'s moving. Thus, the initial mechanical energy decides the final position of $P$.

For the same reason, the Coulomb friction torque is also a threshold value for rotating $P$. Böhringer said that parts seem to reach their equilibrium configurations in experiments [7] and “even if locally we have Coulomb friction, globally we may see a different behavior” [5]. Hence the Coulomb friction model alone can’t be used to explain the behavior of parts under a force field.

### 2.2.2 Viscous Damping

A friction model that can be used in our research is viscous damping. At each point of the part, the damping force is

$$f_d = -k \cdot v,$$

(2.9)

where $k$ is the damping coefficient, which is generally a constant. Luntz and Messner use a similar friction model [27]. The damping force over the whole part is

$$F_d = \int_S -k \, v \, ds$$

$$= -k \int_S v_c \, ds - k \int_S \omega \times r \, ds$$

$$= -kSv_c - k \int_S (\omega Y \cdot \vec{i} + \omega X \cdot \vec{j}) \, ds$$

$$= -kSv_c + k\omega SY_c \cdot \vec{i} - k\omega SX_c \cdot \vec{j}$$
\[ = -kSv_c, \quad (2.10) \]

where \( r = X\hat{i} + Y\hat{j} \) and \( XOY \) is the local coordinate system whose origin is the COM of \( P \). The reason why \( k\omega SY_c \cdot \hat{i} - k\omega SX_c \cdot \hat{j} = 0 \) is that \( (X_c, Y_c)^T = (0, 0)^T \).

The damping torque over the part is:

\[
T_d = \int_S -k r \times v ds = -k \int_S r \times v_c ds - k \int_S r \times \omega \times r ds = -k \int_S (Xv_{cy} - Yv_{cx}) \cdot \vec{k} ds - k \int_S \omega(X^2 + Y^2) ds \cdot \vec{k}
\]

\[
= -\frac{k\omega}{\rho} I_z \cdot \vec{k}, \quad (2.11)
\]

where \( \rho \) is the mass density of the part. From our calculations above, we know that the damping force applied on a part is proportional to the translational velocity of its COM and the damping torque is proportional to the angular velocity about its COM. This property is important for many differential equations used in Chapter 3.

**Viscous Damping is Suitable for Force Fields**

The reason is straightforward. If at some position the part has zero velocity, there is no viscous damping force. However, if the external force at this position is not zero, the part will be moved and viscous damping appears. So \( P \) can only stop at a configuration that satisfies

\[
F = 0.
\]

\[
T = 0.
\]

This configuration is what we call the equilibrium configuration.

In physics and mechanics, friction and damping are still interesting topics for many researchers. Despite extensive studies, the mechanisms of friction and damping have not been completely understood. For the force field technique there may exist a better damping model. However, until such a new model is available, we have to work with existing models. The rest of the thesis will be based on viscous damping.
Chapter 3

Part Assembly with Force Fields

This chapter talks about our assembly strategy and discusses the properties of the selected programmable force fields. So far related research on force fields has studied the issues of part positioning and part orientations. All proposed force fields generally handle a single part. When two parts are in a field, interactions between the parts (assuming the parts do not affect the field) make manipulation much harder. To avoid these interactions, we initially put the assembled parts in distinct half planes with a force field effective only in a half plane. Our assembly procedure proceeds in three stages:

- Centering two parts in their own half planes; see Figure 1.2 (a) and (b);
- Rotating the parts to their assembly orientations; see Figure 1.2 (c) and (d);
- Pushing the inserting part to its assembly position; see Figure 1.2 (e) and (f);

As discussed in the introduction, the rotation and translation control of parts is the key to assemble two parts. We present a dynamic curl force field to control the parts’ rotation in section 3.2 and a dynamic push force field to control the inserting part’s translation in section 3.3. Before that, some preprocessing is needed to center the two parts in their own half planes. We present a radial force field for centering the parts in section 3.1. Section 3.4 discusses fine control relaxation to make our strategy more robust.

This approach uses two coordinate frames. The $xoy$ frame has the origin at the center of the half plane and is used in the force field. The $XOY$ frame is a local coordinate frame whose origin is the COM of a part and is used for the part.
3.1 Centering Parts Using Radial Fields

A radial force field is of the form:

\[ f(x, y) = (-\beta x, -\beta y), \quad (3.1) \]

where \( \beta \) is the magnitude of the radial field. Our strategy uses a radial force field with \( \beta = 1 \). Figure 3.1 shows such a radial field.

![Radial Force Field](image)

Figure 3.1: A radial force field with \( \beta = 1 \).

3.1.1 Part Translation under Radial Fields

A radial field is a potential field [8, 20]. The force applied on a part is

\[ F_x = \int_S f_x ds = -\int_S x ds = -Sx_c, \quad (3.2) \]
\[ F_y = \int_S f_y ds = -\int_S y ds = -Sy_c, \quad (3.3) \]

where \( x_c = \frac{1}{S} \int_S x ds \) and \( y_c = \frac{1}{S} \int_S y ds \) are the coordinates of the COM of \( P \). Any part has a stable translational equilibrium position at which \( F_x = 0 \) and \( F_y = 0 \). Thus the equilibrium condition is equivalent to the COM of \( P \) sitting at the origin of the radial field. In the x direction, Equation 2.4 can be written as

\[ F_x + F_{dx} = Ma_{xc}, \quad (3.4) \]
where \( a_{xc} \) is the x-component of the translational acceleration of the COM of \( P \). By substituting in \( F_x, F_{dx}, v_{zc} = \dot{x}_c \) and \( a_{xc} = \ddot{x}_c \), Equation 3.4 is a second order differential equation:

\[
\rho \ddot{x}_c + k \dot{x}_c + x_c = 0. \tag{3.5}
\]

Suppose the initial position of \( P \) is \((x_0, y_0)^T\) and the initial velocity of \( P \) is zero. We solve Equation 3.5 with the initial conditions \( \dot{x}_c(0) = 0 \) and \( x_c(0) = x_0 \):

\[
x_c(t) = e^{-\frac{kt}{2\rho}} [x_0 \cos(\lambda t) + \frac{kx_0}{4\rho - k^2} \sin(\lambda t)], \tag{3.6}
\]

where \( \lambda = \sqrt{\frac{4\rho - k^2}{2\rho}} \) is the oscillation frequency and \( \frac{k}{2\rho} \) is the attenuation frequency. The differential equation of \( y_c(t) \) is almost the same as Equation 3.5; the only difference is \( y_c(0) = y_0 \), so

\[
y_c(t) = e^{-\frac{kt}{2\rho}} [y_0 \cos(\lambda t) + \frac{ky_0}{4\rho - k^2} \sin(\lambda t)]. \tag{3.7}
\]

Dividing Equation 3.7 by Equation 3.6, we get \( \frac{y_c(t)}{x_c(t)} = \frac{y_0}{x_0} \). This proves that the trajectory of the COM of \( P \) is a straight line passing through the origin of the radial field and the initial position of \( P \). The distance of \( P \) from the origin is calculated by

\[
d(t) = \sqrt{x_c^2(t) + y_c^2(t)}
= \sqrt{x_0^2 + y_0^2 e^{-\frac{kt}{2\rho}} \left[ \cos(\lambda t) + \frac{k}{4\rho - k^2} \sin(\lambda t) \right]}. \tag{3.8}
\]

\( P \) will oscillate about the origin of the force field a few times and then stop due to \( e^{-\frac{kt}{2\rho}} \). Figure 3.2 shows how \( d(t) \) changes with time when \( k = 0.5 \) and \( \rho = 1.0 \). Also from Equation 3.8, if the two parts have the same mass density \( \rho \), they will have the same oscillation frequency \( \lambda \) and same attenuation frequency \( \frac{k}{2\rho} \), which guarantee that they reach the centers of their half planes at the same time. This is a good property, which makes the two parts ready for the next stage simultaneously.

### 3.1.2 Lack of Rotation under Radial Fields

**Claim 3.1:** If a part has zero initial angular velocity, the part can’t be rotated by a radial field.
**Proof:** From Equation 2.5, if no torque is applied on $P$, $P$ can't be rotated. The external torque about the COM of $P$ is $\mathbf{T} = \int_S \mathbf{r} \times f(r) \, ds$. In the XOY coordinate frame of $P$, $\mathbf{r} = X \cdot \mathbf{i} + Y \cdot \mathbf{j}$. Using coordinate translation, we get

\[
\begin{pmatrix}
X \\
Y
\end{pmatrix} = \begin{pmatrix}
x - x_c \\
y - y_c
\end{pmatrix}.
\]

So,

\[
\mathbf{T} = \int_S (X \cdot \mathbf{i} + Y \cdot \mathbf{j}) \times \left[ -(x_c + X) \cdot \mathbf{i} - (y_c + Y) \cdot \mathbf{j} \right] ds
\]

\[
= \int_S (x_c Y - y_c X) dX dY \cdot \mathbf{k} = x_c SY_c - y_c SX_c = 0,
\]

because $s (X_c, Y_c)^T = (0, 0)^T$. Thus, a radial field does not apply a torque on $P$. However, a damping torque may still exist:

\[
\mathbf{T}_d(t) = I_z \alpha(t),
\]

\[
\dot{\omega}(t) = -\frac{k\omega(t)}{\rho}.
\]

The general solution of the above equation is $\omega(t) = Ce^{-\frac{1}{\rho}t}$. When substituting the initial condition $\omega(0) = 0$, the solution is $\omega(t) = 0$. This means the orientation of the part will not be changed by its damping torque.
From the above analysis, the radial field can center $P$ without changing $P$'s orientation if the initial angular velocity of $P$ is zero.

3.2 Rotating Parts Using Dynamic Curl Fields

3.2.1 Why Use Dynamic Curl Fields

Since each part has an arbitrary initial orientation, we need a force field to rotate $P$ by any desired angle to a given orientation. Currently, there are three proposed potential fields which can be used to orient parts. They are the squeeze, elliptical, and radial plus gravity fields.

A polygonal part can be oriented uniquely up to symmetries with a sequence of squeeze fields [8, 9]. However, we have to select different sequences of squeeze fields for different parts. Furthermore, the parts can only be oriented up to symmetries in the convex hull of the part.

An elliptical field can orient an "asymmetric" part to one of its two stable orientations [8, 19]. If different orientations are required, we need only to rotate the main axes of the elliptical field. However, the final orientations are still not unique. Also, an elliptical force field can’t orient some symmetric parts such as squares.

The radial plus gravity force field can orient a part whose pivot point is different from its center of mass to an unique orientation [8, 20]. But there are also several problems. First, so far there is no rule for selecting the magnitude of the gravity field. Second, the calculation of the pivot point of a part under a radial field has been shown to be intractable [20]. Last, parts with the same pivot point and center of mass may have more than one final orientation under a radial plus gravity field.

Since current force fields can’t control the rotation of the part easily, this research presents a dynamic curl field to reach this goal; see Figure 3.3. Because such a curl field does a nonzero amount of work for $P$ along some closed path, the curl field is not a potential field. Under a nonconservative force field, there does not exist an equilibrium configuration of $P$, since such a nonconservative field can only move a
part but may not stop it. This characteristic requires us to use a dynamic component in order to stop a part under a nonconservative force field. Consider a dynamic force field of the form:

$$f(x, y) = (1 - \sin t)(-\beta y, \beta x)^T, \ t \in [0, \frac{\pi}{2}].$$  \hspace{1cm} (3.9)

Regarding the selection of the dynamic component \((1 - \sin t)\), there are two considerations. First, at \(t = \frac{\pi}{2}\), we would like to have \(f(x, y) = (0, 0)^T\) and \(f'(x, y) = (0, 0)^T\) in order to fade the curl field smoothly. If the force field fades smoothly, there is no sudden impact on \(P\) when the force field disappears. Second, a trigonometric wave is easy to generate by an analogue circuit. This would make it possible to implement such a dynamic curl field with current technology, such as MEMS and arrays of air jets.

### 3.2.2 Lack of Translation under Curl Fields

After the first stage, the two parts have been centered at the centers of their own half planes (see Figure 1.2 (b)). We position them approximately so that a curl field can be used afterwards. Suppose the COM of \(P\) is located at the origin of the curl field;
i.e., \((x_c, y_c)^T = (0, 0)^T\). So

\[ F_x = - \int_S (1 - \sin t) \beta y ds = -(1 - \sin t) \beta y_c S = 0. \]

For the same reason, \(F_y = 0\). So the external force applied on \(P\) is zero. The only force affecting translation is the damping force. From Equation 2.10, the viscous damping model, \(\mathbf{F}_d = -k \mathbf{v}_c S\). If the initial velocity of the COM is zero, \(\mathbf{F}_d = 0\) at all times. Thus \(P\) will not be translated by the curl field.

### 3.2.3 Rotation Control Using Curl Fields

From Equation 2.5, the rotation angle of \(P\) is decided by the torque applied on \(P\). Because the dynamic curl force field only exists during the time from 0 to \(\frac{\pi}{2}\), the total rotation angle of \(P\) is calculated in two steps. The first step involves the torques from both the force field and damping. The second step involves only the torque from damping. The calculation proceeds as follows:

- **Step 1:** Both the external torque (from the curl field) and the damping torque apply to \(P\), for \(t \in [0, \frac{\pi}{2}]\). The external torque is

\[
T(t) = \int_S \mathbf{r} \times \mathbf{f} ds
\]

\[
= (1 - \sin t) \beta \int_S (X \cdot \mathbf{\hat{i}} + Y \cdot \mathbf{\hat{j}}) \times (-y \cdot \mathbf{\hat{i}} + x \cdot \mathbf{\hat{j}}) dX dY
\]

\[
= (1 - \sin t) \beta \int_S (X \cdot \mathbf{\hat{i}} + Y \cdot \mathbf{\hat{j}}) \times [-y_c + Y \cdot \mathbf{\hat{i}} + (x_c + X) \cdot \mathbf{\hat{j}}] dX dY
\]

\[
= (1 - \sin t) \beta \int_S (X^2 + Y^2) dxdy + \int_S (y_c Y + x_c X) dXdY \cdot \mathbf{\hat{k}}
\]

\[
= (1 - \sin t) \beta \frac{I_z}{\rho} \mathbf{\hat{k}}. \tag{3.10}
\]

Again \(\int_S (y_c Y + x_c X) dXdY = 0\) because \((X_c, Y_c)^T = (0, 0)^T\). Notice that \((x_c, y_c)^T\) is not necessary equal to \((0, 0)^T\) in the above calculation. Once \(P\) is completely under the influence of the curl field, the external torque has the same magnitude over the whole field at any moment. Plugging \(T(t)\) and \(T_d(t)\) in Equation 2.5, we get

\[
T(t) + T_d(t) = I_z \alpha_1(t),
\]
\[(1 - \sin t)\frac{\beta}{\rho} - \frac{k}{\rho} \omega_1(t) = \alpha_1(t), \]
\[\dot{\omega}_1(t) + \frac{k}{\rho} \omega_1(t) = (1 - \sin t)\frac{\beta}{\rho}. \tag{3.11}\]

We solve Equation 3.11 with the initial condition \(\omega(0) = 0\) and get
\[\omega_1(t) = C_1 \cos t + C_2 \sin t + C_3 - (C_1 + C_3)e^{-\frac{k}{\rho}t}, \tag{3.12}\]
where
\[C_1 = \frac{\beta \rho}{\rho^2 + k^2}, \quad C_2 = -\frac{\beta k}{\rho^2 + k^2}, \quad C_3 = \frac{\beta}{k}.\]

The rotation angle during step 1 is
\[\theta_1 = \int_0^{\frac{\pi}{2}} \omega_1(t)dt. \tag{3.13}\]

- **Step 2**: Only the damping torque is in effect, for \(t \in (\frac{\pi}{2}, +\infty)\). The differential equation for \(\omega_2(t)\) is
\[\dot{\omega}_2(t - \frac{\pi}{2}) = -\frac{k}{\rho} \omega_2(t - \frac{\pi}{2}), \quad t \in (\frac{\pi}{2}, +\infty). \tag{3.14}\]

The above differential equation can be rewritten as
\[\dot{\omega}_2(t) = -\frac{k}{\rho} \omega_2(t), \quad t \in (0, +\infty). \tag{3.15}\]

We solve it with the initial condition \(\omega_2(0) = \omega_1(\frac{\pi}{2})\) and get \(\omega_2(t) = \omega_1(\frac{\pi}{2})e^{-\frac{k}{\rho}t}\). The rotation angle during step 2 is
\[\theta_2 = \int_0^{\infty} \omega_2(t)dt. \tag{3.16}\]

So, the total rotation angle is
\[\theta = \theta_1 + \theta_2 \]
\[= C_1(1 - \frac{\rho}{k}) + C_2(1 + \frac{\rho}{k}) + C_3 \frac{\pi}{2} \]
\[= \frac{\beta \pi - 2}{2k}. \tag{3.17}\]
Let us define the damping reciprocal

$$\eta = \frac{\pi - 2}{2k}. \quad (3.18)$$

Equation 3.17 can be rewritten as:

$$\theta = \beta \eta. \quad (3.19)$$

Since $k$ is a constant under a force field, $\eta$ is a constant. Thus we can easily control the rotation angle by properly selecting the value of $\beta$, which is the magnitude of the curl field. For example, if $k = 0.3$ and a part needs to be rotated by $\theta = \pi$, $\beta = \frac{\theta}{\eta} = \frac{0.6\pi}{\pi - 2} \approx 1.65$.

### 3.2.4 Flexibility of Curl Fields

It is simple to control the rotation angle with only one parameter. However, this rotating requires preprocessing, namely the centering of the parts. There inevitably exists some error during centering, which means the parts can't be guaranteed to stop at the exact centers of their own half planes. Instead, the parts can only be guaranteed to stop inside a small neighborhood of the centers. Fortunately, the flexibility of curl fields can handle this preprocessing error. In Equation 3.10, the calculation of $T$ did not require that the COM of $P$ is located at the origin of the curl field. In fact, the external torque applied on $P$ has the same magnitude when $P$ is completely contained in the curl field. So a dynamic curl field can rotate a part to its desired orientation as if the COM of the part is at the origin of the curl field. Furthermore, if $P$ stays inside a small neighborhood of the field origin, the external force $F(x,y) = -y \cdot \vec{i} + x \cdot \vec{j}$ is very small. Also, the curl field only appears during $t \in [0, \frac{\pi}{2})$. Hence the dynamic curl field is unlikely to move $P$ far from the field origin. This has been verified by our simulations and the error will be further discussed further in Chapter 4.
3.3 Assembling Parts Using Dynamic Push Fields

The last step of the assembly process is to translate the inserting part $P_1$ from its current position to its assembly position. $P_1$ stays at the center of its own half plane after rotation. It is desired only to translate $P_1$ but not to move the inserted part $P_2$. Before deciding which translation field to use, observe that a straight-line trajectory of $P_1$ is easier to control than a curved-line trajectory. Another consideration is the geometric shape of the two parts. If the assembly position of $P_1$ is on the global x-axis, we call the pair of assembled parts a “symmetric” pair (see Figure 3.4 (a)) and a single translation can complete the assembly; otherwise, we call them an “asymmetric” pair (see Figure 3.4 (b)) and two translations are needed.

Figure 3.4: (a) is a symmetric pair of parts while (b) is an asymmetric pair of parts.

From the previous section, we know that a radial field can translate parts. If a global radial field has the same origin as the curl field for $P_2$, the radial field can translate $P_1$ without moving $P_2$. However, this field can only assemble symmetric pairs of parts, not asymmetric pairs. Under the radial field, a part is always translated toward the origin of the field. So the trajectory of $P_1$ is on the global x-axis, since its initial position is on the global x-axis; see Figure 3.4. For an asymmetric pair of parts, the assembly position of $P_1$ is not on the global x-axis (see Figure 3.4 (b)). This results in assembly failure, since the assembly position of $P_1$ is not on its trajectory under the radial field. Even if the parts are “symmetric”, using a radial field, when
\( P_1 \) reaches its assembly position, its kinematic energy is maximal (converted from its potential energy); so we need a model for the impact of the two parts and a rule to judge whether they can be assembled or not. This is an interesting problem. In this case, we recommend translating the inserted part \( P_2 \) instead of \( P_1 \) because the motion of \( P_2 \) can help aligning \( P_1 \) to itself, like a push-align operation [2, 15].

This thesis proposes a dynamic push force field for translation control, which is only in effect in the half plane of \( P_1 \). A dynamical push force field is of the form:

\[
f = \gamma (1 - \sin t) \cdot \mathbf{n},
\]  

(3.20)

where \( \gamma \) is the magnitude of the field, \( \mathbf{n} \) is unit vector in the direction of translation, with the form \( \mathbf{n} = \cos \phi \cdot \mathbf{i} + \sin \phi \cdot \mathbf{j} \), and \( \phi \) is the angle between \( \mathbf{n} \) and the x-axis. Figure 3.5 shows a push force field with \( \phi = -\frac{\pi}{4} \) and \( \gamma = 1 \).

![Figure 3.5: A push field with \( \phi = -\frac{\pi}{4} \) and \( \gamma = 1 \).](image)

### 3.3.1 Translation Control Using Push Fields

First, the above push field can’t rotate a part, since the external torque is

\[
T(t) = \int_s \mathbf{r} \times f ds
\]
\[ \begin{align*}
&= \gamma (1 - \sin t) \int_S (X \cdot \mathbf{i} + Y \cdot \mathbf{j}) \times (\cos \phi \cdot \mathbf{i} + \sin \phi \cdot \mathbf{j}) dXdY \\
&= \gamma (1 - \sin t) [\sin \phi X_c S - \cos \phi Y_c S] \cdot \mathbf{k} = 0.
\end{align*} \]

Again, the reason why \( T(t) = 0 \) is that \((X_c, Y_c)^T = (0, 0)^T\). Furthermore, because the angular velocity is zero after rotation, the damping torque is always zero, the same reason that \( F_d = 0 \) under the curl field. So \( P \) will only be translated by the push field while keeping its initial orientation.

Translation control is almost the same as rotation control. It proceeds in the following two steps.

- **Step 1:** Both the external force and the damping force apply to \( P \), for \( t \in [0, \frac{\pi}{2}] \).

The external force is

\[ F(t) = \int_S \mathbf{f} ds = (1 - \sin t) \gamma S \mathbf{n}. \]

Plugging in \( F(t) \) and \( F_d(t) \) from Equation 2.4, we get,

\[ (1 - \sin t) \gamma - kv_{c1}(t) = \rho \dot{v}_{c1}(t). \]

We solve the above equation with the initial condition \( v_{c1}(0) = 0 \) and get

\[ v_{c1}(t) = C_1 \cos t + C_2 \sin t + C_3 - (C_1 + C_3) e^{-\frac{k}{\rho} t}, \tag{3.21} \]

where

\[ C_1 = \frac{\gamma \rho}{\rho^2 + k^2}, \quad C_2 = -\frac{\gamma k}{\rho^2 + k^2}, \quad C_3 = \frac{\gamma}{k}. \]

So the translated distance during step 1 is

\[ d_1 = \int_0^{\frac{\pi}{2}} v_{c1}(t) dt. \tag{3.22} \]

- **Step 2:** Only the damping force is in effect, for \( t \in (\frac{\pi}{2}, +\infty) \). The differential equation for \( v_{c2}(t) \) is:

\[ \dot{v}_{c2}(t - \frac{\pi}{2}) = -\frac{k}{\rho} v_{c2}(t - \frac{\pi}{2}), \quad t \in (\frac{\pi}{2}, +\infty). \tag{3.23} \]

The above equation can be rewritten as

\[ \dot{v}_{c2}(t) = -\frac{k}{\rho} v_{c2}(t), \quad t \in (0, +\infty). \tag{3.24} \]
We solve the above differential equation with the initial condition \( v_{c2}(0) = v_{c1}\left(\frac{\pi}{2}\right) \) and get

\[
v_{c2}(t) = v_{c1}\left(\frac{\pi}{2}\right) e^{-\frac{k}{\rho}t}.
\]  \hspace{1cm} (3.25)

The distance during step 2 is

\[
d_2 = \int_0^\infty v_{c2}(t)\,dt.
\]  \hspace{1cm} (3.26)

So, the total translated distance is

\[
d = d_1 + d_2
\]
\[
= C_1\left(1 - \frac{\rho}{k}\right) + C_2\left(1 + \frac{\rho}{k}\right) + C_3 \frac{\pi}{2}
\]
\[
= \gamma \frac{\pi - 2}{2k} = \gamma \eta,
\]  \hspace{1cm} (3.27)

where \( \eta \) is the same damping reciprocal as the one in the curl field. There are two important points for translation control. First, for symmetric pairs of parts, one horizontal push is enough to complete the assembly. However, for asymmetric pairs of parts, two push fields are needed: one along y direction, then one along x direction. Second, the push field is in effect only on \( P_1 \)'s half plane, which means that \( P_2 \) will not be moved. For vertical translation, this is not an issue since \( P_1 \) is not going outside its half plane. For the horizontal push, we have to verify its correctness. Since the push field applies on \( P_1 \) only in \( d_1 \), the field does not need to exist in \( d_2 \) during the whole process. The distance between the centers of the two half planes is twice as long as the distance between the center of either half plane and the global y-axis. So if \( \frac{d_2}{d} > \frac{1}{2} \), the field in \( P_1 \)'s half plane will guarantee the correct results. Generally this is true since

\[
\frac{d_2}{d} = \frac{2\rho^3 - 2e^{\frac{\pi}{2}}(k^2 + k\rho + \rho^2)}{k\pi\rho^2 + k^3\pi - 2k^3 - 2k^2\rho^2}.
\]  \hspace{1cm} (3.28)

From mathematical analysis, we know that \( \frac{d_2}{d} \) will increase with \( \rho \) increasing and \( k \) decreasing. When \( \rho = 1, k = 0.5 \), \( \frac{d_2}{d} = 0.567 \). In most situations \( k \leq 0.5 \). Thus local dynamic push fields in \( P_1 \)'s half plane can translate \( P_1 \) successfully.
3.4 Fine Control Relaxation

Our assembly strategy is divided into three stages: centering, rotating and pushing parts. From the discussion in previous sections, a radial field can center a part without changing its orientation. The part will oscillate around the origin of the radial field for a few times and its oscillation magnitude will be decreased gradually. The stopping speed of the part is decided by the attenuation component $e^{-\frac{t}{\beta}}$ in Equation 3.8. In theory, the part will never stop but instead move closer and closer to the origin of the radial field. However, it is reasonable to set a threshold value to stop the centering process, since an exponential function attenuates fast. Generally, $10^{-5}$ is a very good accuracy both for real experiments and simulations. Suppose $\rho = 1$ and $k = 0.3$; Table 3.1 lists a few values of the attenuation component. In this scenario, once time is greater than 80 seconds, the centering process can be considered completed. There is no longer a need for the radial field.

<table>
<thead>
<tr>
<th>time: $t$ (seconds)</th>
<th>$e^{-\frac{t}{\beta}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>\leq 0.05</td>
</tr>
<tr>
<td>40</td>
<td>\leq 0.003</td>
</tr>
<tr>
<td>60</td>
<td>\leq 0.0002</td>
</tr>
<tr>
<td>80</td>
<td>\leq 6.5 \times 10^{-6}</td>
</tr>
<tr>
<td>100</td>
<td>\leq 3.5 \times 10^{-7}</td>
</tr>
</tbody>
</table>

Table 3.1: Attenuation value table.

The dynamic curl force field can rotate the part by a specified angle with an appropriate magnitude $\beta$ of the curl field. This rotation control proceeds in two steps as described in section 3.2.3. In the calculation of the rotation angle during step 2 (only the damping torque applies on the part), the upper limit of the integration in Equation 3.16 is infinity. This is not practical in the real world. From the previous
calculation, \( \omega_2(t) = \omega_1(\pi \over 2)e^{-\frac{k}{\rho}t} \). The rotation angle until time \( T \) is

\[
\theta_2 = \int_0^T \omega_1(\pi \over 2)e^{-\frac{k}{\rho}t}dt \\
= -\frac{\rho}{k}\omega_1(\pi \over 2)e^{-\frac{k}{\rho}T} \bigg|_0^T \\
= \frac{\rho}{k}\omega_1(\pi \over 2)(1 - e^{-\frac{k}{\rho}T}).
\]

When \( T = \infty \), \( e^{-\frac{k}{\rho}\infty} = 0 \) and \( \theta_2 = \frac{\rho}{k}\omega_1(\pi \over 2) \). In the same way that we considered the attenuation component in the radial field, we can use a threshold value to decide when to stop this rotation. As far as the attenuation speed is concerned, \( e^{-\frac{k}{\rho}T} \) is twice as fast as \( e^{-\frac{k}{\rho}t} \). For example, with the same \( \rho \) and \( k \) as the previous example for the radial field, the time for \( e^{-\frac{k}{\rho}T} \leq 10^{-5} \) is only 40 seconds. Furthermore, since \( \theta_2 \) is always smaller than \( \theta \), assuming the calculation of \( \theta_1 \) does not have any error, the relative error of \( \theta_2 \) is also smaller than the relative error of \( \theta \). So the threshold value for \( \theta_2 \) is also valid for \( \theta \).

A dynamic push force field can translate a part by a specified distance with an appropriate magnitude \( \gamma \) of the curl field. This translation control proceeds in two steps similar to the rotation control. In the calculation of the translated distance during step 2 (only the damping force applies on the part), the upper limit of the integration in Equation 3.26 is also infinity. After doing a similar analysis, we can use the same threshold value as the one for rotation control to stop the pushing process.

By relaxing the fine control during the three stages in the assembly process, some control errors are inevitably introduced. Besides, in a real implementation, all the force fields, parts and contact plane can’t be perfect. Those defects also introduce other errors in the whole assembly process. Chapter 4 will discuss these uncertainty issues in detail.
Chapter 4

Uncertainties in Part Assembly

In order to address uncertainty issues involved in assembling two parts, we first give some related background from robotics.

4.1 Related Background From Robotics

Configuration Space Most of the following definitions are copied from [23]. Suppose there is a robot \( \mathcal{A} \) and a few obstacles \( \mathcal{B}_i \) \( (i = 1, 2, \ldots, n) \) in the workspace \( \mathcal{W} (\mathbb{R}^2) \) in this case). The underlying idea of configuration space is to represent \( \mathcal{A} \) as a point in \( \mathcal{A} \)'s configuration space and map the obstacles in this space. This mapping transforms the problem of planning the motion of a dimensioned object into the problem of planning the motion of a point. A configuration \( \mathbf{q} \) of \( \mathcal{A} \) is a specification of the position and orientation of \( \mathcal{A} \) with respect to a fixed reference frame. The subset of \( \mathcal{W} \) occupied by \( \mathcal{A} \) at configuration \( \mathbf{q} \) is denoted by \( \mathcal{A}(\mathbf{q}) \). The configuration space \( \mathcal{C} \) of \( \mathcal{A} \) is the space of all the configurations of \( \mathcal{A} \). Every obstacle \( \mathcal{B}_i, i = 1, 2, \ldots, n, \) in the workspace \( \mathcal{W} \) maps in \( \mathcal{C} \) to a region:

\[
\mathcal{CB}_i = \{ \mathbf{q} \in \mathcal{C} | \mathcal{A}(\mathbf{q}) \cap \mathcal{B}_i \neq \emptyset \},
\]

which is called a **C-obstacle**. The union of all the **C-obstacles** is called the **C-obstacle region**.

Uncertainty This thesis uses the following uncertainty model. Let the robot be at an actual configuration \( \mathbf{q} \) and let the nominal configuration be \( \mathbf{q}^* \). The nominal configuration means the configuration at which the robot should be. The error \( \mathbf{q} - \mathbf{q}^* \) comes from sensing, the imperfect control of the force fields and the imperfect control
of the motion of the assembled parts. Our model of uncertainty bounds the modulus of this error; i.e., \( \| q - q^* \| \leq e_q \) (\( e_q \) is the error bound), and accepts all the configurations satisfying \( \| q - q^* \| \leq e_q \) as equally probable ones; see Figure 4.1 (a) and (b). Besides configuration uncertainty, there is some uncertainty in control. Let \( \mathbf{v}_0 \) be the nominal velocity. At any instant \( t \), the actual velocity \( \mathbf{v}(t) \) lies in an open cone of angle \( 2\psi \) whose axis points along \( \mathbf{v}_0 \). This cone is called the control uncertainty cone; see Figure 4.1 (c).

**Goal Kernel and Preimage** The goal kernel and preimage definitions in [11, 23] involve too many details which are not related to this thesis. We simplify their definitions as follows. Suppose \( \mathcal{T} \) is the goal region of \( \mathcal{A} \), which \( \mathcal{A} \) tries to reach. The **goal kernel** of \( \mathcal{T} \) is a subset \( \mathcal{X}(\mathcal{T}) \) of \( \mathcal{T} \) whose achievement by a motion command along \( \mathbf{v}_0 \) is recognizable by permitting the uncertainties present, independent of the starting region. A **preimage** of \( \mathcal{T} \) for a motion command is defined as any subset \( \mathcal{P} \) of valid space such that: if \( \mathcal{A} \)'s configuration is in \( \mathcal{P} \) when the execution of the motion command starts, it is guaranteed that \( \mathcal{A} \) will reach \( \mathcal{T} \) and will be in \( \mathcal{T} \) when \( \mathcal{A} \) stops. The valid space is the subset of configuration space in which \( \mathcal{A} \) can move.

### 4.2 Involved Uncertainties

In previous chapters, we assumed that the force fields and the motions of the parts can be perfectly controlled. However, for any real implementation, uncertainty should be taken into account. A lot of research has addressed the uncertainty issue, e.g., [11, 23]. For part assembly, even though we only use sensing to find the initial orientations of the assembled parts, we still need to consider the following uncertainties mainly because of force field control:

- **Position uncertainty**, see Figure 4.1 (a);
- **Orientation uncertainty**, see Figure 4.1 (b);
- **Velocity uncertainty**, see Figure 4.1 (c).
Figure 4.1: Three kinds of uncertainties: (a) shows position uncertainty. (b) shows the orientation uncertainty. (c) shows the velocity uncertainty.

We consider all three uncertainties of the inserting part $P_1$ during the horizontal push in our assembly process because these uncertainties directly decide whether the assembly will succeed or not. However, these uncertainties exist in all three stages of part assembly and the uncertainties in the previous stages affect those in the later stages. So, the three uncertainties in the final horizontal push accumulate from centering, rotation, possible vertical translation and the horizontal push itself. Furthermore, we only need to take into account the uncertainties of $P_1$ because the uncertainties of the inserted part $P_2$ can be transformed to the relative uncertainties of $P_1$ with respect to $P_2$. This transformation simplifies the uncertainty analysis. The problem becomes similar to a classical uncertainty problem, namely the peg-in-hole problem [11, 23].

To construct a product from the two individual parts, around $P_2$ there exists a goal region of $P_1$ and this goal region has to be larger than $P_1$. If the goal region is much larger than $P_1$, we can call this kind of assembly gross assembly; otherwise, we call it precise assembly. To describe the size difference, let us define the size difference rate:

$$\delta = \frac{S_g - S_p}{S_p}, \quad (4.1)$$

where $S_g$ is the size of the goal region and $S_p$ is the size of $P_1$. A threshold value, $\delta_0 \in (0, 1)$, is helpful to distinguish precise assembly and gross assembly. This threshold value can be different under different scenarios.
4.3 Assembly Handling Uncertainties

Consider $P_1$ as a robot $A$ and $P_2$ as an obstacle $B$. This thesis presents an algorithm to resolve the horizontal push of $A$ by considering the uncertainties, as shown in Figure 4.2. All of the C-obstacle, goal region, goal kernel and preimage in this algorithm are computed in $A$'s configuration space.

- Calculate the the C-obstacle [23];
- Calculate the goal kernel from the goal region of the robot by considering the position uncertainty of the robot;
- Calculate the preimage of the goal kernel using Erdmann's backprojection algorithm [11];
- If the position uncertainty disk of the robot before the horizontal push is contained inside the preimage, the assembly is guaranteed to succeed.

Figure 4.2: An algorithm that deals with uncertainties in part assembly.

4.3.1 C-obstacle Calculation

For the C-obstacle calculation, the COM of the robot is selected as the reference point. The algorithm we use to calculate the C-obstacle is detailed in Figure 4.3. Here $\text{vert}(A)$ is the vertex set of $A$ and $\ominus$ is Minkowski difference, $X \ominus Y = \{x - y| x \in X, y \in Y\}$. This algorithm is taken from [23].

Because of orientation uncertainty, the configuration space should be extended from $R^2$ to $R^3$. However, this can be avoided by enlarging $A$. One solution is to calculate the minimal box containing the robots with all possible orientations. Since the COM
• Triangulate non-convex robot and obstacle;

• Compute C-obstacles for each pair of convex subset of $\mathcal{A}$ and convex subset of $\mathcal{B}$ by the following method:
  
  – Compute $\text{vert}(-\mathcal{A})$;
  
  – Compute $\text{vert}(-\mathcal{A}) \ominus \text{vert}(\mathcal{B})$;
  
  – Compute the convex hull of $(\text{vert}(-\mathcal{A}) \ominus \text{vert}(\mathcal{B}))$.

• Compute the union of all the C-obstacles.

Figure 4.3: C-obstacle computation algorithm. $\text{vert(\mathcal{A})}$ is the vertex set of $\mathcal{A}$ and $\ominus$ is Minkowski difference.

of the robot is the reference point (the origin of $\mathcal{A}$'s coordinate frame), under the orientation uncertainty, every vertex $\mathcal{A}_i$ of the robot may lie at any position on an arc between $\mathcal{A}_i^l$ and $\mathcal{A}_i^g$, where $\mathcal{A}_i^l$ and $\mathcal{A}_i^g$ are the $i$th vertices of $\mathcal{A}$ with the two extreme orientations. Such an arc is contained inside the triangle $\mathcal{A}_i^g \mathcal{A}_i^l \mathcal{A}_i^o$; see Figure 4.4. $\mathcal{A}_i^o$ is the intersection point of the two tangent lines at $\mathcal{A}_i^l$ and $\mathcal{A}_i^g$. Suppose the orientation uncertainty parameter $\phi = 0.1$. Figure 4.5 shows a rectangular robot and the box computed by the above method.

4.3.2 Goal Kernel and Preimage Calculation

Since the assembly process does not use any sensing in the last stage, the goal kernel calculation becomes simple. $\mathcal{A}$ may be at any position inside the uncertainty disk whose center is the nominal position. So the goal kernel can be obtained by shrinking the goal region by the radius of the uncertainty disk; see Figure 4.6. Erdmann presented an algorithm to obtain the preimage of a goal region $\mathcal{T}$ by computing the backprojection of $\mathcal{T}$ [11]. This algorithm computes the backprojection by tracing
Figure 4.4: A method to calculate the box containing the robots with all the possible orientations when the orientation uncertainty is $\phi$.

Figure 4.5: (a) is a rectangular robot. (b) shows a box to contain the robot with all possible orientations when the uncertainty $\phi = 0.1$.

... every vertex in $C_{\text{contact}}$, where $C_{\text{contact}}$ is the boundary of $C$-obstacle. Erdmann's algorithm works nicely for $T$s with different shapes. However, in our algorithm handling uncertainties (Figure 4.2), we only need to calculate the backprojection of the goal kernel. Since the goal kernel does not contain any vertex in $C_{\text{contact}}$, the computation of the backprojection can be simplified as shown in Figure 4.7. Figure 4.8 shows how the preimage, goal kernel and $C$-obstacle look in a gross assembly.

All of the above algorithms have been implemented in C++ by using CGAL (Computational Geometry Algorithms Library).
Figure 4.6: The goal region $\mathcal{T}$ is a rectangle in a configuration space. The goal kernel $\mathcal{X}$ is obtained by shrinking $\mathcal{T}$ by the radius of the uncertainty disk.

- Mark every vertex of the goal kernel as white if it faces toward the initial position of $\mathcal{A}$;
- Mark the highest and lowest white vertices as black;
- At every black vertex erect two rays parallel to the sides of the inverted velocity uncertainty cone. Compute the intersection of these rays.
- Trace out the backprojection region. By starting from the highest black vertex and following its erected ray in the direction that the leaves the goal kernel on the left hand side. Continue this process until arriving back to the start vertex.

Figure 4.7: Backprojection calculation algorithm.
Figure 4.8: (a) is the same box as shown in Figure 4.5 (b). (b) is the inserted part (obstacle in this case). (c) shows the C-obstacle (the light line), the goal region (the area below the horizontal black solid line), the kernel of the goal region (the area below the dotted line) and the preimage of the goal kernel (the thick line).
Chapter 5

Programmable Force Field Simulators and Experimental Results

5.1 A Simulator for Static Force Fields

5.1.1 Introduction

Our software simulates the motion of a part when placed upon some kind of force generating surface. The specific dynamics of a particular implementation of such a plate (MEMS, arrays of air jets, etc.) vary somewhat at the physical level, but the abstraction of a field of forces acting on a planar surface in two dimensions is a common factor in all of these devices. Because of this we have chosen to implement the simulation of these devices on the level of a programmable force field.

This simulator can support two different physical models. One is the quasi-static model (first-order dynamics), in which velocity is proportional to force. This model is widely used in the literature when discussing the properties of programmable fields [7]. The other supported model is the dynamic (Newtonian) mechanical model in which acceleration, velocity and friction are taken into account. The second model is closer to the actual physics.

The program is suitable for simulating the behavior (equilibrium configuration) of a single polygonal part in some force field, or of multiple parts which do not interact with each other.
Figure 5.1: Two different parts reaching equilibrium in the elliptic force field $f(x, y) = (-4x, -2y)$ using the quasi-static model.

5.1.2 User Interface

The software currently supports a command line interface and works with Geomview [12] to produce visual output. The program itself writes gcl (Geomview command language) commands to standard output where a small Perl script intercepts them and sends them to Geomview as well as to a file for storage. The executable is named 'sim_text' and is invoked through the shell script 'run'. The simulator is invoked by the following command:

$$\text{run } a \ b \ [-x_i \ v_{x0}] \ [-y_i \ v_{y0}] \ [-r_i \ \omega_0] \ \text{part_file},$$

where $a$ and $b$ are two parameters which may be used in whichever force field is implemented. The following three arguments are the initial velocities of the part: $v_{x0}$ and $v_{y0}$ are the x and y components of the initial translational velocity, and $\omega_0$ is the initial angular velocity. If any initial velocity, say $v_{x0}$, is not set in the command line, the program takes zero as this initial velocity; i.e., $v_{x0} = 0$. The last argument is the name of the part's file that should be loaded. Since the simulator can support multiple parts, if using more than one part, the user needs to list the parts' file names sequentially.
Supported Force Fields  Currently, the simulator supports the Curl, Elliptical, Push, Radial plus Gravity, Skew, and Squeeze force fields. Among those force fields, the curl and push fields are not potential fields.

5.1.3 Force and Torque Calculation

Of primary importance in the simulation of a part in a force field is the computation of the translational forces and rotational torques on the part as the part’s motion is determined by these factors.

5.1.3.1 Mathematics

Under a force field \( f(x, y) \), the external force applied on the part \( P \) is \( \mathbf{F} = \int_S f(x, y) \cdot ds \). The COM of \( P \) is defined as \( \frac{\int_S \rho \cdot ds}{M} \), where \( M \) is total mass and \( \rho \) is the mass density. Without loss of the generality, let \( \rho = 1.0 \). The moment of inertia at the COM about the \( z \)-axis is defined as \( \int_S r^2 \rho \cdot ds = \int_S r^2 \cdot dXdy = \int_P (x^2 + y^2) \cdot dXdy \). The external torque is \( \int_S (r \times f) \cdot dXdy \).

5.1.3.2 Discretization

We do not simulate the continuous force field and the continuous motion of the part, instead we discretize them. The force generating plane is broken into a square grid. At each pixel of the grid, a force value approximates the actual force over the pixel. Any size may be specified for the grid; a \( 1500 \times 1500 \) grid seems to be a good choice for a \( 1 \times 1 \) square, the force generating plane.

Instead of using integration to calculate \( \mathbf{F} \), the program multiplies the force of each pixel (whose center is inside the part) by the size of each pixel and then adds the results. Since our field is on a unit square, the size of a pixel is \( 1/grid.size^2 \), where \( grid.size \) is the dimension of the array, 1500 in this simulator. The following
equation shows this calculation:

\[ \mathbf{F} = \sum_{\text{pixels inside } P} \mathbf{f} \cdot S_{\text{pixel}}, \]  

where \( S_{\text{pixel}} \) is the size of a pixel, which is \( 1/\text{grid.size}^2 \). Other integrations are performed similarly, adding the values for each square whose center is inside the part and multiplying by \( 1/\text{grid.size}^2 \).

These numeric integrations are performed repeatedly at discrete timesteps and at each timestep the part is moved as though the computed force were constant across the timestep. The timestep may be specified at an arbitrary value; 0.002 seconds works nicely.

5.1.3.3 Damping Force and Torque

In dynamic simulations, if we do not take into account the damping force and torque, the part will swing about the origin of the plate forever. In fact without damping, no force field can pose and orient the part under the dynamic model. Our simulator uses the viscous damping model. From Section 2.2.2, the damping force is \( \mathbf{F}_d = -kS\mathbf{v}_e \) and the damping torque is \( \mathbf{T}_d = -k \frac{L}{\rho} \mathbf{\omega} \), where \( k \) is the damping coefficient, \( \mathbf{v}_e \) is the instantaneous translational velocity of the COM, and \( \mathbf{\omega} \) is the instantaneous rotational velocity of P. Since we have the analytical expressions for \( \mathbf{F}_d \) and \( \mathbf{T}_d \), discretization is not needed for damping. The damping coefficient \( k \) is a constant and is generally in \((0, 1)\).

5.1.4 Coordinate Frames

5.1.4.1 Global Coordinate Frame

The simulation area is taken to be the unit square with corners \((.5, .5, 0)\), \((- .5, .5, 0)\), \((- .5, -.5, 0)\), \((.5, -.5, 0)\). If \((x, y)\) are the coordinates of a point in the force field \((x \text{ and } y \in [-0.5, 0.5]), (\text{grid.size} \times x + \lfloor \text{grid.size}/2 \rfloor, \text{grid.size} \times y + \lfloor \text{grid.size}/2 \rfloor)\) are the coordinates of the same point in the grid.
5.1.4.2 Local Coordinate Frame

Each part has its own coordinate frame, in which its COM is at \((0, 0)\). This frame has the same scale as the simulations frame; i.e., iff two points are distance \(d\) apart in the local frame they are distance \(d\) apart in the global frame. Thus the current configuration of the part is described as a 3-tuple \((x, y, \theta)\) such that the point \((X, Y)\) on the part is the point \((X \times \cos(\theta) - Y \times \sin(\theta) + x, X \times \sin(\theta) + Y \times \cos(\theta) + y)\) in the simulations frame.

5.1.5 Features

5.1.5.1 Quasi-static Simulation

The simulation supports a quasi-static physics model in which the force and torque are computed on a part for a certain timestep, scaled appropriately by mass and moment of inertia, and then taken to be the velocity (instead of the acceleration); i.e.,

\[
\mathbf{v} = \frac{\mathbf{F}}{M},
\]

\[
\omega = \frac{\mathbf{T}}{I_z}.
\]

The part's position is then updated accordingly. Since the velocity of the part is proportional to the external force, the part, once at equilibrium, will stay there. No damping is applied in the quasi-static model.

5.1.5.2 Dynamic Simulation

In the dynamic model, during every timestep the external force and torque are calculated as in the quasi-static, but are used to compute acceleration. Also given an initial translational velocity \(\mathbf{v}_0\) and rotational velocity \(\omega_0\), the program calculates the damping force \(\mathbf{F}_d = -k\mathbf{v}_0\) and damping torque \(\mathbf{T}_d = -k\omega_0\). The external force and damping force are used to calculate the translational acceleration \(\mathbf{a}\). The external torque and damping torque are used to calculate the rotational acceleration \(\alpha\).
Then we can find the final translational velocity \( \mathbf{v} = \mathbf{v}_0 + \mathbf{a}t \) and rotational velocity \( \mathbf{\omega} = \mathbf{\omega}_0 + \mathbf{\alpha}t \). These velocities will be used as the initial velocities in next timestep. The translation distance and rotation angle of the part during a timestep can be calculated by multiplying the average translational velocity \( \frac{\mathbf{v}_0 + \mathbf{v}}{2} \) and rotational velocity \( \frac{\mathbf{\omega}_0 + \mathbf{\omega}}{2} \), with timestep, \( t \), respectively.

5.1.5.3 Limitations

- Multiple parts are supported. However, currently no collision detection or interaction is supported between the parts; they will simply “pass” through each other.

- There is no boundary checking to ensure that the parts remain on the grid; certain force field implementations may crash the program if the part leaves the square plate, others may function perfectly well.

5.2 A Simulator for Part Assembly

The simulator for part assembly is based on the simulator for static force fields. The difference is the following.

- The plane is enlarged to \((\pm 1, \pm 0.5)\) and each force field is only effective in one half plane. The two force fields in the two half planes do not affect each other. Each half plane is still discretized into a \(1500 \times 1500\) grid.

- As it is known how to calculate the external force and torque of the radial, curl and push fields, numeric integration to calculate \( \mathbf{F} \) and \( \mathbf{T} \) is not needed any longer. However, the motion of the part is still discretized by the same timestep, 0.002 seconds in this simulator.

- Dynamic force fields are implemented. Adding a dynamic factor, e.g. \((1 - \sin t)\), can do it simply. During the second step of rotation and pushing, the force field
disappears and the simulator uses a zero force field for the motion calculation.

- The driving function of the assembly simulator is changed from one stage (for static force fields) to three stages (for assembly). Each previous stage affects the subsequent stages. For example, the exact positions of the parts after centering are used as the parts’ initial positions for the rotation by the curl field. The exact positions and orientations of the parts after rotation are used as the parts’ initial configuration for the final push stage.

5.3 Experimental Results

The simulator of static force fields can nicely mimic the motion of a part in a static force field and the simulation result is close to the theoretical prediction. For squeeze fields, elliptical fields and skew fields, the part can reach its equilibrium configuration (both position and orientation) quickly.

5.3.1 Simulation Results of Static Force Fields

5.3.1.1 Results of Elliptical Force Fields

This experiment verifies the correctness and accuracy of the simulations of a polygonal part under an elliptical force field. Put a rectangular part into an elliptical force field, \( f(x, y) = (-x, -2y) \). Suppose the rectangular part has a unique mass distribution and is described by its vertices: \( \frac{1}{25} \{(1, -0.5), (1, 0.5), (-1, 0.5), (-1, -0.5)\} \). According to [19], such a rectangular part has two stable configurations: \((0, 0, 0)\) and \((0, 0, \pi)\). This experiment uses 0.6 as the damping coefficient. Table 5.1 compares the simulation results with the expected results. Considering the numeric integration error and the discretization, the simulation results are acceptable.
<table>
<thead>
<tr>
<th>Initial configuration</th>
<th>Experimental equilibrium configuration</th>
<th>Expected configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.1, -0.1, 2)</td>
<td>($4 \times 10^{-4}$, $4 \times 10^{-4}$, 3.1416)</td>
<td>(0, 0, $\pi$)</td>
</tr>
<tr>
<td>(-0.2, 0.1, 1.57)</td>
<td>($4 \times 10^{-4}$, $4 \times 10^{-4}$, 3.1416)</td>
<td>(0, 0, $\pi$)</td>
</tr>
<tr>
<td>(-0.2, -0.2, 1)</td>
<td>($2 \times 10^{-4}$, $2 \times 10^{-4}$, $4 \times 10^{-6}$)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>(0.1, -0.1, 0)</td>
<td>($2 \times 10^{-4}$, $2 \times 10^{-4}$, $3 \times 10^{-6}$)</td>
<td>(0, 0, 0)</td>
</tr>
</tbody>
</table>

Table 5.1: Simulation results for an elliptical force field.

### 5.3.1.2 Results of Squeeze Force Fields

This experiment verifies whether or not the simulation of the motion of a part under a squeeze force field obeys the theoretical prediction. The experimental part is the same as the part used in the experiment for the elliptical field. The damping coefficient is also 0.6. The squeeze force field is of the form:

$$f(x, y) = \begin{cases} 
(-1, 0), & x < 0; \\
(1, 0), & x > 0; \\
(0, 0), & x = 0;
\end{cases}$$

Table 5.2 compares the simulation results and the predicted results. The predicted results are computed from [9].

A skew force field is of the form: $f(x, y) = -\text{sign}(x)(1, \epsilon)$, where $\epsilon \in R$ but $\epsilon \neq 0$.

<table>
<thead>
<tr>
<th>Initial configuration</th>
<th>Experimental equilibrium configuration</th>
<th>Expected configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.1, -0.1, 2)</td>
<td>($4 \times 10^{-4}$, -0.1, 1.57081)</td>
<td>(0, 0, $\frac{\pi}{2}$)</td>
</tr>
<tr>
<td>(-0.2, 0.1, 1.57)</td>
<td>($4 \times 10^{-4}$, 0.1, 1.57069)</td>
<td>(0, 0, $\frac{\pi}{2}$)</td>
</tr>
<tr>
<td>(-0.2, -0.2, 1)</td>
<td>($4 \times 10^{-4}$, -0.2, 1.57075)</td>
<td>(0, 0, $\frac{\pi}{2}$)</td>
</tr>
<tr>
<td>(0.1, -0.1, 0)</td>
<td>($2 \times 10^{-4}$, -0.1, 4.71243)</td>
<td>(0, 0, $\frac{3\pi}{2}$ or $\frac{\pi}{6}$)</td>
</tr>
</tbody>
</table>

Table 5.2: Simulation results for an squeeze force field.
When $\epsilon = 0$, the force field becomes a squeeze field. We also ran a few simulations for skew fields; the results are similar to the results for squeeze fields.

5.3.1.3 Results of Radial Plus Gravity Force Fields

The experimental results for radial plus gravity force fields are not so promising as the elliptical fields and squeeze fields. Our simulation results show that a radial plus gravity field can uniquely pose a part to its equilibrium position with accuracy similar to that of the elliptical field. However, the final orientations of the part are undecided under a radial plus gravity field, because the pivot point of a part in a radial field can be very close to its COM. The simulator uses numeric methods to calculate the COM, the external force, and the external torque of P, which inevitably have some calculation error. This calculation error may result in a confusion between the pivot point and the COM of P. According to [20], whether a part has a pivot point distinct from its COM is critical to deciding the orientation ability of radial plus gravity fields. In the future, we would like to explore analytical solutions to calculate the COM, the external force, and the external torque of the part.

5.3.2 Simulation Results of Part Assembly

The simulator of part assembly works nicely for polygonal parts. As an example, Figure 1.2 shows snapshots for the three stages in two part assembly. Currently the simulator supports only polygonal parts. We have not yet obtained experimental results for nonpolygonal parts. However, since we do not make any assumption about the shapes of the assembled parts, it is reasonable to expect that nonpolygonal parts can also be assembled by this assembly strategy.
Chapter 6

Conclusion and Future Work

Our strategy provides a feasible solution for part assembly. During the entire analysis of the force fields, we have not made any assumption about the shape of the parts; i.e., any pair of parts can be assembled with this strategy. The simulator illustrates quite a few assembly operations and verifies the correctness of this assembly strategy. All three kinds of force fields used in our approach are simple and hence we expect that practical implementations will be possible. For radial fields, both MEMS and arrays of the air jets could be used. For dynamic curl and push fields, air jet arrays may be more appropriate since one can control the magnitude and frequency of each air jet. Furthermore, the dynamic component, $(1 - \sin t)$, is easy to generate by an analog circuit. Most of all, the manipulation at each stage is simple, as shown in Table 6.1. This is the most attractive feature of this research.

<table>
<thead>
<tr>
<th>task</th>
<th>field</th>
<th>dynamic</th>
<th>plan steps</th>
<th>parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>centering</td>
<td>radial</td>
<td>no</td>
<td>1</td>
<td>none</td>
</tr>
<tr>
<td>rotation</td>
<td>curl</td>
<td>yes</td>
<td>1</td>
<td>(\beta)</td>
</tr>
<tr>
<td>translation</td>
<td>push</td>
<td>yes</td>
<td>2 at most</td>
<td>(\gamma)</td>
</tr>
</tbody>
</table>

Table 6.1: Assembly using force fields.

By relaxing the fine control and considering the involved uncertainties, this assembly process can become more robust. To our knowledge, this research is the first step toward assembling parts using force fields. There are a few problems we would
like to investigate in the future.

**Does there exist a strategy with fewer stages that can assemble the two parts with force fields?** The assembly characteristics are critical here, since the two parts have to be in their assembly orientations and \( P_1 \) has to align to its assembly position before the last horizontal push. In our strategy, the process deals with translation control and rotation control separately. If translation and rotation control can be done simultaneously, the strategy will be shorten by one stage. However, dealing with rotation and translation separately makes control easier. Luntz *et al.* present similar force fields (push and curl fields) to control the translation and rotation of the parts separately [28]. Their work came out almost at the same time as ours.

**How can we model interactions between the two parts when placing them inside the same field?** Impact and friction between the parts are the two main interactions. We propose two assumptions: (1) the inserting impact between the two parts is perfectly plastic, which means once the impact happens, the two parts will move together. Other impacts are perfectly elastic. (2) there is no friction between the two parts except that there exists a large viscous force among the impact edges of the two parts (guaranting plastic impact). The impact edges are the edges that touch the other part and are not parallel to the last pushing direction.

**Does there exist other more realistic damping models?** It seems to us that Coulomb friction is more realistic. However, we have proved that Coulomb friction alone can’t explain the behavior of parts under potential fields, and experimental observation shows that the parts do indeed reach their desired configurations. Even though viscous damping can be used in theory and simulations, real damping is decided by the physical implementation. For arrays of air jets, since this is a contactless manipulation (there is an air flow between the parts and the plane), viscous damping seems to be a correct choice. This needs to be verified by Yim and Berlin.

**What if the two parts are in the same half plane at the beginning?** A part separation preprocessing step is needed to move the two parts to different half
planes. A dynamic push field may be useful, but more exact control is needed for the locality of the force field. In general, part separation is itself an important problem, which could be as hard as part assembly.

Under some abstractions and assumptions, we have tried to understand what can be done for part assembly using programmable force fields. We hope that this work will inspire people who implement force fields in hardware to move in certain new directions.
Appendix A

Implementation Issues Concerning the Simulators

The simulator is split into two components, the force field and the part, both represented as abstract C++ classes to allow for flexibility.

A.1 Force Fields

VecField is the abstract base class for the force fields. It specifies an interface that includes functions to update the field, display the field in Geomview, and retrieve the force at a grid point on the field. Specific fields must inherit from this class since only the display function is not pure virtual. RAGField implements the radial plus gravity force field, EllipticField implements the elliptic field, SkewField implements a skewed squeeze field and TimeField implements a test field that varies as a function of time. There are several others. Profiling has shown that most of the program execution time is spent in retrieving forces from the field, but that caching the values in memory and recomputing them at each timestep differ little in performance. Thus for smaller memory requirements, all values are recomputed. We suspect that in more mathematically complicated fields caching values to take advantage of locality will be a good performance solution.

A.2 Parts

A.2.1 File Format for Polygonal Parts

Polygonal parts (the only kind currently supported) are specified in files of numbers separated by whitespace. Figure A.1 shows a file used to describe a square in our
simulator. The first number is a scaling factor by which all of the vertices are divided. The next two numbers are the x and y coordinates of the COM of the part in its initial state. These numbers are given relative to the square with corners at \((\pm.5, \pm.5)\). The forth number specifies the initial orientation. The part is rotated counterclockwise about its COM by this number of radians. Each subsequent pair of numbers specifies a vertex, in counterclockwise order.

\[
\begin{array}{cccc}
40 & 0.1 & -0.23 & 2.0 \\
1.0 & 1.0 \\
-1.0 & 1.0 \\
-1.0 & -1.0 \\
1.0 & -1.0 \\
\end{array}
\]

Figure A.1: A file for a polygonal part.

A.2.2 Algorithm of Polygon Filling

A polygon filling routine was implemented in the software. It is based upon the fast polygon filling algorithm presented in [10], although the handling of special cases comes from [14]. Thus horizontal lines are not considered and the endpoint of each line segment with the greatest \(y\) coordinate is ignored. The algorithm first computes the required size of the table by calculating the number of times the boundaries go from positive slope to negative slope and from negative to positive. This yields an upper bound of the number of zero that can cross a single scan line. Each boundary is then scan converted by choosing only those pixels inside the polygon. Each pixel here represents the center of a grid point. These pixels are then recorded in the table and the polygon is defined by those points inside starting and stopping numbers. For example, if the entry for the scan line \(y = 13\) is 1, 5, 8, 10 then all pixels between 1
and 5 inclusive and between 8 and 10 inclusive are inside the polygon. Each row of the table is sorted prior to the polygon filling.

**A.2.3 class Part**

Part is the abstract base class for the parts. It provides an implementation for the move routine (which specifies the physics model, either quasi-static or dynamic) and a specification for display, translate and force/torque calculation functions.

**class PolyPart** PolyPart is currently the only concrete instance of a part which implements the required functionality for polygonal parts. It represents polygons as a list of vertices, in counter-clockwise order. Since most of a parts functionality relies upon it representation, nearly all of a parts implementation appears in the class derived from Part.

When a PolyPart is created all of its information is read in from a file and placed into a list of vertices. The part’s COM is calculated and then the part’s vertices are translated so that the center of mass is at (0,0). Next the constructor calculates the parts moment of inertia about its COM. Coordinates of vertices are kept as double precision in the original frame of the part, where the center of mass is at (0,0).

**A.3 Other**

List, ListEnum, ListNode, Enum and DataType are classes that implement a list class in a Java-esque style, things which will be put in the list only need to extend ListNode. Tuple2 is a simple class to package two numbers and is generally used to describe a vector. The main function creates the force field, creates the specified parts, and places them in a list. Then the main function iterates through the parts, calculating the force and torque on each as it updates their position and displays them to Geomview via standard output. The loop continues until all parts have stopped moving.
Bibliography


