Thesis proposal: robotic origami folding

Devin J. Balkcom
Carnegie Mellon Robotics Institute
Pittsburgh, PA 15213
devin@ri.cmu.edu

November 15, 2002

Committee
Matthew T. Mason (Chair)
James Kuffner
Doug James
Jeff Trinkle (Sandia National Labs)

Abstract

I will present origami folding as an exciting challenge problem for the field of robotic manipulation. The problem is familiar, but also challenging – an origami design can be described as a flexible closed chain with a large number of degrees of freedom. Through an exploration of origami folding, my thesis will give insight and provide a partial solution to some very hard manipulation problems.

Contents

1 Introduction **4**

2 Problem statement and scope of the thesis **4**

3 Related work **5**
3.1 The geometry of paper 6
3.1.1 Developable surfaces 6
3.1.2 Geometry of creases 8
3.1.3 Modelling paper with natural creases 10
3.1.4 Origami mathematics and simulation 10
3.2 Models and simulation of flexible objects 11
3.2.1 Rigid multibody dynamic simulation 11
3.2.2 Cloth simulation 12
3.2.3 Haptic simulation 13
3.2.4 Fourier models 14
3.2.5 Continuum models 15
3.3 Planning 16
3.3.1 Planning for flexible objects 16
3.3.2 Planning for closed chains 17
3.4 Applications and control 19
3.4.1 Rope handling 19
3.4.2 Wire bending and insertion 20
3.4.3 Manipulation of fabric 20
3.4.4 Grasping of flexible objects 20
3.4.5 Sheet metal bending 20
3.4.6 Box folding 21

4 Overview of completed work **23**

5 Experimental work **23**
5.1 Tool design 24
5.2 Folding the paper 24
5.3 Bending the paper 26
5.4 Placing shallow creases 28
5.5 Folding creased paper 28
5.6 Folding an envelope 29
5.7 Evaluation 29
5.7.1 Manipulation primitives 29
5.8 Proposed thesis work 35
5.8.1 Design of a new folding mechanism 35
5.8.2 Grasping origami 36

6 Modelling one-dimensional paper **36**
6.1 Formulation of the model . . . 37
6.2 Potential energy functions . . . 39
6.3 Differential kinematics . . . 39
6.4 Kinetic energy and dynamics . . . 41
6.5 Force control . . . 42
6.6 Evaluation . . . 43

7 Rigid origami **43**
7.1 Origami with struts . . . 45
7.1.1 The minimum number of struts to make a facet rigid . . . 46
7.1.2 Grübler, for origami . . . 47
7.2 Topology of the configuration space . . . 48
7.3 Evaluation . . . 52
7.4 Proposed thesis work . . . 52

8 Simulation and planning **53**
8.1 Formulation . . . 54
8.2 Moving on the surface . . . 54
8.2.1 Euler steps . . . 54
8.2.2 Normal steps . . . 55
8.2.3 Efficient calculation . . . 55
8.3 Planner implementation . . . 55
8.3.1 Joint limits . . . 56
8.3.2 Facet collisions . . . 57
8.3.3 Pruning using a hash table . . . 57
8.4 Evaluation . . . 58
8.4.1 Limitations of struts . . . 58
8.4.2 Keyholes . . . 58
8.4.3 Step sizes and a heuristic . . . 59
8.5 Proposed thesis work . . . 59

9 Summary of proposed thesis work, and timeline **59**

10 Conclusion **62**

11 Acknowledgments **62**

1 Introduction

The goal of my thesis is to explore the manipulation of flexible objects, particlarly paper. The problem of manipulating flexible objects has been largely ignored by the robotics community. Putting on a shirt, reading the newspaper, and tying shoes are everyday tasks far beyond the capabilities of most robots. Origami will provide a focus and a motivating problem for the new field of robotic manipulation of flexible objects.

There are two reasons that folding origami is a good focal problem. The first is the complexity and richness of the behavior of paper. Origami paper is flexible, and stores energy like a spring. A good model might have a large or infinite number of degrees of freedom. Paper also has some interesting geometric properties. For example, it is not possible to perfectly wrap a piece of paper around a globe without stretching the paper. Creased paper behaves differently than uncreased paper. We might model the creases as joints, and the uncreased regions as rigid bodies. If the creases cross, it turns out that the modelled mechanism is a closed kinematic chain.

Origami therefore provides motivation to consider some important problems in robotic manipulation that are not well understood. How do we plan foldings motions for a complicated closed chain? How many 'hands' are needed to manipulate a closed chain, and how should it be grasped? What if the links are flexible, and springy?

The second reason that origami makes a good focal problem is that it is familiar and easily described: fold a flat piece of paper into a given shape. Origami books provide thousands of origami designs. Origami instructions also describe the order in which creases and folds should be made. Human 'origamists' provide a standard by which to judge success, and their techniques may provide inspiration and intuition.

Understanding origami will also lead to a better understanding of tasks for which robots are already used. For example, one of the biggest sources of error in automated sheet-metal bending is the unmodelled 'droop' of the metal. Understanding when origami paper can be treated as an articulated rigid body, and when the flexibility must be modelled, may lead to a better understanding of how to solve this problem.

2 Problem statement and scope of the thesis

This section summarizes the work that I plan to complete as part of the thesis. The details and methodology will be discussed in later sections.

Statement of intent: *I will explore the problem of origami folding from the perspective of robotic manipulation.*

In my preliminary work, I have identified some 'manipulation primitives': basic skills which a robot must have in order to fold origami. Some of the primitives include *positioning*, *folding*, and *flipping* paper. These primitives are sufficient to fold a class of origami. More complicated origami requires more sophisticated skills. Folding a crane requires simultaneously folding along multiple creases, and folding a balloon typically involves bending the paper. Typically, origami foldings that include pre-creasing steps will require more complicated manipulation skills than simple folding and flipping of

the paper.

The nature of my thesis will be exploratory. The exploration will have three components, and will be guided by the manipulation primitives. The *experimental* component will focus on the implementation of simple primitives: positioning, folding, and flipping. The *analytical* and *planning* compononents will focus on more complicated skills.

- *Experiments.* I will design and build a mechanism to fold simple origami shapes. I have used an Adept arm to fold a simple envelope in the lab; I will build a mechanism to fold slightly more complicated shapes more accurately.

- *Analysis.* I will focus on a model that considers origami to be an articulated rigid body. As will be discussed below, folding a crane requires manipulating links of a closed chain. The primary goal of this component of the exploration will be to understand the skills required to manipulate the closed chains that arise in rigid origami. In my preliminary work, I have analyzed the topology of the configuration space of the simplest origami design involving closed chains. The thesis will explore the topology of more complicated origami designs. A secondary goal of the work will be to find ways of classifying origami by *manipulation complexity*. How many hands are needed to fold an origami design? What manipulation skills are needed?

- *Planning.* I will design and implement a planner that can find continuous foldings of a pre-creased piece of origami paper. My preliminary work shows that most plans require that the origami go through singular configurations. Therefore, understanding the topology of the configuration space will be important in the design of the planner. Although it may turn out to be difficult to enumerate important singularities automatically, instructions for folding origami include sequence information which may provide some hint about necessary via points in the trajectory. Because of the computational complexity of the problem, the complexity of origami shapes for which the planner is applicable will be very limited, probably on the order of five to ten creases.

In order to provide a bridge between the theoretical and experimental work, I will also consider the problem of grasping origami. I will design an algorithm that will place fingers to quasistatically immobilize rigid origami. I will evaluate the effectiveness of the generated grasps experimentally.

3 Related work

I have chosen to focus on related work in a few areas: the geometry of paper, some examples of simulation of flexible systems, manipulation of flexible objects, and manipulation involving folding or bending.

3.1 The geometry of paper

3.1.1 Developable surfaces

Paper stretches much less than materials like cloth or sheet metal, and assuming that it does not stretch at all may be a useful approximation. If paper does not stretch, the class of shapes it can assume without creasing is restricted – wrapping an initially flat piece of paper onto the surface of a sphere is impossible. The possible shapes are called *developable surfaces*. We will need some concepts from differential geometry to describe the characteristics of developable surfaces; Thorpe [51] is my reference for basic differential geometry, and Hilbert and Cohn-Vossen [20] has an extensive section on the properties of developable surfaces.

An *isometry* between two surfaces is defined as a continuous bijective mapping that preserves dot products of tangent vectors. Since lengths of paths and areas of regions on a surface are defined in terms of dot products of tangent vectors, isometries preserve length and area.

The simplest isometries are rigid body transforms (rotations and translations), but there are more complicated isometries. If an initially flat paper cannot stretch, then there must be an isometry between the flat paper and any uncreased configuration of the paper. A path drawn on the flat piece of paper will have the same length along the bent piece of paper, and a rectangle will have the same area.

Even if there is an isometry between two surfaces, it may not be possible to smoothly transform one surface into the other. A *bending* between two surfaces is a one-parameter family of isometries that continuously deforms one surface into the other. Consider a knotted piece of string. Glue the ends together to form a knotted loop. Although there is an isometry between the knotted loop and an unknotted loop, there is no bending between the two configurations that avoids self-intersection – the knot cannot be removed.

A property of a surface is said to be *intrinsic* if it is preserved under isometries. *Geodesics* are curves on a surface that have no component of acceleration tangent to the surface. The shortest paths on a surface are geodesics; on a flat piece of paper the geodesics are straight lines. Since isometries preserve length, it is not surprising that geodesics are intrinsic. So, the curves created by drawing straight lines on a flat piece of paper and then bending the paper are geodesics on the bent paper.

The *Gauss map* takes all of the (unit) normal vectors of a surface to the origin. Since all the normal vectors are of unit length, the image of a surface under the Gauss map must fall on a unit sphere centered on the origin. This unit sphere is called the *Gaussian sphere*. The image is called the *spherical indicatrix*.

If we draw a triangle around a point p on a surface, the triangle encloses some area on the surface; call this area a. The image of the region within the triangle under the Gauss map has an area on the Gaussian sphere; call this area g. We define the *Gaussian curvature* G at p to be the limit of the ratio of these two areas as the area of the triangle goes to zero.

$$G(p) = \lim_{a \to 0} \frac{g}{a} \tag{1}$$

There is another way to find the Gaussian curvature. If we take the intersection

of a surface with a plane that includes the normal at p, we get a plane curve which we call a *normal section* (or slice) at p. Define the *principle curvatures* at p to be the maximum and minimum curvatures of the normal sections, evaluated at p. The Gaussian curvature at p is the product of the two principle curvatures at p.

Surprisingly, Gaussian curvature is an intrinsic property of a surface. (Gauss' *Theorem Egregium* [15].) The Gaussian curvature of a plane is zero, since both principle curvatures are zero, and since the Gauss map takes the entire surface to a single point of zero area on the sphere. Since there is a local isometry between any developable surface and the plane, the Gaussian curvature of a developable must also be zero everywhere. For example, a piece of paper can be folded into a circular cone. At any point on the paper, one principle curvature is zero (along the line from that point to the vertex), and the other is equal to the curvature of the great circle on the cone containing the point. Since one principle curvature is zero, the product must be zero; Gaussian curvature is preserved.

If the Gaussian curvature is zero, then at least one of the principle curvatures must also always be zero. From this it is possible to show that developable surfaces are *ruled surfaces*; through any point in the surface there is a line segment (a *ruling*) contained in the surface and extending to the boundaries of the surface. However, not all ruled surfaces are developables: a developable may be defined as a ruled surface for which the tangent plane is the same at any point along a line embedded in the surface. This gives an additional way to describe developable surfaces – as the envelope of a one-parameter family of tangent planes.

A number of authors have used the geometric properties discussed to derive representations of developable surfaces. Redont [44] used the zero-curvature property as well as the fact that geodesics are intrinsic in order to show that a developable can be described by a parameterized path on the Gaussian sphere. Since the path on the Gaussian sphere gives the normals to the surface, the formulation is in terms of an ordinary differential equation, together with an initial condition. Although the differential equations usually cannot be solved analytically, Redont points out that if the trajectory on the Gaussian sphere is a circular arc, then the developable is a segment of a circular cone. Redont therefore proposes a method of approximating developable surfaces using C^1-connected circular arcs on the Gaussian sphere. Thus, the class of surfaces considered are composed of segments of right circular cones.

Sun and Fiume [49] used a representation similar to Redont's to build a geometric modelling program. The authors used their software to create models of a hanging scarf and of a bow made out of ribbon. Leopoldseder and Pottmann [31] have also explored the problem of approximating developable surfaces by right circular cones. They point out that one difference between their work and Redont's is that they are concerned primarily with approximating local properties of the general developable surface, whereas Redont's algorithm is global in nature.

Pottmann and Wallner [43] also propose an alternate representation of developable surfaces, based on the definition of a developable surface as the envelope of a one parameter family of tangent planes. Since four numbers can be used to represent a plane using homogenous coordinates, there is a duality between developable surfaces and trajectories in projective Cartesian space. The authors present metrics in the dual space, and use this to derive a method for approximating a set of tangent planes with

developable surfaces of a certain class.

Weiss and Furtner [60] considered the problem of finding a developable surface that connects two space curves. The rulings of the developable are used to connect the curves. However, arbitrarily connecting the two curves by rulings will yield a ruled surface, but not necessarily a developable; the additional constraint is that the tangent plane to the surface must be the same at each point along the ruling. The authors propose a metric measuring the extent to which the four endpoints of two adjacent rulings are co-planar. An iterative algorithm generates appropriate rulings, and thus a polyhedral approximation of a developable surface connecting the two curves.

Aumann [4] presents an important extension of Weiss and Furtner's work. Two general curves cannot always be connected by a developable – bending the edges of a pieces of paper into certain shapes will lead to crinkling and creasing of the paper. Aumann considers the special case where the two curves to be connected are Bézier curves, and determines necessary and sufficient conditions for the interpolating developable patches to exist and be free of singularities.

3.1.2 Geometry of creases

The work on developable surfaces presents a detailed picture of the shapes a piece of paper can be bent into without creasing. But what happens if we crease the paper? Huffman studied this problem in [23], also from a geometric perspective. Huffman's motivating application was scene analysis. One goal of the work was to extend the generality of the models that could be considered in computer vision. Huffman wrote,

> Objects bounded by planes were reasonable ones upon which to do initial research in scene analysis... No two neighboring points on an arbitrary surface need have the same tangent plane. By contrast, all points on a plane surface have the same tangent plane. On a developable...all points on a given line embedded in the surface have the same tangent plane... [A] paper surface offers a complexity that is, therefore, in a very real sense exactly midway between that of a completely general surface and that of a plane surface. Consequently, paper surfaces constitute a class that may be ideally suited to be both richer than that of plane surfaces and more tractable analytically than that of totally arbitrary surfaces.

Huffman first examined the simpler problem of polyhedral vertices. Consider the vertex of the cube shown in the upper left of figure 1. The Gauss map takes each of the three faces to a point on the Gaussian sphere. We may consider the dihedral angles of the cube to correspond to edges connecting these points. Thus the Gauss map of the area of the surface enclosed by a small loop around the vertex is a triangle on the Gaussian sphere, shown in the upper right of figure 1. As the area of the loop shrinks to zero, the triangle on the Gaussian sphere is constant. Therefore, the Gaussian curvature at a vertex of a cube is infinite.

If we assign a direction to the loop around the vertex, we can associate a direction with each edge of the triangle on the Gaussian sphere. We consider the area of the triangle to be positive, since the area is enclosed in a clockwise fashion. (Area enclosed in a counterclockwise fashion would be considered to be negative.) The area of the

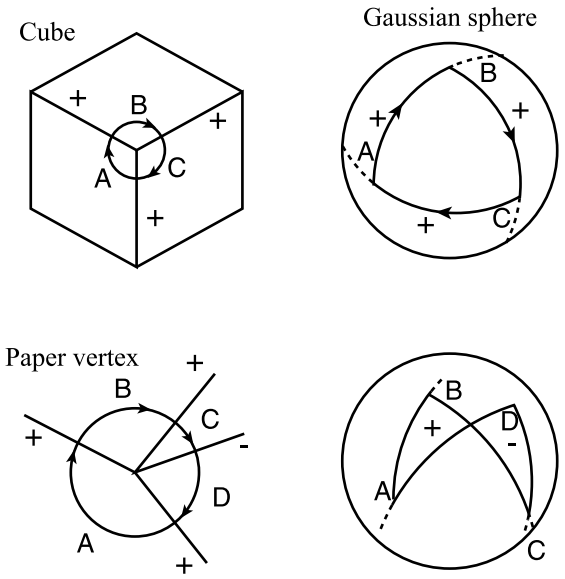


Figure 1: Polyhedral vertices on the Gaussian sphere. Re-drawn from [23].

triangle is $+\pi/2$. If the cube were cut along an edge and laid flat, the 'missing angle' would also be $+\pi/2$. In fact, a similar observation is true for all polyhedral vertices – the area of the region enclosed on the Gaussian sphere is equal to the 'missing' angle (positive area) or 'excess' angle (negative area) if a cut were made from the vertex along an edge and the facets were laid flat.

If we form a vertex by creasing paper, there will be no missing or excess angle. Huffman analyzed the simplest interesting example of a case where the area on the Gaussian sphere is zero; this example is shown in the bottom of figure 1. Since three creases intersecting at a point will always lead to a triangle on the Gaussian sphere (with non-zero area), the simplest example must have four creases. Furthermore, the configuration must be such that either three of the creases are convex, and one concave, or vice versa.

The above discussion assumes that the faces of the polyhedron are rigid. However, if the faces are actually parts of the paper where there are no creases, the faces should be permitted to bend. For example, if a pieces of paper contains only three creases which intersect at a point, then any non-flat configuration of the paper will require that the uncreased portions bend. In this case, each facet is a developable surface, and can be represented by a curve on the Gaussian sphere. Finally, Huffman considered the case of curved creases. The shape of the curve places a restriction on the orientations of the nearby tangent planes.

3.1.3 Modelling paper with natural creases

Sometimes paper is creased intentionally, and sometimes creases occur because there are constraints applied that are inconsistent with the paper remaining a smooth developable surface. We will call creasing of the second type *natural* creasing. Kergosien *et al* [27] (*Bending and Creasing Virtual Paper*) is the only work that I know of that combines the work on developable surfaces with a model of natural creasing.

In spirit, the work is most similar to that of Weiss and Furtner [60] and Aumann [4]. The location of rulings was used to describe the uncreased sections of paper; the locations of rulings were parameterized along a trajectory around the edge of each section. The locations were discretized and all bending was assumed to occur along rulings. It was shown that there is a linear constraint between the positions of the rulings and the 'developability' of the surface (the extent to which rulings share a common tangent plane at each endpoint). The authors implemented a graphical interface that allowed the user to apply spring forces to the surface. The forces typically deformed the object in a way that violated the constraint that the surface was developable. A constraint projection step was then used to ensure developability.

When the rulings of the paper began to cross, a creasing model was triggered. Thus, the paper was described by a data structure containing both developable patches and creased regions. The authors point out that the creasing patterns that may be used are non-unique; they studied both point creases and short line creases. The specific choice of crease type was heuristic; placing the creases was posed as an optimization problem and solved using sequential quadratic programming.

3.1.4 Origami mathematics and simulation

There is a rich field of work on the mathematics of origami design. Demaine *et al.* [12] is a good survey. According to [12], the field "essentially began with Robert Lang's work on algorithmic origami design, starting around 1993." Given a desired *origami base* (an origami silhouette from a restricted class of shapes) Robert Lang's *TreeMaker* software (described in [28]) finds a crease pattern allowing the paper to be folded into the base.

Demaine *et al.* [12] classifies work in computational origami as *universality results*, *efficient decision algorithms*, and *computational intractability results*. As an example of universality results, the authors state that "any tree-shaped origami base, any polygonal surface, and any polyhedral surface can be folded out of a large enough piece of paper". As an example of an efficient decision algorithm, "there is a polynomial time algorithm to decide whether a ... grid of creases marked mountain and valley can be folded by a sequence of simple folds." Intractability results include that the problem of determining whether a crease pattern can be folded flat is NP-hard, and that "given a crease pattern... finding the overlap order of a flat folded state is NP-hard".

Miyazaki *et al.* [39] describes software that allows the folding of "virtual" origami by the user. The origami is treated as a collection of rigid facets connected by hinge joints. Two basic primitives are designed: *folding* and *tucking in*. During folding, facets rotate around a single hinge joint. During tucking, a pair of facets connected by a hinge joint is reflected through a plane perpendicular to the facets. The authors were

able to use their system to virtually fold a crane and a paper airplane.

3.2 Models and simulation of flexible objects

The work on developable surfaces and creasing is concerned with what shapes paper (or other developables) *might* take, assuming that paper does not stretch. If there are enough additional constraints, this may be sufficient to determine the shape of the paper kinematically – for the purposes of high level motion planning, we could treat a piece of paper stretched tightly over a drumhead as a rigid body. Typically, however, the configuration of paper is determined by internal spring forces as well as external forces and constraints.

The simplest way of modelling these internal forces is to attach discrete springs to various parts of the flexible body. Another possibility is to use a constitutive law describing the potential energy as an integral of a continuous function representing the shape of the flexible object.

3.2.1 Rigid multibody dynamic simulation

Many of the most successful methods for simulating flexible objects treat the flexible object as a collection of a large number of rigid bodies. Therefore it is appropriate to briefly discuss efficient simulation of high-DOF rigid body systems.

The simulation techniques may be broadly classified as two types: those with implicit constraints (joint space, or generalized coordinates), and those with explicit constraints. The former techniques may make use either of Lagrangian or Newton-Euler formulations of the dynamic equations; the latter introduce Lagrange multiplier forces to maintain the constraints.

Craig [11] points out that the first algorithms used to simulate the dynamics of robot arms used a straightforwards Lagrangian formulation. The approach relied on calculating and inverting the (dense) mass matrix, and was $O(n^4)$ in the number of links. The first efficient ($O(n)$) algorithms were based on recursive Newton-Euler formulations of the dynamics, and efficient Lagrangian formulations were also eventually developed [11].

Lagrangian representations with implicit constraints typically consider dynamics equations of the form

$$M\ddot{q} = f \tag{2}$$

where M is the mass matrix, q is the configuration of the system, and f is the vector of external and velocity-dependent forces. The structures of M and f depend on the choice of coordinates for q, and the choice of coordinates ensures that the constraints are always satisfied.

Unfortunately, it may be difficult to determine appropriate generalized coordinates to represent the constraints. Another difficulty with these approaches is that some choices of generalized coordinates lead to a system which is difficult to simulate efficiently. Fortunately, efficient dynamic simulation techniques that allow explicit constraints have also been demonstrated. In this case, the system consists of an equation

similar to that of equation 2, together with a constraint equation of the form

$$g(q) = 0 \tag{3}$$

Since the constraint equation is satisfied at each time, the time derivative is also 0:

$$\frac{d}{dt}g(q) = J(q)\dot{q} = 0 \tag{4}$$

where J is the matrix of partials known as the *constraint Jacobian*. Simulation techniques which allow explicit constraints typically involve writing a matrix equation involving the external forces, the mass matrix, the constraint Jacobian, and the Lagrange multipliers (constraint forces). Since the mass matrix and the constraint Jacobian are sparse, sparse matrix methods can be used to efficiently solve for the Lagrange multipliers. Gleicher [16] used a conjugate gradient method; Baraff [6] presents an algorithm which is $O(n + m^3)$, where n is the number of links and m is the number of closed loops – the method is linear if their are no closed loops.

Baraff's algorithm is linear in the number of links, but cubic in the number of closed loops. Ascher and Lin [3] present an algorithm which is linear even if there are a large number of closed loops. The algorithm breaks each closed loop to form an open chain. At each time step, the open chain is simulated using a method similar to that used by Baraff [6] and others. An iterative method is used to satsify the loop closure constraints. It is shown that the number of iterations is independent of the number of links, although it is dependent on the topology of the configuration space. The algorithm was applied to simulate the dynamics of a four-connected mesh with up to 1000 links (25×20). In each case only two constraint satisfaction iterations per time step were required. Simulations were conducted with a time step of $.2$ seconds, for a simulation time of a few seconds. No information on run-time was provided.

It is interesting that in addition to models that approximate flexible systems by high degree of freedom systems of rigid bodies, there are also examples of approximating discrete high degree of freedom rigid body systems by continuous models – for example, Minksy's elephant trunk arm [38] and Chirikjian's work on the inverse kinematics of high-DOF binary manipulators [9].

3.2.2 Cloth simulation

Impressive dynamic simulations of cloth have been achieved by modelling the cloth as a high-DOF system of rigid bodies. Some of the most notable recent techniques are presented by Choi and Ko [10], and by Bridson *et al* [8]; these papers also present a more complete survey of previous cloth simulation work than is possible here. [10] focuses on the internal dynamics of the cloth, and [8] concerns itself primarily with problems of contact and friction.

Most cloth simulation algorithms do not place hard constraints on the relative motion of particles or small facet elements; rather, stiff springs are used to keep the cloth from stretching very much. As an example, figure 2 shows the configurations of springs and particles used by Choi and Ko [10]. Typically, these stiff springs introduce instability into the dynamic simulation. The usual solution, proposed first by Baraff and

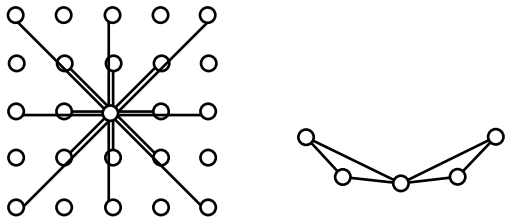


Figure 2: Connections between a central particle and its neighbors used by Choi and Ko [10]. Top view (left) and side view (right). Re-drawn from [10].

Witkin [7], is to use implicit integration techniques. Implicit integration formulations may also ameliorate problems related to stiffness in the differential equations due to contact and friction; for example, see Stewart and Trinkle [47]. (For a discussion of implicit integration methods, see the referenced papers.)

Since there are no constraints, and since the only direct interactions between particles of the cloth are local, the mass matrix is sparse. Iterative sparse matrix methods (*e.g.*, conjugate gradient) are therefore used to solve the dynamic equations efficiently. External contact forces due to collisions, friction, and self-intersection are usually handled by attaching virtual springs at contact points.

3.2.3 Haptic simulation

Cloth simulations do not typically run in real-time; even the most efficient algorithms require minutes or hours to create a realistic-seeming dynamic simulation. Therefore, motion planning or control of flexible objects using these algorithms is problematic.

A different perspective on simulation of flexible objects is provided by a number of papers on haptic simulation; James and Pai [25] provides a good survey. Unlike the cloth simulation algorithms, haptic simulation algorithms typically use a quasistatic model. That is, it is assumed (or proven) that if forces are applied to a flexible object, then it will eventually reach an equilibrium configuration. *Green's functions* relate displacements of the material to forces applied.

James and Pai represent the surface of a flexible object by a set of discrete points, or nodes. Each node is either fixed in space, or has forces applied to it. The set of nodes which are fixed describes the problem type. (Typically, the base of a flexible object might be fixed, while the remainder of the nodes would be free. Poking at the object with a finger would introduce a new spatial constraint, and change the type.)

The authors consider linear models; models for which (by definition) there is a linear relationship between displacements and applied forces. If the model is linear, there is a linear basis for the Green's functions. This basis can be computed off-line. Once the basis has been computed, simulation can be carried out quickly by simple matrix multiplication.

The linear model also allows efficient re-use of computation if the type of the problem changes slightly (for example, if a finger pokes the object). Computing the response of the system to a set of forces or constraints requires a matrix inversion. If the

type of the problem is known, this inversion can be done off-line. If the type changes slightly, *capacitance matrix algorithms* can be used to efficiently update the inverse.

3.2.4 Fourier models

Hirai *et al* [22] modelled the shape of a cross section of a piece of bent (but not creased) paper. Fourier coefficients were used to describe the shape of the paper. (This technique is described in more detail in section 6.) The model was used to find equilibrium configurations of the paper under a set of geometric constraints. The authors write an equation for the potential energy in terms of the Fourier basis coefficients, and use non-linear optimization software to minimize the energy. They used five coefficients, and demonstrated experimentally that the model predicted the behavior of a piece of copy paper reasonably well, for some simple examples. In [54] a similar approach was used to model yarn in a knitted piece of fabric.

One difficulty of the method described is the problem of bifurcation. There may be two or more possible configurations of the paper consistent with a given set of geometric constraints. Non-uniqueness of solutions is a familiar problem in physical simulation algorithms. For example, it is well-known that the problem of determining the accelerations of two contacting rigid bodies under the Coulomb friction assumption may have no solutions, one solution, or many solutions (see Painlevé [42], Lötstedt [32], Erdmann [13], and Stewart [48]).

There are various approaches to dealing with the problem of non-uniqueness of solutions for the purposes of simulation. The simplest is to modify the assumptions so that the solutions are unique; this is the approach taken by Lötstedt [32], and Anitescu and Potra [1] in designing their rigid-body simulation algorithms. For the purposes of planning, analysis of the model to determine when the solutions are non-unique or non-existent may allow plans to be generated that are guaranteed to work in spite of the uncertainty; for the rigid body contact planning problem, this is the approach taken by Erdmann [13], Trinkle *et al.* [53], and Balkcom and Trinkle [5].

Wada *et al.* [57] extended the Fourier model developed in Hirai *et al* [22] to the case of a rod in three dimensions, and considered dealing with the bifurcation problem by using an optimizer that tends to find local rather than global potential energy minima. This approach is not really satisfactory, since it is not clear what metric should be used to decide which minima are 'close' and which are 'far'; I expect that some modelling of dynamics is necessary to determine which of several local minima the system eventually reaches.

Wakamatsu *et al.* [59] considered the problem of simulating the dynamics of rod-like objects. If the system is conservative, then the trajectories it follows must minimize the integral of the difference between kinetic and potential energies, subject to the constraints. The authors wrote equations for the kinetic and potential energies as functions of Fourier basis coefficients that were used to approximate the rod's configuration and velocity. They then used non-linear optimization software to solve for the accelerations at each discretized time step.

There is an interesting connection between the problem of finding minimum energy configurations of a rod and finding optimal (or near-optimal) trajectories for robots. Although the application is much different, the algorithm proposed by Hirai *et al.* [22]

appears to be identical to that used by Fernandes, Gurvits, and Li in *A Variational Approach to Optimal Nonholonomic Motion Planning* [14].

The Fourier model just discussed uses a finite number of variables to approximate the configuration of a flexible rod. This method is similar to classical techniques used to analyse vibration. Symon's *Mechanics* [50] describes the configuration of a vibrating string by an infinite Fourier series. In this case, the series describes the x and y coordinates of each particle, rather than the angle of the tangent. As a result, there is no implicit arc length constraint, and the string can stretch. The dynamics are modelled, and the frequency of vibration is determined analytically.

3.2.5 Continuum models

There is a vast field of research on elasticity and the mechanics of continua. Only the briefest summary is possible here; Antman [2] provides a good survey. Typically, the configuration of the flexible object is represented by a parameterized function. Constitutive laws are formulated to describe the local behavior of the material. Potential and kinetic energy are described as functionals. Various techniques are then used to analyze the behavior. For example, minima of the potential energy functional correspond to equilibrium states; variational approaches attempt to solve for the equilibria directly, or, when that is not possible (the usual case), used to find properties of the equilibria (*e.g.* bifurcations points, geometric properties, *etc.*).

The method of Lagrangian dynamics can also be extended to objects whose configurations are described by continous functions. Chapter 13 of *Classical Mechanics* by Goldstein *et al.* [17] describes this technique in detail. It turns out that Lagrange's equations yield partial differential equations describing the dynamics, rather than ordinary differential equations.

What if the flexible object is thin, like paper or string? Pai [41] considered the problem of simulating thin elastic solids that both bend and twist. Pai points out that "modelling these [objects] as 3D elastic solids requires very fine FEM meshes to correctly capture the global twisting behavior…models using meshes of mass particles and springs have similar problems since they require a large number of particles and springs…"

Pai uses a *Cosserat* model to describe the behavior of a thin elastic rod that can twist – a *strand*. The configuration is described by the trajectory of a frame of three *directors*. One of the directors is the tangent vector to the strand; the other two describe the twisting. If the trajectory is of constant speed, then the strand may bend but does not stretch. Pai formulates (ordinary) differential equations describing equilibria for the case where one end is fixed and force is applied at the other end. He discretizes the equations, and presents a linear-time algorithm to solve the discretized equations.

Cosserat models also exist for shells and points, as well as for rods (strands). Antman [2] and Rubin [45] are standard sources. Rubin points out that an advantage of modelling flexible objects using thin, directed media is that thinness may simplify the form of the dynamic equations. In tabular form,

Model	Dynamic equations
Cosserat point	ODE in time
Cosserat rod	PDE in time, and in one spatial coordinate
Cosserat shells	PDE in time, and in two spatial coordinates
3D elastic	PDE in time, and in three spatial coordinates

Since ODEs are easier to solve than PDEs, Rubin also presents a number of methods to numerically simulate Cosserat rods, shells, and general 3D elastic materials using a collection of points.

3.3 Planning

3.3.1 Planning for flexible objects

Most work involving flexible objects has focused on dynamic simulation and control of flexible objects; there is relatively little work on the problem of manipulation planning for flexible objects among obstacles. The lack of work in this area may be due to the difficulties inherent in applying traditional planning techniques to flexible systems. Some of the problems involve:

1. **Dimensionality:** Models of flexible and continuously deformable objects typically use a large number of variables to approximate the possible configurations of the system. The combinatorics of search algorithms makes planning in high-dimensional spaces infeasible without strong heuristics.

2. **Predictability and Repeatability:** Modelling flexible objects is difficult in itself, and most models do not adequately capture all aspects of the system. Over short time intervals, flaws in the model may not become apparent, and closed-loop control can ensure that the model is synchronized with the world. However, planning is intrinsically an open-loop process.

3. **Underactuation:** Ultimately, only a few controls will be available to manipulate a flexible object. How should a flexible object be grasped? Should the planner operate in the lower-dimensional control space, or in the higher-dimensional configuration space? If the planner operates in the control space, there is an additional difficulty: the configuration may be dependent not only on the current grasp configuration, but on the entire history of grasp configurations.

Kavraki *et al* [26] explored the problem of planning the motions of a thin, rectangular, elastic plate among obstacles. The plate was modelled as a Bézier surface with control points. A potential energy function was defined over the location of the control points. The plate was assumed to be grasped by two opposite edges. Once the configurations of the edges had been determined, a conjugate gradient method was used to find a minimum of the energy function. A table of minimum-energy configurations was precomputed.

The planner was a probabalistic roadmap planner. Each random configuration was generated by fixing an edge, randomly selecting the configuration for the other grasped

edge, finding a minimum-energy configuration from the pre-computed table, and randomly selecting a rigid transformation to apply to the body.

The approach was tested using two hundred pre-computed minimum-energy configurations. A problem involving moving the plate through a triangular hole was solved with an average running time of about three minutes.

An alternate method was also explored, in which it was assumed that the Bézier control points could be directly manipulated to determine a configuration; the potential energy function was used only to exclude 'unreasonable' configurations. In this case, the problem was of much higher dimensionality (26 DOF). The planner took about five hours to explore the space of solutions for a problem involving bending the plate and moving it through a U-shaped hole.

3.3.2 Planning for closed chains

Most origami designs involve creases that cross. When creases cross, the mechanism contains closed chains. The problem of motion planning for closed chains is difficult; no complete and general algorithms have been implemented. In this section I will discuss some promising approaches that have been applied to a limited class of problems.

One of the first successful applications of randomized path planning techniques to problems involving closed chains is described by LaValle *et al.* [29]. Planar mechanisms with multiple closed chains were considered; the environment was assumed to contain planar static obstacles. The goal was to move some subset of the links to a target configuration.

Each kinematic loop was broken to form an open chain. Explicit equality constraints were used to describe the closure condition. During execution of the algorithm, random configurations of the open chain were generated. A randomized descent function used a configuration of the open chain as a starting point, and attempted to find a new configuration satisfying the equality constraints. Once a number of configurations of the closed chain had been generated, a randomized local planar was used to connect nearby nodes. The algorithm was applied to a ten-link chain containing one loop, and to a seven-link chain with two loops.

Han and Amato [19] presents another application of probablistic techniques to closed chains. The authors cite two primary differences between this work and the work of [29]. Closed chains are broken to form pairs of open chains. However, rather than writing explicit equality constraints, some of the chains were designated as active, and others were designated as passive. The configurations of the system were described by the configurations of the active chains, and inverse kinematics were used to find compliant configurations of the passive chains. The second difference is that a two-stage approach was used. The environment was first disregarded, and a roadmap describing the configurations of the mechanism alone was built. During the second phase, collision detection was performed on the roadmap. The authors report that these differences led to significant improvements in running time. The planner was applied to planar problems involving up to fifteen links, and three-dimensional problems involving up to eleven links.

Although the results achieved by [29] and [19] were good, there are some issues that the randomized planners described do not address. Choosing random configurations

for the open chain and then minimizing the error to find a closed chain configuration is problematic. As the authors point out, the probability that a generated configuration satisfies the equality constraint is zero. If there are a large number of connected closed loops, the local optimization algorithm is not likely to converge. In [19], these problems are dealt with by using inverse kinematics rather than optimization to find compliant configurations; the same problem seems likely to occur if there are a large number of links or closed chains.

Probably the most significant problem is the fact that the topology of the configuration space may be very complicated. For many closed chains, the configuration space may consist of a number of manifolds of a certain dimension, connected by manifolds of lower dimension. The probability of finding these lower dimensional 'keyholes' using random sampling is zero. It is also unclear which joints should be removed to form open chains. Even for a simple four-bar mechanism, there may be situations where no single pair of driving links can be found, since each pair of links may have to go through a singular configuration to reach the goal.

There are efficient algorithms that do not exhibit these problems in the case where there are no obstacles (or self-intersections) and there is a single closed chain. These approaches are based on global topological properties of the configuration space. Lenhart and Whitesides' [30] $O(n)$ algorithm determines a sequence of moves that transforms any planar closed loop into a triangle. The algorithm is applied to both the start and goal configurations. Then a sequence of moves is used to tranform one triangle into the other.

One difficulty with extending Lenhart and Whitesides' method to planning problems with obstacles is the requirement of an intermediate triangular configuration. If there are are obstacles, joint limits, or self-intersections, it might not be possible to achieve this configuration.

An alternate approach is suggested by Milgram and Trinkle [37, 52]. The algorithm first checks if the configuration space has one or two disconnected components. (These are the only two possibilities.) If there are two, it turns out that planning is easy; a subset of the links can be chosen as the driving links, and linear interpolation can be used to monotonically drive the system to the goal. If there is only one component, planning is more difficult. Two or more links are chosen, and driven to their final relative angle, while the remaining links comply. The relative angle between the chosen links is then fixed (a *linkage reduction*). This procedure is applied iteratively. It is shown that the algorithm is guaranteed to terminate, either when it is determined that the start and the goal are in disconnected components of the configuration space, or once the mechanism has been reduced to a four-bar linkage. In the second case, a complete four-bar planner is used to plan the trajectories of the final links. Topological properties of the configuration space were derived and used to prove that this planner is complete; some details of the methodology are discussed in more detail in section 7.2.

The planner described in Milgram and Trinkle [37, 52] is less efficient than that described by Lenhart and Whitesides ($O(n^3)$ vs. $O(n)$), but the planner was able to find plans for chains of one class with 500 links (in sixteen hours), and chains of the other class with $10,000$ links. The resulting plans do not require a fixed intermediate configuration. Therefore, although the authors do not consider self-intersections, joint limits, or obstacles, implementing a more general planner based on their method might

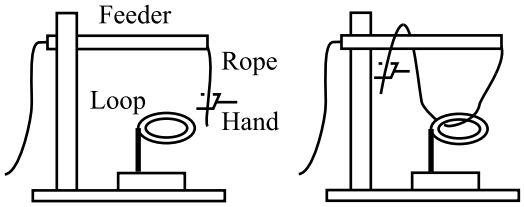


Figure 3: One of the first examples of robotic manipulation of a flexible object, redrawn from [24].

be feasible.

3.4 Applications and control

3.4.1 Rope handling

Inoue and Inaba's paper, *Hand-Eye Coordination in Rope Handling* [24], describes an experiment in robotic manipulation of rope. Figure 3 shows the task. The rope was fed from a horizontally positioned hollow tube. The robot grasped the rope, and pulled a length of it from the tube. The robot re-grasped the rope closer to the tube, pulled out more rope, and dropped the dangling endpoint through a wire loop. The robot then released the rope, and regrasped a point on the rope below the wire loop. It then pulled more of the rope through the loop, and laid the endpoint over the feeder tube.

Inoue and Inaba's system comprised a six degree of freedom arm, a stereo vision system, tactile sensors in the fingers, an arm controller, and a micro-computer running custom software. Although this paper appears to be the first published work describing a robot manipulating a flexible object, the focus is actually the design and architecture of this system – the rope handling was seen as a virtuoso demonstration of capabilities of the robot.

This focus on the system architecture defines the perspective with which the authors view the manipulation task. Surprisingly, visual servoing was seen as the only feasible approach.

> The task of rope handling seems simple. But it inherently requires intensive machine vision. So far, robots have handled only solid objects...a flexible object like a rope does not keep its initial shape and it changes during motion. Therefore, visual feedback is essential in order to manipulate flexible objects successfully.

How should we classify the model of flexible objects used in this work? Based on the visual servoing approach, the model of the system appears to have a dimension of nine: three numbers for the position of the rope endpoint, and six numbers for the configuration of the gripper. After each move, the robot waited for the dangling rope to reach equilibrium; thus the model is essentially quasistatic. Constraints were not considered in the paper; the plan was hand-crafted, and reasoning about the motion of the rope through the ring was made by the researchers, not the robot.

3.4.2 Wire bending and insertion

Nakagaki *et al* [40] considered the task of inserting a flexible wire into a hole. When force acts on the tip of the wire, some deformation occurs; if the deformation is entirely elastic, the wire returns to its original shape when the force is removed. There may also be some *plastic* deformation; once the wire is released, it may not return to its original shape. These authors use a Fourier basis model similar to that described above, together with a vision system and a force sensor at the wire tip to determine how much of the deformation is plastic, and how much is elastic. A gripper is used to remove the plastic deformation and straighten the wire. Visual servoing is then used to insert the wire into a hole.

3.4.3 Manipulation of fabric

If we grab the edges of a piece of paper and move them around, we expect to be able to indirectly control other points on the paper. This is an implicit assumption in most visual servoing approaches to the manipulation of flexible objects. Wada *et al.* [56, 55] explores this approach for the problem of manipulating a piece of fabric during automatic stitching. The authors model the cloth using a simple mass-spring system, and show that a small number of interior points can be controlled by moving a few corners of the cloth. They also show that if deformations are small, their visual servoing controller will still converge even without a model of the cloth. Hirai *et al.* [21] describes a similar technique for manipulating a grasped sponge.

3.4.4 Grasping of flexible objects

Wakamatsu *et al.* [58] considers the problem of grasping deformable objects. The authors point out that the principle of force closure is not really approprate for flexible objects. They propose that grasps of flexible objects can be evaluated using the idea of *bounded force closure* – can the current grasp resist a force of bounded magnitude in an arbitrary direction? The authors conducted some simulations involving the grasping of a thin rod.

3.4.5 Sheet metal bending

The problem of sheet metal bending has been studied by a number of authors. Possibly the most complete and practical system is that described by Gupta *et al* [18]. Their system consists of a high level planner that determines the sequence of bends, together with a number of low level planners. The low level planners provide 'primitive' actions, including:

- selecting appropriate bending tools and configuration
- inserting the part into the station
- removing positional error by gaging
- creating a bend

- removing the part from the station
- using a second gripper to allow re-grasping of the part

The hardware consisted of a robot arm (used to position the sheet metal), a gripper that was used for re-grasping, and a press brake that used a die and punch to create the bends. In fact, there are a large number of specialized tools which are used for sheet metal bending. The punch and die selection planner attempted to recognize features from the intermediate part shape, and used these features to select appropriate tools from a large database. A wide variety of grippers is also available; the system chose from a library of fifty different designs.

Each of the planners relied on kinematic models of the sheet metal and the press brake system. Most of the planners considered the sheet metal to be a single rigid body; the exception is the grip location planner, which also considered the fact that the gripped portion of the sheet metal may move during bending in the brake. Since the angle of the bends were chosen by a human designer, and all bends occured atomically, the top level planner needed to determine only the order of the bends, a discrete planning problem.

3.4.6 Box folding

Lu and Akella ([34],[35]) describes a planner which enumerates all collision free sequences to fold a carton blank into a carton. The carton blank is a flat piece of cardboard with creases separating the panels. The carton was modelled as a collection of joints (the creases) and links (the panels). The possibility that bends are not atomic was considered, so the configuration space was continous rather than discrete.

The model of the cartons was kinematic, and the number of degrees of freedom was equal to the number of creases; typically between five and nine for the problems considered. The model included both explicit constraints (self-intersection collisions) and implicit constraints (the configuration of the carton was described in joint space.)

The configuration space was represented by a recursive tree representation based on that used by Lozano-Perez [33] to model serial arms; some modification was necessary to take into the account the possibly branching sequences of links. Since creases in the cartons do not typically cross, the authors do not consider the possibility that the structure is a closed chain.

The intended use of the planner was to enumerate possible fold sequences to help a human design a carton-folding fixture. The authors applied the planner to a number of types of carton blanks, and designed a single fixture to fold two similar types of blank. They used a five-joint Adept arm to move the carton blanks through the fixture, and achieved good results – success rates were nine out of ten and nine out of sixteen for the two types. Figure 4 shows an example.

A similar problem was also explored by Song and Amato [46]. Because cartons may have many flaps, the dimensionality of the configuration space may be very high. Fully exploring the configuration space therefore may be prohibitively expensive. The authors designed and implemented a probabalistic roadmap planner, and applied it to the problem of folding a carton with twelve creases. The planner was also successfully applied to the problem of folding protein molecules with about one hundred degrees of

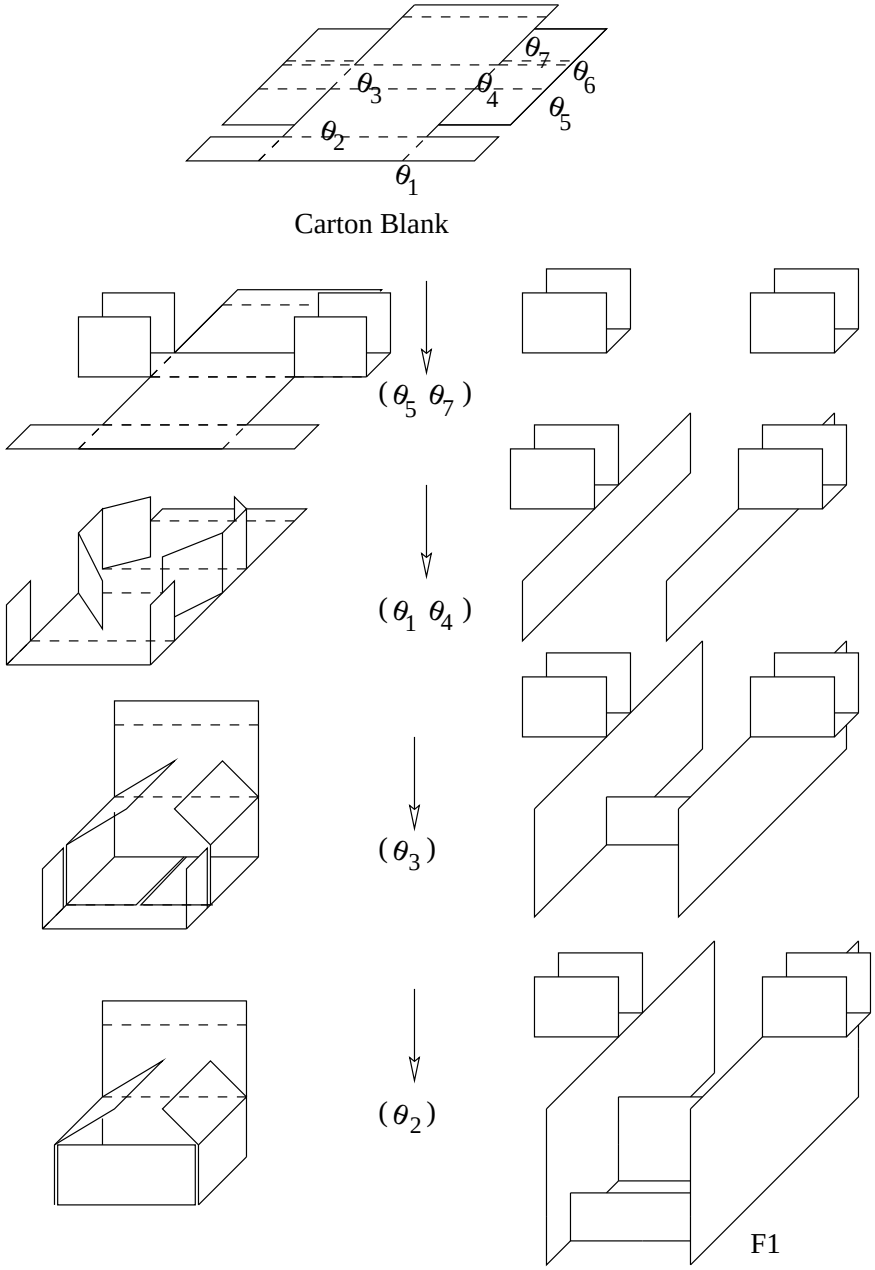


Figure 4: Folding a box using a fixture. Reprinted from [34] by permission.

freedom, and to a paper folding problem with eleven degrees of freedom. The problems considered involved only open chains.

4 Overview of completed work

The problem statement is, 'Explore the problem of folding origami with a robot.' How does the work I have done lead towards this goal? In any scientific exploration, it is important to show that the problem is feasible, to determine important characteristics of the problem, to derive models to capture these characteristics, and to analyse the models to learn new things about the problem. The work I have completed makes a strong beginning in each of these directions.

- **Feasibility.** I have established that the problem is feasible – simple origami can be folded with a robot. I have also shown that it is possible to plan a path between two configurations for simple origami designs.
- **Important characteristics of the problem.** The experimental work suggests that although origami paper is flexible, it may be reasonable to model origami designs as articulated rigid bodies. When seen as articulated rigid bodies, most origami designs involve closed chains. Planning for these closed chains will ultimately require some knowledge of the topology of the configuration space.
- **Modelling.** I have derived a model of flexible paper, and a model of origami as an articulated rigid body. Each model captures some importants aspects of the problem of folding origami.
- **Analysis.** I have analyzed the topology of the configuration space of the simplest origami design that includes closed chains. The articulated rigid body model also suggests a formula to determine the mobility of an origami design, and one measure of the complexity of an origami design. I have explored some ways of moving around in the configuration space of an origami design.

The next four sections discuss the completed work in detail. The sections are: experimental work, flexible one-dimensional paper, rigid-body origami, and simulation and planning. The structure of each section is as follows. First, the goals of the completed work are presented, then the work is described. The work is then evaluated. The evaluation consists of two components. The first is an evaluation of how well the goals were achieved. The second summarizes the important results and issues that were raised, and attempts to answer the question, "What have we learned?". Finally, each section discusses the questions that the thesis will address, and indicates the experiments and methodology that will be used to answer these questions.

5 Experimental work

I conducted a series of experiments with a four-joint Adept arm to gain a better understanding of the problem of robotic origami folding. I designed a simple tool, and

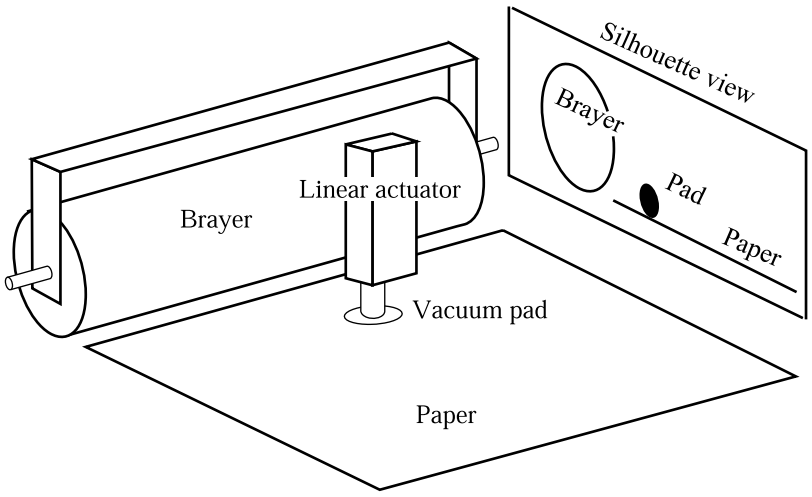


Figure 5: Tool design for making simple folds.

programmed the Adept using hand-selected 'via' points. A primary goal was to demonstrate the feasibility of folding origami with an Adept arm. This goal was achieved. In the last few experiments, the robot did crease and fold a square piece of origami into a simple envelope.

Another important goal of the work was to determine a set of 'manipulation primitives' that could be combined to fold simple origami. I have designed some primitive actions based on an analysis of a few origami designs of medium complexity (a samurai hat, a paper airplane, and partial foldings of a crane and a balloon.) The current mechanical design can execute some of the primitives (placing the paper, simple folds), but not others (separating flaps and tucking, for example.)

5.1 Tool design

The design of the tool I used is shown in figure 5. A vacuum pad (suction cup) was used to pick up the paper; a pneumatic vacuum generator supplied the vaccum. *Ink brayers* are usually used to roll ink onto rubber stamps. They are available in sizes from about 8 cm in length to about 15 cm in length, and are made from a variety of materials including foam, soft rubber, hard rubber, and acryllic. A 15 cm long soft rubber ink brayer was attached to the tool to allow the robot to 'roll out' creases in the paper. A pneumatically powered linear actuator moved the vacuum pad out of the way while the brayer was being used. I also built a wooden table to act as a workspace; for many of the experiments, the table played the role of a second hand.

5.2 Folding the paper

The first experiments I conducted were motivated by the observation that humans often place precise creases by aligning two edges or corners, and then 'rolling out' the cur-

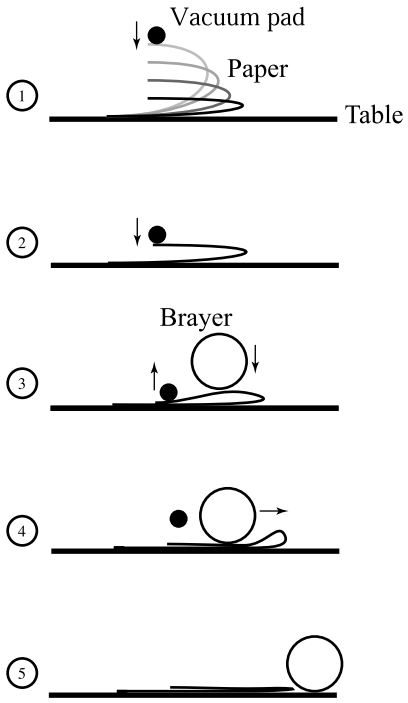


Figure 6: Placing a simple crease.

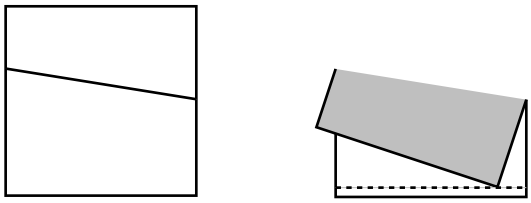


Figure 7: Creasing as a reflection.

vature until the paper is creased. The alignment of the edges or corners determines the location of the crease – a person who has difficulty drawing a straight line free-hand can still fold a straight crease.

Figure 6 shows the experiment I designed to explore this technique. The operation requires two hands. As a substitute for the second hand, I taped one edge of the paper to the table. The robot grasped the other end of the paper with the vacuum pad, and pressed it down into contact with the table (1, 2). Once the brayer was firmly holding the paper against the table, the vacuum pad was removed (3). The robot then rolled the brayer forwards (4), rolling out the curvature and eventually creasing the paper (5).

One interesting observation is that not all paper configurations can be 'rolled out' in this way. When the taped edge and the grasped edge were precisely aligned, this simple operation was fairly repeatable, and the accuracy of the crease placement appeared to be within a few millimeters. Occasionally, the paper tore along endpoints of the crease, typically when I had not place the paper carefully before the gripper grasped the top edge. When I intentionally rotated the gripper ten degrees before placing the top edge, the paper crumpled badly as the brayer rolled across it.

If there is only a single crease in a piece of paper, it acts as a line of reflection. If there is no line of reflection consistent with the placement of the two edges, then we should not expect the formation of a single crease. Figure 7 shows an example. If the upper right corner of the paper is dragged in either direction along the dotted line without rotation, there will be no single crease consistent with the relative position of the upper and lower facets of the paper. Although it seems likely that most relative placements of the edges can be achieved with three creases, the 'rolling out' action does not appear to permit the formation of more than one crease.

5.3 Bending the paper

The Adept arm I used has only four degrees of freedom; we can describe the configuration of the tool by the coordinates x, y, z, and θ, the rotation about the z axis. This means that there is no way to flip a rigid body while grasping it from the top. If the paper is grasped, the grasped point cannot be flipped over to create a 180 degree bend. In my second experiment, I investigated the possibility of using the flexiblity of the paper to create a bend. Figure 8 shows the motion of the gripper and paper during the experiment.

The robot first gripped the paper, and dragged it off of the table. It allowed the paper to droop, and then swept the paper back onto the table. During this phase, the

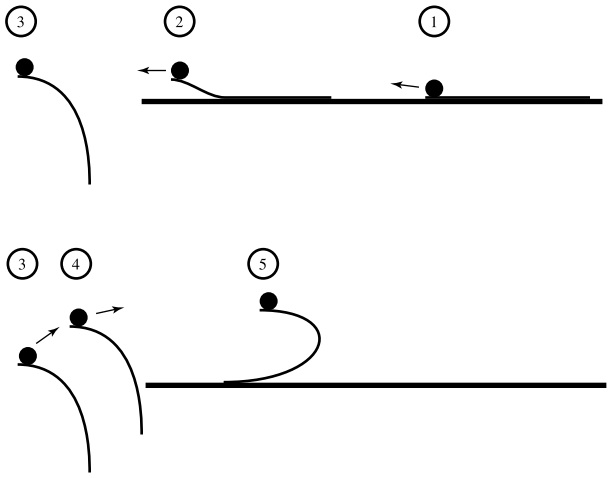


Figure 8: Bending the paper.

edge of the table caused the paper to bend.

I was surprised by the fact that the paper *did not* always droop once it was dragged off the edge of the table. This may have been because the paper crinkled slightly when gripped by the vacuum pad. When support was removed from the paper it remained in an essentially horizontal configuration. This is an interesting example of paper behaving like a rigid body.

When people hold a newspaper up to read it, it is important that the top edge not droop down. However, for this experiment, some droop was necessary. I implemented a strategy to ensure that the rigidity was broken. After moving the paper off of the edge of the table, the robot moved the (ungrasped) edge of the paper underneath the table edge. The arm then lifted the paper, using the table edge to break the rigidity. The arm then swept the paper across the table edge and back onto the table to place the desired bend.

The flexibility of the paper also allows it to be flipped with one hand. If the vaccuum pad releases the paper once the bend has been placed, the paper slides off, and usually ends up upside down. Unfortunately, the springiness of the paper makes this motion dynamic and very unpredictable!

Although the dynamic aspect could be ameliorated by grasping the bottom edge with a simple gripper before releasing the top edge, a larger problem is that the motion of the bottom edge and shape of the bend are difficult to predict. Typically, some twist develops between the top and bottom edges. Also, if the robot drags the paper from left to right starting in the final position shown in figure 8, the bottom edge will eventually move, trailing behind the robot. However, if the robot moves the gripper from right to left, the bottom edge does not move until it eventually flips over. This would probably make it hard to control the position of the bottom edge even with a feedback control law.

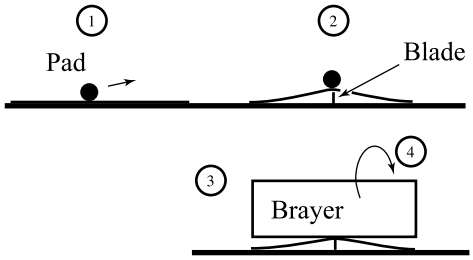


Figure 9: Placing a shallow crease using a blade.

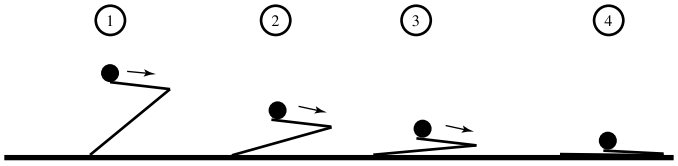


Figure 10: Folding pre-creased paper.

5.4 Placing shallow creases

Although bending and flipping the paper using the edge of the table turned out to be unpredictable, the method is more successful if there is already a crease in the paper. If there is already a crease, the deformation of the paper tends to occur along that crease during the bending phase. I explored pre-creasing the paper in the experiment outlined by figure 9.

I attached the blade of a long paint scraper to the table. The robot placed the paper above the blade. The brayer then rolled across the paper along the blade, forming a crease along the line of the blade. The process is analogous to bending sheet metal in a brake – the blade acted as the punch and the soft rubber brayer acted as the die.

The accuracy and repeatability of this procedure were not very good. As the brayer moved across the blade, the paper tended to be dragged along a few millimeters by the brayer. Increasing the accuracy is a problem for a future mechanical design.

5.5 Folding creased paper

The method just described places a very shallow crease in the paper. However, even a shallow crease may be useful, since creased paper behaves differently from uncreased paper. Once there is a crease, applying forces to the edges of the paper tends to cause more bending along the crease than elsewhere in the paper. I used this observation to fold the paper once a shallow crease had been placed. Figure 10 outlines the experiment.

The motions of the paper for this experiment were the same as the motions shown in figures 6 and 8. The paper was grasped from above, near the crease. The paper was then oriented so that the crease was parallel to the edge of the table. The robot

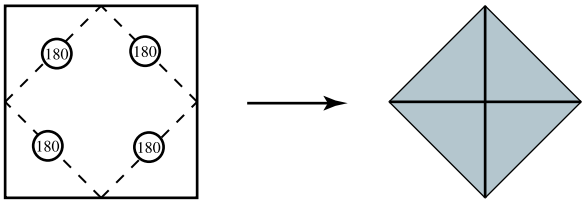


Figure 11: An ‘envelope’ origami shape. The four corner flaps are folded into the center of the square.

dragged the paper off the table, and used the edge of the table to break any rigidity and bend the paper. The robot then dragged the paper along the table while squeezing downwards; most of the bending of the paper occurred along the crease. Once the paper was essentially flat on the table, the robot released the vacuum grip and rolled the brayer across the paper to sharpen the crease. It is interesting that in addition to allowing the fold to be made precisely, pre-creasing also allows the paper to be flipped: part of the paper is upside down after the motion, and positioned in a known location.

5.6 Folding an envelope

The experiments discussed to this point developed a series of primitives. These primitives can be combined to fold the simple origami shown in figure 11. Figure 12 shows the strategy. First, one crease was placed in the paper using the method described in section 5.4. The robot then dragged the paper off the table, and used the table edge to ‘fold under’ the flap formed by the crease. The fold was made along the crease by squeezing the paper against the table while dragging the paper to the right. The brayer rolled over the paper to sharpen the crease. At the end of the procedure, one corner of the paper was folded under. The process was repeated for each of the three remaining flaps, folding the paper into the desired envelope shape.

5.7 Evaluation

“How well did it work?”. The robot was able to repeatably place all four creases, and fold the flaps to form the envelope. The error in the pose of the first crease was small – on the order of a few milimeters and a few degrees. However, error accumulated, and the pose error of the last crease was typically nearly a centimeter, and about ten degrees. The primary source of error seemed to be the slip of the paper on the table as the brayer rolled over it to place a crease. Reducing this error will require that the paper be grasped during creasing, probably on both sides of the crease. Some simple sensing of paper edges could be used to reduce the problem of error accumulation.

5.7.1 Manipulation primitives

“Can the method be extended to other origami designs?” Currently, the accumulation of error means that four creases is probably about the maximum possible. If the accuracy

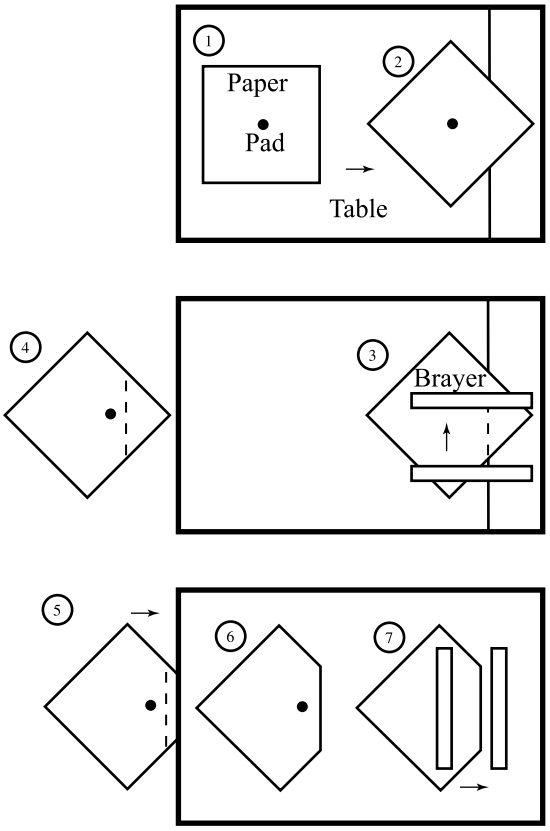


Figure 12: Folding the envelope.

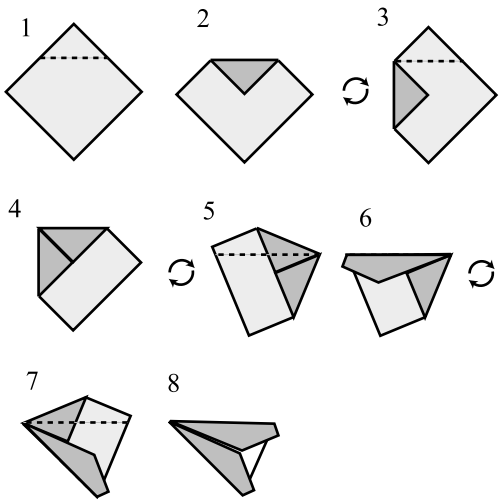


Figure 13: Partial folding of a paper airplane, using just two manipulation primitives, *positioning* and *folding down*.

could be increased, then there is a class of origami that the primitives available could be used to fold. We will say that a manipulation task is *feasible* if the available primitives would be sufficient to complete the task, if all primitives were executed with perfect accuracy. The *feasibility class* is the set of tasks that can be completed given some set of primitives.

Consider the partial folding of a paper airplane shown in figure 13. The manipulation primitives used are *positioning* of the paper, and *folding down*. *Positioning* of the paper requires that the robot be able to move a flat piece of a paper rigidly to a new configuration, and is indicated by a pair of arrows arranged in a circle. *Folding down* is indicated by a horizontal dashed line. A successful execution of the *folding down* primitive involves folding the section of paper above the line down (towards the reader) across the crease line, creating what is often called a 'valley fold'. Both of these primitives can be executed by the current system to some degree.

The envelope and the paper airplane should probably not be placed in the same feasibility class. Each of the creases in the envelope is only one layer thick; in the airplane, multiple layers of paper are simultaneously folded. The current implementation of the *folding down* primitive first creases the paper and then uses the edge of the table to execute the fold; this is much more likely to fail if there are multiple layers. In order to fold the airplane, the implementation of the primitives must be able to successfully deal with the case where there are multiple layers.

What other primitives should be available? Consider the folding of the samurai hat shown in figure 14. Two additional primitives are introduced. The first is *flipping*, denoted by a looping arrow; the meaning of this primitive is obvious. The combination of *flipping*, *folding down*, and *positioning* primitives allows both 'valley folds' and 'mountain folds' to be formed. The second primitive introduced is *separating*, shown

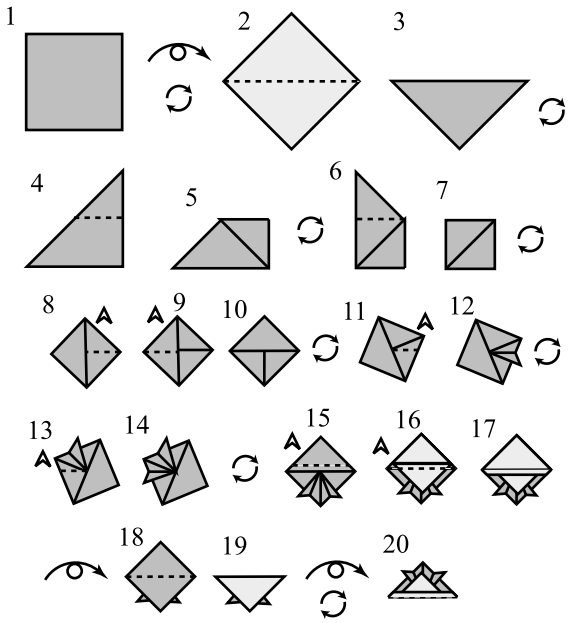


Figure 14: Folding a samurai hat with four primitives.

by the hollow arrow first introduced in step eight. Sometimes it is necessary to fold down only one of two flaps. Whereas the primitives introduced so far may be thought of as verbs in a manipulation sentence, the *separating* primitive should be considered to be a modifier or adverb. *Separating* is always performed in combination with another primitive, and implies that some number of flaps should be separated before applying the action primitive. Although separation of flaps was not necessary until step eight, it is interesting that all but one of the remaining folds require it.

The partial crane folding shown in figure 15 introduces another simple primitive, *unfolding*, which is denoted by an upwards-pointing arrow. When combined with *folding down*, *unfolding* permits pre-creasing the paper.

As was discussed briefly in the introduction, a rigid body model of origami with crossing creases contains closed chains. If we unfold the envelope or the airplane, we find that no creases cross. This is not the case with the samurai hat, but the sequential nature of the folds means that the crossing creases never become an issue.

The crane folding shown in figure 15 does require manipulation of closed chains. Step 6 requires a complicated simultaneous manipulation of multiple facets. The arrows denote folding the designated creases to a single line. If each facet is considered to be a rigid link, and each crease a hinge joint, then the eight-link mechanism is a closed chain with a single loop. We might call the primitive required to fold this origami *closed chain manipulation*. (For the simple case where there are two colinear creases, Miyazaki *et al.* [39] refers to this operation as *tucking*.)

What about more complicated primitives? Figure 16 shows a partial folding of a balloon. There are only two crease lines; all four creases are valley creases. Huffman's

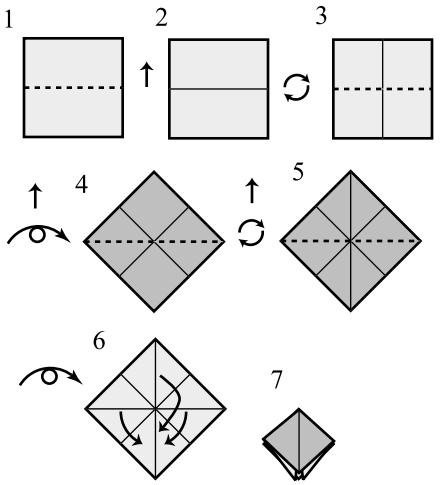


Figure 15: Partial folding of the crane.

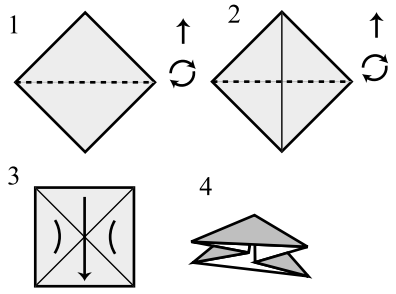


Figure 16: Partial folding of an origami balloon.

results tell us that there cannot be four valley creases and no mountain creases if the facets are planar. Step 3 requires bending the two side facets. So we might define an additional primitive, *flex*, that could be used to modify other primitives.

Once the shape shown in step 4 has been squeezed flat, two additional creases are introduced. So it would be possible to avoid bending the side facets by precreasing the paper. (In fact, the partial balloon folding and the partial crane folding are then the same, up to a rotation of the paper before creasing.) An interesting question is whether pre-creasing can always be used to avoid bending. If so, is it ever necessary to add additional creases that will not be found in the final origami?

This collection of primitives is not complete, particularly since the primitives *closed chain manipulation* and *flex* are too general to be implemented. In fact, the software, *Origami, the Secret Life of Paper*, states that each of the traditional origami designs entails a *good move* – that is, a move that is 'both surprising and satisfying'. It is also hard to see how simple primitives can be used to describe the folding of complicated three-dimensional designs. However, the basic primitives discussed (*positioning, flipping, folding down, unfolding, separating*) seem sufficient to fold a rich class of origami. This class will be the focus of the experimental portion of the thesis.

The primitives allow a comparison not only between types of origami, but between origami and other folding manipulation. The following table lists some skills that are required to fold *simple* origami (for example, the samurai hat), *flat* origami (the crane), *modular* origami (many simple components), *origami sculpture* (state-of-the-art, often with a human or animal subject), sheet metal, and cartons.

	Simple	Flat	Modular	Sculpture	Sheet metal	Carton
2D position	✓	✓	✓	✓	✓	✓
Flip	✓	✓	✓	✓	✓	✓
3D position		✓	✓	✓	✓	✓
Simple fold	✓	✓	✓	✓	✓	✓
Fold to angle			✓	✓	✓	✓
Flex		✓		✓		
Curved creases				✓		
Separate	✓	✓		✓		
Grasp section			✓	✓		✓
Pre-creasing		✓		✓	✓	✓
Closed chain		✓		✓		
Flap insertion		✓	✓	✓		✓
Large forces					✓	

The skills for each class should be interpreted loosely as the minimum requirements to fold some representative examples of the class. The definition of each skill also depends on the material. For example, 'pre-creasing' of sheet metal refers to the practice of cutting out a wedge of material before making a bend; this greatly reduces the amount of force that needs to be applied, and reduces warping of the metal.

5.8 Proposed thesis work

"What have we learned?" The experimental work and the above discussion suggest a number of important characteristics of the problem.

- *A large class of origami can be folded with just a few manipulation primitives.* Although the samurai hat requires a large number of folds, and although the creases cross in the unfolded hat, no complicated primitives are necessary.
- *Certain origami designs require complicated manipulation to fold.* Examples include the crane and the balloon.
- *Two hands may be needed.* Paper is flexible, and grasping it at just a single point can make manipulation difficult. How many hands are needed? The experiments showed that if the paper is pre-creased, one grasping hand may be enough. (The table was used as a second, non-grasping hand.) If the design requires complicated manipulation of closed chains, more hands may be necessary. Are there origami designs that cannot be folded by a human being?
- *Creased paper and uncreased paper behave differently.* The experiments showed that it may be relatively easy to fold with one hand once a crease has been placed.
- *Error accumulates each time the paper is released.* This problem was compounded by the fact that the paper was not grasped in any way while the creases were placed.

The thesis will explore and address each of these issues. The primary goal of experimental work is to design a system that implements a few manipulation primitives, and is somewhat more accurate than the current system. I will focus on three primitives: *positioning*, *flipping*, and *folding down*. I will also partially address the problem of implementing the primitives *unfolding* and *separating*.

5.8.1 Design of a new folding mechanism

Figure 17 shows a possible design for the folding mechanism. A vacuum pad would be used to grip the paper, and the Adept arm would *position* the paper over the two 'palms' shown in the figure. The right palm is in fact just the table. The left palm is actuated, and uses vacuum to grip the paper.

Folding down would be accomplished by the procedure shown in the figure. Once the paper was positioned, the vacuum in the left palm would grip the paper. The vacuum pad would be removed by the Adept, and be replaced by a sharp wedge. The left palm would then rotate about the center joint, creating an initial fold. The wedge would be removed, and folding would be completed. *Flipping* could be accomplished by a similar procedure.

The advantages of the new design over the old include the fact that the paper would always be gripped by either the left palm or the vacuum pad. This should significantly reduce the accumulation of error. Folding and creasing also would occur simultaneously, so the paper could be treated as a single rigid body. If the right palm were

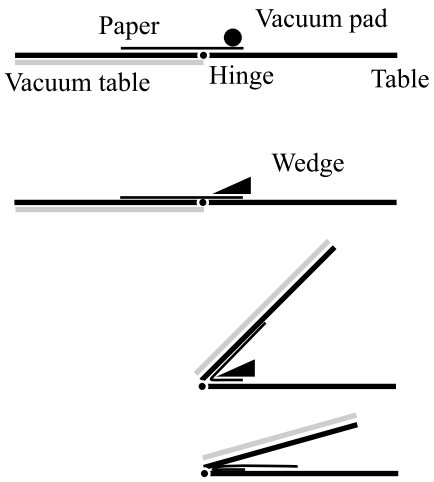


Figure 17: Possible folding mechanism design

designed to tilt somewhat to the left, it would be possible to remove error by releasing the vacuum grip while the paper was being flipped. Gravity would then cause the paper to slide to the bottom of the 'V', removing rotational and positional error along one axis. Tilting both palms forwards would allow error to be removed along the other axis. In either case, some vibration of the palms might be necessary to break friction.

5.8.2 Grasping origami

I will also analyze the problem of grasping origami. Although this component of the thesis will be primarily theoretical, some experimental verification of results will be necessary as well. I will consider the problem of grasping the partial balloon folding, and the partial paper crane folding. The proposed theoretical work will be described in more detail in a later section.

6 Modelling one-dimensional paper

Creased and uncreased paper behave differently. The work discussed in this section explores the behavior of paper that is bent but not creased. A primary goal of the work was to extend principles used in the manipulation of rigid bodies to a model of flexible paper.

The experimental work suggests that a one-dimensional model of paper that considers the paper 'viewed from the side' may be a good simplification. This model is applicable when the rulings of the paper are approximately parallel in R^3. Although this was not an explicit goal of the experimental design, a number of factors led to this being the case during most of the experiments. In the experiments described by figures 6 and 8, all motion of the paper was made in a direction perpendicular to the desired direction of the rulings. The robot also grasped the paper at the center of the

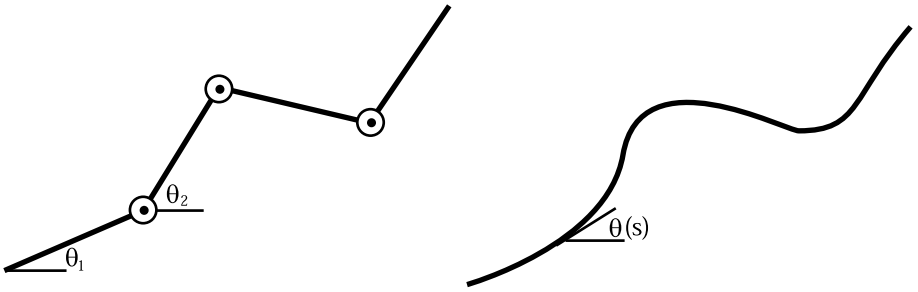


Figure 18: One-dimensional paper as the limit of a planar revolute arm.

desired rulings, so that bending would be more symmetric. In the experiments described by figures 9, 10,and 12, the creases were placed parallel to the desired direction of rulings.

In order to better understand the behavior of bending paper, I implemented a method described by Hirai *et al.* [22] to find likely configurations of paper subject to a set of geometric constraints. The model used by [22] describes the configuration of the paper using a set of Fourier basis coefficients. I used the model to extend some analysis techniques typically used for rigid bodies to the flexible paper. I determined a formula describing the forwards kinematics of the paper. The principle of virtual work can also be applied to the model. I derived the dynamic equations that are implied by the model, and considered the problem of force control of the paper.

Although I achieved some interesting results with this model, I do not anticipate that this work will be a primary focus of future thesis work, for reasons discussed below (section 6.6). The derivations in the remainder of the section are provided for completeness, but may be skipped by the impatient reader.

6.1 Formulation of the model

One way to model paper is to describe the configuration by a function that gives the angle of the tangent at each point on the curve

$$\theta(s) : [0, 1] \mapsto [0, 2\pi) \tag{5}$$

and an initial condition (x_0, y_0) which describes the location of a particle on an edge of the paper. Figure 18 shows an example. The location of each particle is then given by

$$x(x_0, \theta(\cdot), s) = x_0 + \int_0^s \cos\theta(\tau)\, d\tau \tag{6}$$

$$y(y_0, \theta(\cdot), s) = y_0 + \int_0^s \sin\theta(\tau)\, d\tau \tag{7}$$

The tangent vector to the parameterized curve described by equations 6 and 7 is always of unit length; the paper was assumed not to stretch. Since θ is a continous

function of one variable, it can be approximated by a finite Fourier series. The Fourier coefficients then give a finite-dimensional representation for the configuration of the paper. Hirai used this method to find equilibrium points of model of copy paper held at two opposite edges.

If the Fourier coefficients are collected in a vector p then the state of the paper q is given by

$$q = \begin{pmatrix} x_0 \\ y_0 \\ p \end{pmatrix} \tag{8}$$

If we define the vector $b(s)$ containing two leading zeros followed by the Fourier basis functions,

$$b(s) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ \sin 2\pi s \\ \cos 2\pi s \\ \sin 4\pi s \\ \cos 4\pi s \\ \vdots \end{pmatrix} \tag{9}$$

then we may write θ, x, and y as functions of q and s:

$$\theta(q,s) = q^T b(s) \tag{10}$$

$$x(q,s) = x_0 + \int_0^s \cos q^T b(\tau)\, d\tau \tag{11}$$

$$y(q,s) = y_0 + \int_0^s \sin q^T b(\tau)\, d\tau \tag{12}$$

We may describe trajectories of the paper by allowing q to be a function of time.

Hirai *et al.* [22] used this formulation together with non-linear programming software to find configurations of the paper that minimize potential energy due to some constraints. I implemented this algorithm. One difficulty with the algorithm is that there may be multiple local minima of the potential energy function. In order to better understand how the paper actually behaves, I used the Fourier basis representation to find the Lagrangian dynamics of the system. I will also briefly discuss the problem of controlling some control points or a subset of the degrees of freedom of the paper by applying forces at specified points along the paper. Although the dynamics of this model were also considered by [59], their algorithm involved numerically minimizing the difference between kinetic and potential energy at each time step; my derivation is analytical.

6.2 Potential energy functions

Hirai used the following formula to calculate the potential energy due to gravity

$$V_g(q) = \rho \int_0^1 y\, ds = \rho y_0 + \rho \int_0^1 \int_0^s \sin q^T b(\tau)\, d\tau\, ds \tag{13}$$

where ρ is the mass of the paper per unit length, assumed constant over the paper for simplicity. The multiple integral can actually be simplified to a single integral:

$$V_g(q) = \rho y_0 + \rho \int_0^1 (1-\tau) \sin q^T b(\tau)\, d\tau \tag{14}$$

Hirai used a constitutive law to model the spring energy of the paper

$$V_s = k \int_0^1 \theta'^2\, ds, \tag{15}$$

where k is a spring constant determined experimentally. Hirai calculated the energy numerically for a given configuration using the Fourier basis representation:

$$V_s(q) \quad = \quad k \int_0^1 (q^T b')^2\, ds \tag{16}$$

Actually, the integral can be found analytically. If we define a diagonal matrix K with k_i as the ith diagonal element of K

$$k_i = \begin{cases} 0 & \text{if } i = 1 \\ k(i-2)^2\pi^3 & \text{if } i \text{ is even, } i \geq 2 \\ k(i-3)^2\pi^3 & \text{if } i \text{ is odd, } i \geq 3 \end{cases} \tag{17}$$

$$\tag{18}$$

it turns out that the spring energy can be written as a simple weighted sum of squares:

$$V_s(q) = q^T K q \tag{19}$$

6.3 Differential kinematics

In a typical manipulation task, we might want to be able to determine the motion of a number of points of interest on the paper, due to a change in the configuration of the paper as a whole. For example, the points of interest might be the set of points of contact between the paper and a manipulator. We can describe the points of interest by a vector of parameters:

$$\mathbf{s} = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \end{pmatrix} \tag{20}$$

The location of the points is given by the vector

$$\mathbf{x}(q, \mathbf{s}) = \begin{pmatrix} x(q, s_1) \\ x(q, s_2) \\ \vdots \\ y(q, s_1) \\ y(q, s_2) \\ \vdots \end{pmatrix} \tag{21}$$

We first consider how the x coordinate of a single point changes as a single basis function coefficient is varied.

$$\frac{\partial x(q,s)}{\partial q_1} = \frac{\partial x(q,s)}{\partial x_0} = 1 \tag{22}$$

$$\frac{\partial x(q,s)}{\partial q_2} = \frac{\partial x(q,s)}{\partial y_0} = 0 \tag{23}$$

For $i \geq 3$,

$$\frac{\partial x(q,s)}{\partial q_i} = \frac{\partial}{\partial q_i} \int_0^s \cos\left(q^T b(\tau)\right) d\tau \tag{24}$$

Since everything is continuous, we can exchange differentiation and integration:

$$\frac{\partial x(q,s)}{\partial q_i} = \int_0^s \frac{\partial}{\partial q_i} \cos\left(q^T b(\tau)\right) d\tau = -\int_0^s b_i(\tau) \sin\left(q^T b(\tau)\right) d\tau \tag{25}$$

We may calculate the partials of $y(s)$ similarly. For $i \geq 3$,

$$\frac{\partial y(q,s)}{\partial q_i} = \int_0^s \frac{\partial}{\partial q_i} \sin\left(q^T b(\tau)\right) d\tau = \int_0^s b_i(\tau) \cos\left(q^T b(\tau)\right) d\tau \tag{26}$$

Define the vector functions $x_q(q,s)$ and $y_q(q,s)$ which contain the partial derivatives of x and y with respect to q. Then define the $2m \times n$ *Jacobian* matrix:

$$J(q, \mathbf{s}) = \begin{bmatrix} x_q(q, s_1)^T \\ x_q(q, s_2)^T \\ \vdots \\ y_q(q, s_1)^T \\ y_q(q, s_2)^T \\ \vdots \end{bmatrix} \tag{27}$$

Then

$$\dot{\mathbf{x}}(q, \mathbf{s}) = J(q, \mathbf{s})\dot{q} \tag{28}$$

6.4 Kinetic energy and dynamics

In order to find the kinetic energy of the system, we first consider the velocity of each particle. By the multivariable chain rule:

$$\dot{x} = x_q^T \dot{q} \tag{29}$$
$$\dot{y} = y_q^T \dot{q} \tag{30}$$

The square of the velocity of the particle at position s is

$$v^2(q, \dot{q}, s) = \dot{x}^2 + \dot{y}^2 \tag{31}$$
$$= \dot{q}^T (x_q x_q^T + y_q y_q^T) \dot{q} \tag{32}$$

We define ρ to be the mass of the paper per unit length; we assume ρ is constant over the paper. Then the kinetic energy is

$$T = \frac{\rho}{2} \int_0^1 v^2(q, \dot{q}, s)\, ds \tag{33}$$
$$T = \frac{\rho}{2} \int_0^1 \dot{q}^T (x_q x_q^T + y_q y_q^T) \dot{q}\, ds \tag{34}$$
$$T = \frac{1}{2} \dot{q}^T \left(\rho \int_0^1 x_q x_q^T + y_q y_q^T\, ds \right) \dot{q} \tag{35}$$

We define the mass matrix

$$M(q) = \rho \int_0^1 y_q y_q^T + x_q x_q^T\, ds \tag{36}$$

The kinetic energy is

$$T(q, \dot{q}) = \frac{1}{2} \dot{q}^T M(q) \dot{q} \tag{37}$$

Once the kinetic energy and mass matrix have been defined, we can derive the dynamical equations. The details of the derivation are ommited, but it turns out that the equations can be written in a familiar form:

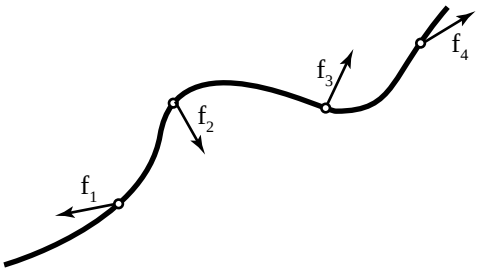
$$M\ddot{q} + \dot{M}\dot{q} = T_q - V_q + J^T f \tag{38}$$

One use of the dynamic equations is simulation. We could truncate the Fourier series, invert M to solve for $\ddot{q}$, and numerically integrate $\ddot{q}$ to find the trajectory $q(t)$. Since the mass matrix is symmetric, we could increase the numerical stability of the simulation by using a Cholesky decomposition at each time step, rather than inverting M directly.

The dynamic equations also contain information about the static case. If $\dot{q} = \ddot{q} = 0$, then

$$V_q = J^T f \tag{39}$$

6.5 Force control



What if we can control forces at some specified points? Let J_1 be the Jacobian that relates the motion of the points where forces are applied to the generalized velocity $\dot{q}$. Then write the dynamic equations.

$$\ddot{q} \quad = \quad M^{-1}(T_q - V_q - \dot{M}\dot{q}) + M^{-1}J_1^T f \tag{40}$$

Let $\mathbf{s}$ be the vector of parameters describing the points we want to control. Then $\mathbf{x}(\mathbf{s})$ will be the Cartesian location of these points. Let J_2 be the Jacobian relating $\dot{q}$ and $\mathbf{x}$.

$$\dot{\mathbf{x}}(q, \mathbf{s}) = J_2(q, \mathbf{s})\dot{q} \tag{41}$$

Differentiate with respect to time.

$$\ddot{\mathbf{x}} \quad = \quad J_2\ddot{q} + \dot{J}_2\dot{q} \tag{42}$$

$$J_2\ddot{q} \quad = \quad \ddot{\mathbf{x}} - \dot{J}_2\dot{q} \tag{43}$$

Premultiply equation 40 by J_2:

$$J_2\ddot{q} = J_2M^{-1}(T_q - V_q + -\dot{M}\dot{q}) + J_2M^{-1}J_1^T f \tag{44}$$

Substitute:

$$\ddot{\mathbf{x}} = J_2M^{-1}(T_q - V_q - \dot{M}\dot{q}) + \dot{J}_2\dot{q} + J_2M^{-1}J_1^T f \tag{45}$$

which has the form

$$\ddot{\mathbf{x}} = d(q, \dot{q}) + C(q)f \tag{46}$$

C is square, and non-singular as long as both J_1 and J_2 have full row rank. (The number of Fourier coefficients chosen to represent the system should be large compared to the number of control points.) We can choose $\ddot{\mathbf{x}}$ as we like, and solve for f.

$$f = C^{-1}(\ddot{\mathbf{x}} - d) \tag{47}$$

6.6 Evaluation

"What have we learned?" One observation is that directly simulating the system using the dynamics equations derived requires the mass matrix to be computed and factored at each time step. Since the mass matrix is dense, this is computationally expensive. The inverse dynamics method of force control described suffers from the same problem. The result is that only a few Fourier coefficients can be used if numerical simulation or control is the goal. Much better results could probably be achieved by modelling the paper as a serial chain, and using efficient simulation techniques such as those described in [6].

On the other hand, the model derived may be useful from the perspective of analysis. It is particularly interesting that under the Fourier basis representation, the spring energy of the system is a simple weighted sum of squares. Another possible benefit is the smoothness of the model; derivatives are defined everywhere along the curve. (This would not be the case if the paper were modelled as a serial chain.)

The completed work suggests some interesting questions and approaches to controlling flexible paper. However, when compared to other aspects of folding origami, the problem of controlling flexible objects using models similar to that derived has been relatively well-addressed in the literature. Therefore, extending this work will not be a focus of the thesis.

7 Rigid origami

We return to two observations made in section 5:

- *A large class of origami can be folded with just a few manipulation primitives.* Although the samurai hat requires a large number of folds, and although the creases cross in the unfolded hat, it seems that no complicated primitives are necessary.
- *Certain origami designs require complicated manipulation to fold.* Examples include the crane and the balloon.

The experimental work focusses on the first of these issues: developing simple primitives that can be used to fold a variety of origami shapes. The goal of the work described in this section is an analysis of some more complicated cases. We focus on one specific problem: that of folding a pre-creased closed chain.

The partial crane folding shown in figure 15 requires pre-creasing, and then simultaneously folding along multiple crease lines to form the base. Typical instructions for the balloon base require that some of the facets be bent, while simultaneously folding along multiple crease lines. Figure 19 shows how two additional creases can be introduced to avoid bending of the facets. How bad can it get? Figure 20 shows the pre-creasing required to fold the stellated octohedron. There are eighty-eight creases. Fortunately, not all of the creases are folded simultaneously to create the final design.

If we fold an origami shape, and then unfold it, the paper will be creased. Since creases are typically made along straight lines in origami, the uncreased regions will be polygonal facets. When the origami is folded flat, the facets will be planar. We will

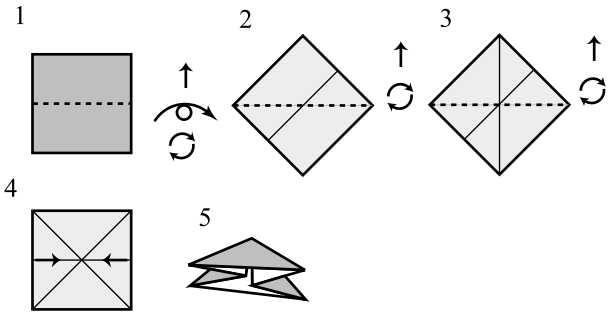


Figure 19: Folding the balloon base using pre-creasing.

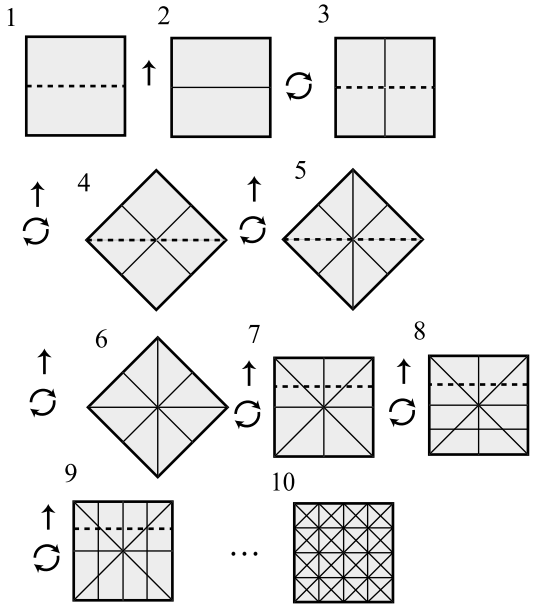


Figure 20: Complicated pre-creasing required to fold the stellated octohedron.

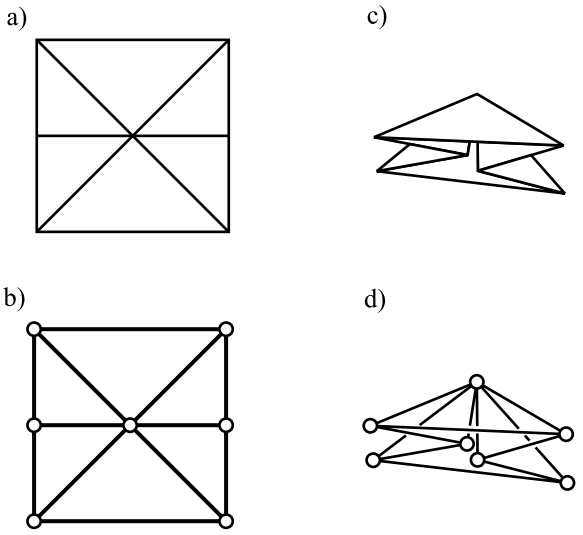


Figure 21: Rigid origami.

define *creases* as the edges of the facets. Creases may be colinear; we will call a line containing multiple creases a *crease line*.

The simplest way to model pre-creased origami is to consider each facet as a rigid link and each crease as a hinge joint; this model is similar to those used by Huffman [23] (geometry of creases), Lu and Akella [34] (carton folding), and Gupta *et al* [18] (sheet metal bending). Even for this simple model of origami, the kinematic structure may be quite complicated. If crease lines cross, then the structure includes closed chains.

Configuration spaces of closed chains are often difficult to parameterize, and for most closed chains no smooth one-to-one global parameterization exists. Since constraints may be dependent, it is often difficult to even determine the number of degrees of freedom. In this section, I will discuss some of the interesting features of the rigid origami model, including the number of degrees of freedom, and the topology of the configuration space of the simplest closed-chain rigid origami.

7.1 Origami with struts

In order to analyze rigid origami, it is often convenient to consider an equivalent mechanism made up of struts and ball joints. Consider the origami shown at the top of figure 21. There are six facets and six creases. The mechanism shown at the bottom of the figure has twelve struts and seven ball joints; the struts are placed around the border of each facet, and the ball joints are placed wherever struts meet. Since each facet is triangular, the struts ensure that each facet remains rigid. Folding is possible along any crease, but along no other line.

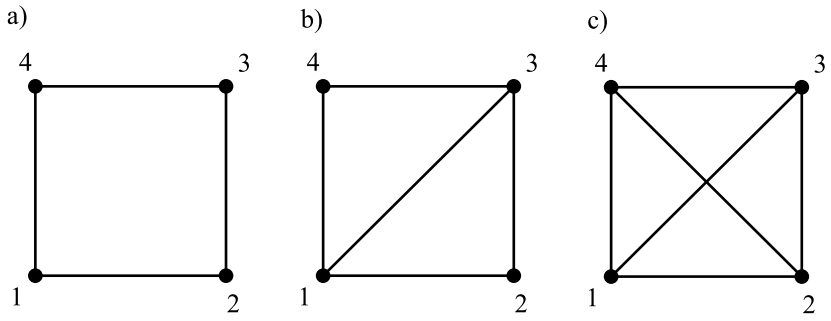


Figure 22: Rigidifying a four-bar linkage.

7.1.1 The minimum number of struts to make a facet rigid

If all facets are triangular, placing struts along the borders of facets and ball joints at the intersections of struts is sufficient. However, if facets have four or more edges, then this placement of struts might allow deformation of facets.

Consider the four-bar linkage shown in figure 22a. If we fix a base to the ground, the linkage still has one degree of freedom that allows flexing in the plane. Figure 22b shows a way to place a fifth strut to prevent flexing in the plane. It might seem that the mechanism shown in figure 22b will behave as a rigid body, but in fact it can flex out of the plane. Figure 22c places a sixth strut to prevent this out-of-plane flexing.

A body is *rigid* if the distance between any two points on the body is fixed. If there are two, three, or four vertices, then it is necessary to connect each pair of vertices with a strut to ensure rigidity. If there are more vertices, it may not be necessary to use ${}_n\mathrm{C}_2$ struts.

A set of rigid bodies is *rigidly connected* if there are sufficient constraints such that the entire set of bodies must move as a rigid body. If we have a set of n vertices in R^3, we need at least

$$n_s = 3n - 6 \tag{48}$$

struts to rigidly connect them, since each vertex has three degrees of freedom, each strut removes at most one degree of freedom, and the rigid body has six degrees of freedom.

Two rigid bodies that share three non-colinear points are rigidly connected, since the common points describe a frame. We can use this observation to minimally rigidly connect any set of vertices, as long as no three of the vertices are colinear. Figure 23a illustrates one procedure.

Procedure #1:

1. Choose an ordering for the vertices.
2. Use three struts to connect the first three vertices.
3. Use three struts to attach each new vertex to the first three.

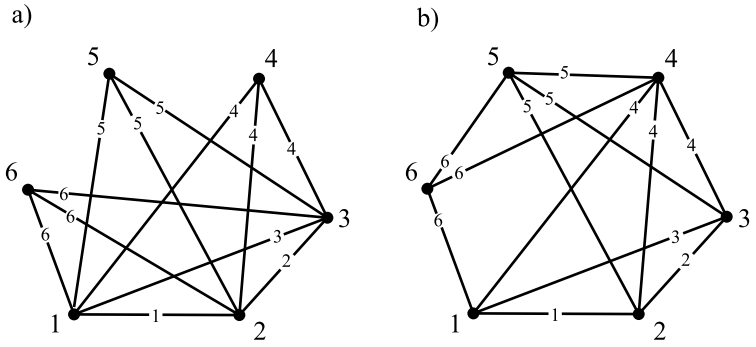


Figure 23: Two ways of rigidly connecting six points.

This procedure is simple, but has a disadvantage. All of the vertices of an origami facet lie in a plane; the facet is a polygon. In order to minimize the number of struts used to form the entire origami mechanism, the number of struts interior to each facet must be minimized. Figure 23b illustrates a procedure that rigidly connects the vertices, while ensuring that a strut is placed along each edge.

Procedure #2 (rigidly connecting a polygon):

1. Choose a polygonal ordering of the vertices from 1 to n.
2. Use three struts to connect the first three vertices.
3. Connect each of vertices $4 \ldots n-1$ to its three predecessors.
4. Connect vertex n to vertices $n-2$, $n-1$, and 1.

Since each new vertex is rigidly connected to a rigid body, each procedure rigidly connects the vertices. Since each procedure places a single strut for each of the first three vertices, and three struts for each additional vertex, $3n-6$ struts are placed, which is minimal.

7.1.2 Grübler, for origami

The origami shown in figure 21 has six facets and six creases. If we model the facets as rigid bodies and the creases as hinge joints, the mechanism is a closed chain. What is the mobility of the mechanism in a generic position, if we fix a base? Grübler's formula for the mobility of spatial closed chains is $M = 6f - 6$, where f is the number of freedoms for each joint. Each hinge joint has one freedom, so Grübler would tell us that the mobility is zero. However, we know that the mechanism has some degrees of freedom. The problem is that the constraints are not independent.

It turns out that the mobility is three. How do we arrive at this number? The construction using struts and ball joints gives a solution. For the strutted version, we expect the constraints (the lengths of the struts) to be independent unless two facets are co-planar. Therefore, we should be able to count the number of vertices and struts to

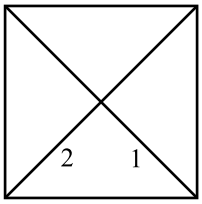


Figure 24: An origami design with crossing creases.

find the number of degrees of freedom. If all facets are triangular, $M \geq 3n - n_s - 6$. (Equality occurs whenever constraints are independent.) The mechanism in figure 21d has seven vertices and twelve struts, so $M \geq 21 - 12 - 6 = 3$.

If there are non-triangular vertices, we must rigidly connect the vertices of each facet. If p_i is the number of facets with i vertices, then the formula is:

$$M \geq 3n - n_s - 6 - \sum_{i=4}^{n} p_i(2i - 6) \tag{49}$$

7.2 Topology of the configuration space

When constraints are not dependent, equation 49 gives a way of counting the freedoms of rigid origami. What happens when constraints are dependent? Figure 24 shows an origami design. There are four facets and four creases, but only two crease lines. There are five vertices and eight struts, so equation 49 tells us that we should expect the mobility to be one. In fact, once a fold is made along the first crease, it is not possible to fold along the other crease until the paper has been folded flat.

The joint angles of at least three creases are necessary to determine the configuration of the paper. Two crease angles are enough to describe *trajectories* of the paper from any initial configuration. Figure 25 shows the topology of the configuration space, parameterized by the angles of the two creases labeled '1' and '2' in figure 24. Filled circles on the graph indicate nodes where transitions are possible, and unfilled circles indicate nodes where a transition between two edges would involve self-intersection. The numbers and origami diagrams show a possible trajectory to fold the paper into a triangle.

As can be seen from the figure, the configuration space of this simple origami is not a manifold, but rather a union of manifolds. Figure 25 was derived by considering each of the possible folded configurations of the paper. This will be difficult for more complicated origami shapes.

Milgram and Trinkle [37, 52] suggest a method for finding the topology of simple closed chains, if we ignore the possibility of self-intersections. The procedure is:

1. Break the loop by removing a single link.

2. Consider the $n - 1$ bar open chain. Fix the first link, and designate a point on the last link as an end-effector.

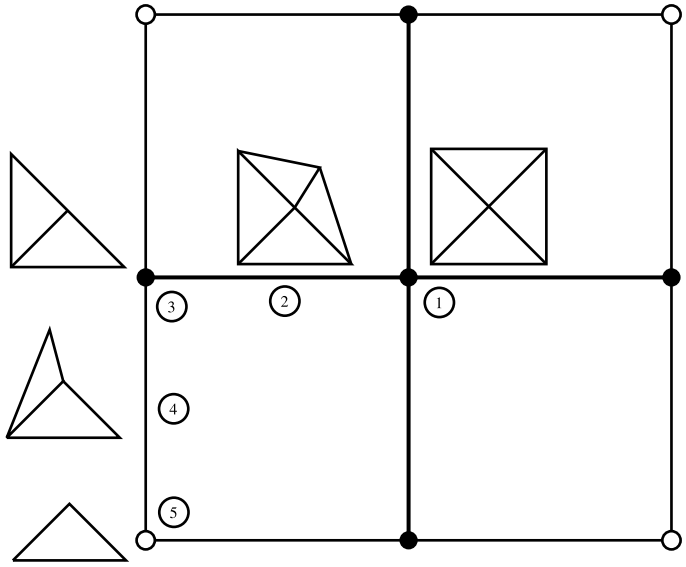


Figure 25: Topology of the configuration space of a simple origami design.

3. Determine the topology of the workspace of the end-effector. Where is the mapping from joint space to workspace one-to-one? Two-to-one? Many-to-one?

4. Intersect this workspace with the (simple) workspace of the two-bar linkage containing the fixed link and the removed link.

5. The configuration space is the image of these workspace points under the inverse kinematic mapping.

Milgram and Trinkle show how this procedure can be used to completely determine the topology of the configuration space for planar four- and five-bar closed loops. Although the procedure becomes very difficult for mechanisms with more links, the authors show that the analysis can be applied recursively to describe the topology of an n-bar linkage.

In a discussion and personal communication [36] with the author, James Milgram demonstrated how the procedure may be used to determine the topology of the origami shown in figure 24 in a methodical way. Figure 26 shows the results. Figure 26a shows the origami design. We will label the joints counter-clockwise from one to four, starting from the rightmost joint, and the facets by the index of the joint they follow. Facet one (in the first quadrant) is fixed. We remove facet four, and consider the workspace of a point on the third facet (26b). If we spin the third facet about joint three, the selected point describes a circle. If we spin this circle about joint two, we find that the workspace is a sphere. As the circle spins, the sphere is covered twice, except at the poles, which are covered an infinite number of times. So the mapping from joint angles to workspace is two-to-one, except at the poles, where it is many-to-one.

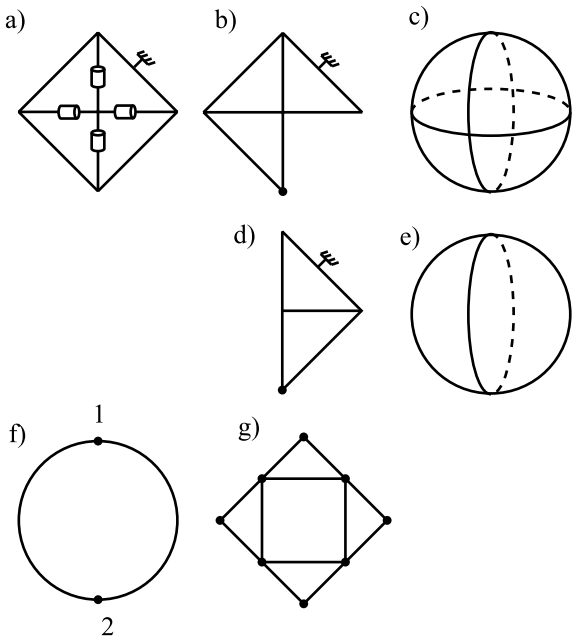


Figure 26: The configuration space of origami with two perpendicular crease lines, ignoring self-intersections. Due to Milgram [36].

We now consider the linkage containing only facets one and four (figure 26d). We consider the workspace of the point on facet four that is initially the same as the point we've already considered. The workspace of the point is a circle. We then intersect the two workspaces (26e, 26f). This gives the workspace of the point for the original closed chain mechanism. The configuration space of the mechanism is the image of this set under the inverse of the mapping from joint angles to workspace. In figure 26f, the numbers 1 and 2 show the points where the mapping was many-to-one (the poles). Along the arcs, the mapping was two-to-one. So the topology of the configuration space is four circles, connected circularly. This can be represented by the graph shown in 26g.

I have applied the method to some variations of the origami design with four intersecting creases. Figures 27 and 28 show the results. The workspace of the open chain shown in figure 27b is a segment sphere cut by a plane perpendicular to the line connecting the poles. The mapping from joint space to workspace is one-to-one at points 1 and 3 in figure 27f, many-to-one at point 2, and two-to-one everywhere else. The topology of the configuration space can therefore be represented by three connected circles, or the graph shown in figure 27g.

The analysis depicted in figure 28 is similar, but the mapping is one-to-one at point 3 in figure 28f. The topology of the c-space is a figure-eight. Physically, if we pick an angle for a single crease, there are two possible configurations for the rest of the creases ('elbow up' and 'elbow down'). To get from one configuration to the other,

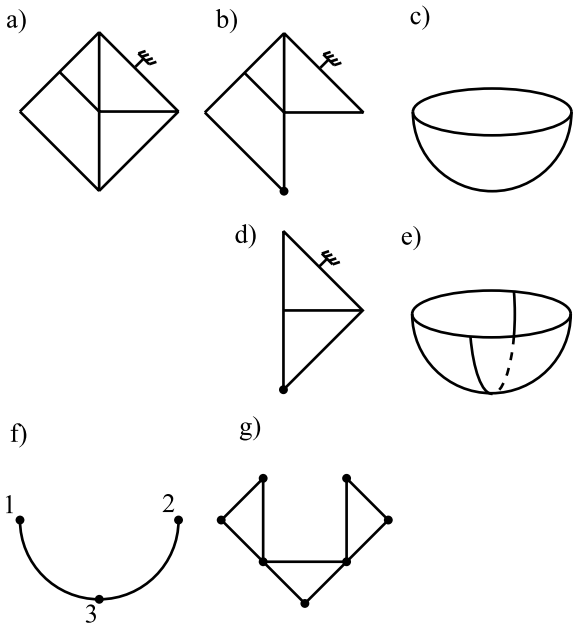


Figure 27: Topology of the configuration space for an origami mechanism with two colinear creases, and two non-colinear creases.

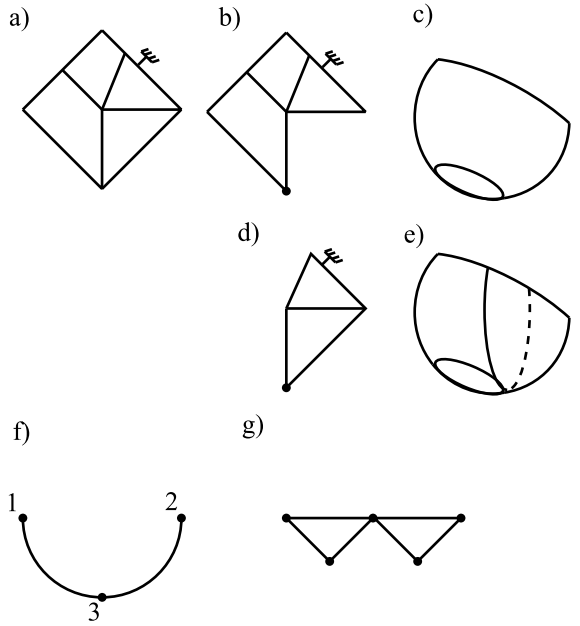


Figure 28: Topology of the configuration space for an origami mechanism with four non-colinear creases.

the mechanism must go through the flat configuration (the intersection point of the two circles in the figure-eight).

It is also interesting that the designs shown in figures 26, 27, and 28 are in the class studied by Huffman (described in section 3.1.2). One of Huffman's results for a mechanism with four non-colinear creases, like that shown in 28, is that at any time three of the creases must be convex, and the remaining concave, or vice versa. If the crease for which we determine the angle is concave, either the other three must be convex, or exactly one of the others must be convex. Since the configuration space only contains two components (the figure-eight), which crease is convex must fully determined by the origami design.

7.3 Evaluation

"What have we learned?" I proposed two different ways of representing the rigid origami model. The first uses struts and ball joints to create facets and creases, and the second uses rigid body facets and hinge joints at the creases. There are advantages to each of the representations.

If creases do not cross, the hinge joint mechanism is more useful, since it will contain no closed chains. There is therefore a one-to-one mapping between joint angles and mechanism configurations, and we have a familiar joint-space representation. It is not hard to analyze the topology of the simplest origami with crossing creases using this representation. Also, only a few numbers are needed to represent the configuration: the angles at each joint.

Even if creases do not cross, the ball joint mechanism will contain at least one (immobile) closed chain for each facet. Many variables are also needed to represent each configuration. However, the usefulness of the mechanism is that the constraints are local, and typically independent. This makes it much easier to count freedoms of the mechanism, particularly if there are multiple closed chains in the origami. The structure of the constraints is also very uniform: each constraint involves on the distance between two vertices. This means that the constraint Jacobian (discussed below), though large, is sparse and has a simple structure. (In fact, the constraint Jacobian is linear in configuration q.)

We have also learned that the topology of the configuration space for simple closed chains can be relatively complicated. A planner that does not take the topology into account is likely to be unable to plan paths between arbitrary points in the c-space.

The singularities of the open chain formed by removing links determine the c-space topology. This suggests that different nodes of the c-space are separated by configurations where facets (not necessarily adjacent) are co-planar. This conjecture needs to be generalized and proved, but might be a useful way of encoding the topology for use by a planner.

7.4 Proposed thesis work

The goal of this section was to explore a complicated manipulation primitive used in folding origami: simultaneous folding of creases in pre-creased paper. I have de-

termined the structure of the configuration space for the simplest case where creases cross.

The thesis work will explore the topology of the configuration space for more complicated designs involving more than four links. It will be difficult to directly apply Milgram and Trinkle's methodology to cases where there are more links or multiple loops, but it may be possible to prove some useful results. An interesting question that was raised by the completed work is whether the ball joint mechanism can also be used to find the topology of the configuration space. It seems likely that zeroes and sign changes of the eigenvalues of the constraint Jacobian correspond to regions of the c-space. (The constraint Jacobian, and some of its limitations, are discussed in the next section.)

8 Simulation and planning

The goal of the work described in this section is the same as that in the last: exploration of folding pre-creased origami. I considered the problem of planning local motions of the ball joint mechanism described in the previous section. I implemented a planner, and applied it to the simplest case, with good results.

Mechanical systems typically involve a set of constraints. Baraff [6] writes

> ...ultimately, we are faced with a basic choice. Either we model constraints by reducing the number of coordinates needed to describe the system's state, or we introduce additional forces into the system to maintain the constraints.

The constraints are *implicit* if generalized coordinates are used to parameterize the possible configurations, and *explicit* if the constraints are described by a number of auxiliary equations.

It is often difficult to find a joint space (or reduced coordinate) representation for origami and other complicated closed chains. One common solution is to break the chain. Motion is then planned for the first segment (now an open chain), and inverse kinematics are used to enforce compliance of the second segment. The primary difficulty lies in choosing where to break the chain, particularly if there are many closed loops. If the second segment is too short, compliance will not be possible. If it is too long, arbitrary choices must be made in the inverse kinematics solution.

Path planners for serial arms and mobile robots typically assume a reduced coordinate system. Explicit inequality constraints are used to express the fact that collisions should be avoided. However, I am not aware of any practical, general path planning algorithms that permit explicit equality constraints. This is surprising, given the success and ease of implentation of efficent dynamic simulation algorithms using explicit equality constraints (for example, see [6]).

In this section, I will discuss my preliminary work on planning for origami using explicit equality constraints. The formulation is similar to that used in non-linear programming and other optimization problems.

8.1 Formulation

The strutted origami construction allows a simple description of the configuration space of rigid origami mechanisms. We will describe the configuration of the origami by a vector listing the locations of the vertices. If there are n vertices,

$$q^T = (x_1, y_1, z_1, \ldots, x_n, y_n, z_n) \tag{50}$$

Assume there are m struts. Then there are m distance constraints:

$$f(q) = \begin{pmatrix} \|e_1\|^2 - l_1^2 \\ \|e_2\|^2 - l_2^2 \\ \vdots \\ \|e_n\|^2 - l_n^2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{51}$$

where e_i is the vector between the two vertices contained in edge i, and l_i is the (initial) length of edge i. The constraints described by equation 51 describe the configuration space of the rigid origami. Locally, we expect the configuration space to be a surface embedded in R^{3n}. The mobility at a configuration is the dimension of the tangent space to the surface, if the tangent space is defined. The planning problem may be stated: given two configurations satisfying equation 51, find a continuous connecting path that satisfies equation 51 at every point.

The formulation described will allow rigid body motions of the origami mechanism. The easiest way to deal with this problem would be to introduce additional constraints fixing the locations of key vertices. For reasons of efficiency and numerical stability, it is better to factor fixed vertices out of the configuration q.

8.2 Moving on the surface

In this section, I will discuss a simple continuation method for moving on a part of the surface that is not too near any singularity.

8.2.1 Euler steps

Since equation 51 must hold at all time, $\dot{f}(q) = 0$. Apply the multi-variable chain rule to calculate $\dot{f}$ in terms of $\dot{q}$.

$$\dot{f}(q) = J(q)\dot{q} = 0 \tag{52}$$

The matrix $J(q)$ is the $m \times n$ constraint Jacobian whose columns are formed by taking partials of f with respect to each configuration variable ($x_1, \ldots, z_n$).

Equation 52 means that the tangent to any trajectory of the system must be in the null space of the constraint Jacobian. This suggests a way of moving around on the surface: given an initial configuration, calculate the null space of the constraint Jacobian, N. Choose $\dot{q}$ to be a linear combination of columns of N. Pick some time step Δt, and use an Euler integration step to find a new q.

$$q(t + \Delta t) = q(t) + \Delta t \dot{q} \tag{53}$$

8.2.2 Normal steps

Each time we apply the first-order integration step described by equation 53, we expect the new value for q to be slightly off the surface. That is, if $f(q(t)) = 0$, then we expect $f(q(t+\Delta t)) \neq 0$. The larger the Δt, the worse we expect the method to perform. After a number of time steps, the divergence may be large. Following a sequence of Euler steps by a constraint satisfaction step that moves in a direction normal to the surface can ameliorate the problem.

We can estimate the change in the error function f using the first-order Taylor approximation:

$$f(t + \Delta t) = f(t) + \Delta t \dot{f} \tag{54}$$

We can write this in terms of $\dot{q}$ using equation 52:

$$f(t + \Delta t) = f(t) + \Delta t J \dot{q} \tag{55}$$

If there is some error ($f \neq 0$), then one approach is to pick $\dot{q}$ to attempt to remove the error in one time step. We want to choose $\dot{q}$ to satisfy

$$f(t + \Delta t) = 0 = f(t) + \Delta t J \dot{q} \tag{56}$$

We can solve for $\Delta t \dot{q}$

$$\Delta t \dot{q} = -J^{+} f(t) \tag{57}$$

where J^{+} is the pseudoinverse of J. Substituting into the Euler step equation (53), we derive a *normal step* equation:

$$q(t + \Delta t) = q(t) - J^{+} f(t) \tag{58}$$

8.2.3 Efficient calculation

Taking an Euler step requires that we find a basis for the null space of the constraint Jacobian at each time step. There are many ways to accomplish this. To take advantage of the sparsity of J, the best choice seems to be to apply a sparse QR decomposition to J^{T}.

Sparsity should also be taken advantage of in the calculation of $J^{+} f(t)$ when taking a normal step. Since J is not square or symmetric, either a sparse QR decompostion or an iterative method would be good choices to solve equation 56.

8.3 Planner implementation

I used the methods of moving on the surface to design a simple graph search planner. Nodes in the graph correspond to configurations of the system. We begin by choosing some configuration as the root node. To take Euler steps, an orthogonal null space basis for the constraint Jacobian is calculated. The Euler step is then chosen from the null space. It seems reasonable to choose the vectors of the null space basis as primitive actions for the search. The outline of the search algorithm is as follows:

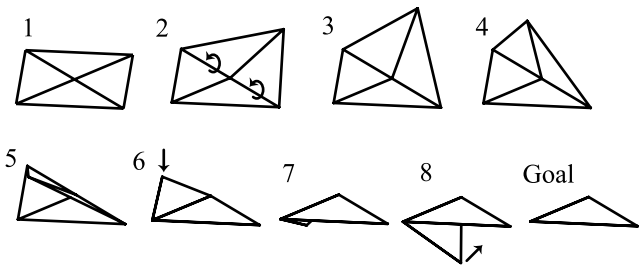


Figure 29: An automatically generated plan to fold origami with two creases.

1. Choose a starting configuration to be the root node.
2. Add the root node to a priority queue with priority zero.
3. Pop a node from the priority queue.
4. Check if the node reached the goal.
5. Expand the node by calculating the null space of J and taking steps in each direction.
6. Take normal steps to bring each child node back to the surface.
7. Check a hash table to see if the child nodes are near previously generated configurations and prune.
8. Add the child node to a hash table used for pruning.
9. Check each child node for collisions.
10. Add each collision-free child node to the priority queue, with priority equal to one plus the parent node's priority.
11. If the priority queue is not empty, go to step 3.
12. Backchain to find the path from start to goal.

The next sections will discuss the details of the implementation. I implemented the planner in C, and applied it to the simple problem described in figure 29. The goal was to make one two 180 degree folds to fold the origami under a single facet. The plan shown was found after searching about a thousand nodes.

8.3.1 Joint limits

A collision occurs if two adjacent facets attempt to pass through each other. If each crease is viewed as a hinge joint, collisions of this type may be prevented by implementing simple joint limits. Given two nearby configurations q_1 and q_2, calculate each crease angle; choose a convention so that the angles fall in $[-\pi, \pi]$. (The maximum

distance between q_1 and q_2 should be chosen to be small relative to the length of the shortest strut.) If the crease angle changes by a large amount ($> \pi$, for example), then any motion between the two configurations must violate the joint limit.

8.3.2 Facet collisions

Non-adjacent facets may also collide. I segment each facet into triangles and use a simple ray-triangle intersection test to check for collisions of facets. Currently, I do not check for intersections along a path segment, but only at single configurations.

8.3.3 Pruning using a hash table

During planning, it is useful to be able to answer the question of whether a new configuration is identical or near to a configuration that has already been generated. Since the configuration space is a surface, configurations are sparse in the embedding space. A hashtable is therefore a efficent way to store the set of visited configurations, and allows queries to be answered quickly.

The hash table must be designed so that two configurations which are nearby map to the same bucket. One method is to discretize the embedding space into small hypercubes. In order to store a configuration q, convert each coordinate into an index into the discretized space, yielding a vector of integers, which we will call q_d. We can then use a simple hash function to hash the q_d.

Define p_h to be the expected size of the hash table. p_h should be a prime not too near a power of 2. Define p_d to be the number of hypercubes per dimension; p_d should also be prime. For one example problem, I chose $p_h = 30011$. I expected each vertex to fall in a cube with sides of length two. I chose $p_d = 47$, so that the sides of each hypercube were slightly larger than $2/47 \approx .04$.

I used the following hash function:

$$h(q_d) = \left(\sum_{i=1}^{3n-1} d_i p_d^{i-1} \right) \bmod p_h \tag{59}$$

where d_i are the elements of q_d. In order to implement the hash function, variable precision arithmetic was necessary; I used the GMP library version 4.1 (available from http://www.swox.com/gmp/ at the time of writing).

Multiple configurations are stored in each hash bucket by chaining. To answer the question of whether a configuration is close to a previously visited configuration, the hash value is computed and the configuration is compared to each configuration in the linked list at the corresponding hash bucket. If the Euclidean distance between two nodes is small, the node is assumed to have been visited.

One problem with this method is illustrated by each of the four (really eight!) corner nodes in figure 25. There may be no trajectory between configurations that are nearby as measured by a Euclidean metric. A partial solution would involve checking for collisions along a short path segment between nearby configurations. If there were a collision, both configurations would be kept.

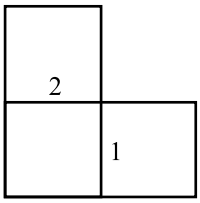


Figure 30: Three facets of a flattened cube.

8.4 Evaluation

I have only applied the planner to very simple problems. However, some interesting issues and questions have already been raised.

8.4.1 Limitations of struts

If a facet has more than three sides, the submatrix formed by taking the corresponding rows in the constraint Jacobian will not have full row rank. Physically, this corresponds to the fact that the (planar) edge lengths constraints in each facet only constrain the vertices to remain in the plane in a second-order fashion. This means that Euler steps tend to violate the planarity of facets with four vertices. Although normal steps cause the constraint to be satisfied, the generation of spurious nodes greatly increases the computation required to find plans. This also suggests that using the constraint Jacobian to determine the topology of the configuration space will be difficult if there are facets with more than three edges. Adding an additional, out-of-plane vertex for each facet might solve the problem.

Another difficulty with the struts is the fact that making a 180 degree fold between two facets and making a -180 fold lead to physically different configurations of the paper. It is possible to detect transitions between these cases collision detection, but for the purposes of pruning the planner currently treats these configurations as identical.

8.4.2 Keyholes

The work on the topology of the configuration space suggests that plans will need to be able to find narrow passages (or keyholes) from one section of the c-space to another. In some cases, the planner described gets lucky: the plan shown in figure 29 does involve a keyhole of this type. For this problem, the second fold cannot be made until the first fold has been completed. Since the dimension of the null space basis is determined numerically, it is not necessary that the critical configuration be reached exactly.

Relying on numerical instability in the null space computation to find keyholes does not seem like a good long term solution! For the problem considered, the topological structure of the configuration space has already been determined; this should make planning easy, since the planner should only need to plan between the enumerated singular configurations. For more complicated problems, determining some topological properties of the configuration space will be necessary.

In addition to topological keyholes, there may be keyholes created by the geometry of the design. To find the plan shown above, it was necessary to disable collision detection. Consider the problem folding up both flaps of figure 30 to create a flat square. Once the first fold has been folded to 90 degrees, the second fold cannot be made past 90 degrees until the first fold has been completed. Completing the second fold at all relies on the thin-ness of the paper. One possible solution would involving 'melding' facets once they become co-planar.

8.4.3 Step sizes and a heuristic

Exploring the configuration spaces of more complicated designs will require larger step sizes and possibly a heuristic. Currently, the distance between nodes on the search graph is limited by the distance over which an Euler step remains close to the surface. The number of nodes in the graph is further increased by the fact that the combination of Euler and normal steps generates configurations that are near the surface, but almost never on it.

8.5 Proposed thesis work

The thesis work will address some of the issues raised. I will implement a planner that can find foldings of the six- and eight-link closed chains involved in folding the crane and the balloon. The planner will incorporate some information about the topology of the configuration space.

The planner described finds foldings of simple origami, but it is hard to see how to implement the plans on a robot. In order to connect the planning work with the experimental work, I will also consider the problem of grasp analysis. Where can a set of fingers be placed to immobilize the origami? The constraint Jacobian suggests a methodical way of approaching the problem. If we add sufficient constraints such that the constraint Jacobian is square and has full rank (or has more rows than columns, and has full column rank), then the rigid origami will be immobilized.

I will consider placing fingers to add the additional constraints. Since the fingers will apply unilateral forces, the constraints cannot be directly appended to the constraint Jacobian. However, as long as all motions in the null space of the constraint Jacobian would violate the unilateral constraints, the rigid origami will still be immobilized.

Once I have found a grasping algorithm, I will conduct experiments to determine its applicability to real (flexible) origami. I will build fixtures to hold the origami, and apply some set of external forces to determine if the grasp is reliable.

9 Summary of proposed thesis work, and timeline

What skills must a robot possess in order to fold origami? I analyzed the folding procedure for each of the origami shown in figure 31. Each of the designs shown can be folded using seven primitives: *position*, *fold down*, *flip*, *separate*, *unfold*, *closed chain manipulation*, and *flex*. One way to classify the difficulty of an origami design is

Figure 31: Origami requiring various levels of manipulation skill.

by the number of primitives that must be used to fold it. The following table lists the primitives typically used to fold each of the designs:

Design	Pos.	Fold	Flip	Sep.	Unfold	Closed chains	Flex
Envelope	✓	✓					
Airplane	✓	✓	✓				
Samurai	✓	✓	✓	✓			
Crane	✓	✓	✓	✓	✓	✓	
Balloon	✓	✓	✓	✓	✓	✓	✓

Alternate folding procedures may allow some primitives to be eliminated. For example, it is possible to fold the balloon without flexing any links by pre-creasing.

The primitives provide a good guide for the exploration of origami folding. The first three, *position*, *fold down*, and *flip*, are simple, but are sufficient to fold interesting shapes, including the paper airplane. I will design and build a set of tools that will allow a four-DOF Adept arm to execute these actions. A tentative design was discussed. The *separate* and *unfold* primitives are more complicated, and my design will only partially address the possibilities.

In order to fold the crane or the balloon, the paper is pre-creased, and multiple creases are manipulated simultaneously. The completed work proposed a rigid-body model of origami for the purposes of exploring skills of this type. I have analyzed the topology of the configuration space for some simple origami designs, and implemented a rudimentary planner.

The thesis work will further explore the topology of the configuration space for pre-creased origami. I will implement a planner that can plan trajectories for pre-creased origami with up to eight creases and a single closed chain. In order to connect the theoretical work with the experimental work, I will also evaluate the problem of grasping origami with fingers. I will design an algorithm that will place fingers to immobilize origami with a large number of facets. I will also build some fixtures to immobilize origami, and conduct experiments to determine how well the origami is grasped.

The following table indicates the goals of the thesis work, and the current status, classified by manipulation primitive. A filled circle (●) indicates that the current status is satisfactory, or that some aspect the thesis will thoroughly explore the problem. A hollow circle (○) indicates that the completed work partially solves the problem, or that the thesis will explore at least some aspects of the problem. The left side shows the status of the completed work, and the right the goals of the proposed work.

Primitives	Exp.	1D bend.	Topol.	Plan	Exp.	Topol.	Plan	Grasp
Position	●				●			
Fold down	○				●			
Flip	○				●			
Separate					○			
Unfold					○			
Closed chains			○	○	○	●	●	●
Flex links		○				○	○	

I expect to complete the thesis by August, 2004. My tentative research plan includes the following components:

- **Writing 1.** The dissertation.
- **Writing 2.** The dissertation and the defense.
- **Grasp analysis 1.** Develop an algorithm for static grasping of simple rigid origami.
- **Grasp analysis 2.** Explore the problem of time-varying grasps, and the problem of flexible links.
- **Topology 1.** Prove some things about the topology of the c-space of rigid origami with up to six links and a single closed loop, ignoring joint limits and self-intersection.
- **Topology 2.** Explore the implications of joint limits and multiple loops. Consider the problem of representing topology for a planning problem.
- **Planning 0.** Apply the planner already developed to more complicated problems by adding heuristics.
- **Planning 1.** Develop a planner that either encodes some knowledge of the topology of the configuration space of closed chains, or automatically determines some topological properties. Plan for mechanisms with five to six links, and a single closed loop.
- **Planning 2.** Plan for mechanisms with six to eight links, and a single closed loop.
- **Experiments and Engineering 1.** Implement the proposed design, or an alternate design that allows *positioning*, *folding down*, and *flipping* of the paper, to a higher degree of accuracy than the current system.
- **Experiments and Engineering 2.** Refine the implemented design, and apply the system to a simple origami test suite.
- **Experiments and Engineering 3.** Design and conduct experiments to test the results of the grasping work.

- **Experiments and Engineering 4.** Refine all experimental designs, collect final data, and collate video documentation of experiments.

My expected schedule is:

Winter 2002	March 1	Planning 0	Grasping 1
Spring 2003	June 1	Topology 1	Experiments 1
Summer 2003	September 1	Grasping 2	Experiments 2
Fall 2003	December 1	Planning 1	Experiments 3
Winter 2003	March 1	Topology 2	Experiments 4
Spring 2004	June 1	Planning 2	Writing 1
Summer 2004	September 1	Writing 2	Writing 2

10 Conclusion

My thesis will present origami folding as an exciting challenge problem for the field of robotic manipulation. Folding origami involves manipulating flexible closed chains with a large number of degrees of freedom. However, we know that origami can be folded. Through an exploration of origami folding, my thesis will give insight and provide partial solutions to some very hard manipulation problems.

11 Acknowledgments

My work has been supported by a Department of Energy Computational Sciences Graduate Fellowship. Thanks to my advisor, Matthew T. Mason, for all the usual, and a lot more. Thanks to my committee (Doug James, James Kuffner, and Jeff Trinkle) for their time and insight. Thanks to Vandi Verma and Siddhartha Srinivasa for their comments on proposal drafts. Meetings and discussions with James Milgram, Alfred Rizzi, David Bourne, Katsu Yamane, Jessica Hodgins, Jeff Schneider, Chris Atkeson, Srinivas Akella, David Hershberger, Kiran Bhat, Gabe Brisson, and Apostolos Lerios have also been a great help. Thanks!

References

[1] M. Anitescu and F. Potra. Formulating multi-rigid-body contact problems with friction as solvable linear complementarity problems. *ASME Journal of Nonlinear Dynamics*, 14:231–247, 1997.

[2] S. S. Antman. *Nonlinear Problems of Elasticity*. Number 107 in Applied Mathematical Sciences. Springer-Verlag, 1995.

[3] U. Ascher and P. Lin. Sequential regularization methods for simulating mechanical systems with many closed loops. *SIAM Journal on Scientific Computing*, 21(4):1244–1262, 1999.

[4] G. Aumann. Interpolation with developable Bézier patches. *Computer Aided Geometric Design*, 8:409–420, 1991.

[5] D. J. Balkcom, E. Gottlieb, and J. Trinkle. A sensorless insertion strategy for rigid planar parts. In *Proceedings, IEEE International Conference on Robotics and Automation*, 2002.

[6] D. Baraff. Linear-time dynamics using Lagrange multipliers. In *SIGGRAPH*, pages 137–146, Aug. 1996.

[7] D. Baraff and A. Witkin. Large steps in cloth simulation. In *SIGGRAPH*, pages 43–54, July 1998.

[8] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact, and friction for cloth animation. In *SIGGRAPH*, 2002.

[9] G. Chirikjian. Inverse kinematics of binary manipulators using a continuum model. *Journal of Intelligent and Robotic Systems*, 19:5–22, 1997.

[10] K.-J. Choi and H.-S. Ko. Stable but responsive cloth. In *SIGGRAPH*, 2002.

[11] J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley, second edition, 1989.

[12] E. D. Demaine and M. L. Demaine. Recent results in computational origami. In *Proceedings of the 3rd International Meeting of Origami Science, Math, and Education*, 2001.

[13] M. A. Erdmann. On motion planning with uncertainty. Master's thesis, Massachusetts Institute of Technology, Aug. 1984.

[14] C. Fernandes, L. Gurvits, and Z. Li. A variational approach to optimal nonholonomic motion planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 680–685, Apr. 1991.

[15] C. Gauss. *Disquisitiones generales circa superficies curvas*. Göttingen: Dieterich, 1828. (English translation by A. Hiltebietel and J. Morehead: Hewlett, NY, Raven Press, 1965.).

[16] M. Gleicher. *A Differential Approach to Graphical Manipulation*. PhD thesis, Carnegie Mellon University, 1994.

[17] H. Goldstein, C. Poole, and J. Safko. *Classical Mechanics*. Addison-Wesley, third edition, 2002.

[18] S. Gupta, D. Bourne, K. K. Kim, and S. S. Krishnan. Automated process planning for robotic sheet metal bending operations. *Journal of Manufacturing Systems*, September 1998.

[19] L. Han and N. M. Amato. A kinematics-based probabalistic roadmap method for closed chain systems. In *Workshop on the Algorithmic Foundations of Robotics*, pages 233–246, May 2000.

[20] D. Hilbert and S. Cohn-vossen. *Geometry and the Imagination*. Chelsea, 1952.

[21] S. Hirai, T. Tsuboi, and T. Wada. Robust grasping manipulation of deformable objects. In *Proceedings of the IEEE Symposium on Assembly and Task Planning*, pages 411–416, May 2001.

[22] S. Hirai, H. Wakamatsu, and K. Iwata. Modeling of deformable thin parts for their manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2955–2960, 1994.

[23] D. Huffman. Curvature and creases: A primer on paper. *IEEE Transactions on Computers*, C-25(10):1010–1019, Oct. 1976.

[24] H. Inoue and M. Inaba. Hand-eye coordination in rope handling. In *Robotics Research: The First International Symposium*, pages 163–174, 1985.

[25] D. L. James and D. K. Pai. A unified treatment of elastostatic contact simulation for real time haptics. *Haptics-e, the Electronic Journal of Haptics Research*, 2(1), September 2001. `http://www.haptics-e.org`.

[26] L. E. Kavraki, F. Lamiraux, and C. Holleman. Towards planning for elastic objects. In *Workshop on the Algorithmic Foundations of Robotics*, pages 313–326, 1998.

[27] Y. L. Kergosien, H. Gotoda, and T. L. Kunii. Bending and creasing virtual paper. In *IEEE Computer Graphics and Applications*, pages 40–48, Jan. 1994.

[28] R. Lang. Trees and circles: an efficient algorithm for origami design. In *Proceedings of the 3rd International Meeting of Origami Science, Math, and Education*, 2001.

[29] S. M. LaValle, J. H. Yakey, and L. E. Kavraki. A probabalistic roadmap approach for systems with closed kinematic chains. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1999.

[30] W. Lenhart and S. Whitesides. Reconfiguring simple polygons. *Discrete and Computational Geometry*, 1994.

[31] S. Leopoldseder and H. Pottmann. Approximation of developable surfaces with cone spline surfaces. *Computer Aided Design*, 30:571–582, 1998.

[32] P. Lötstedt. Coulomb friction in two-dimensional rigid-body systems. *Zeitschrift für Angewandte Mathematik und Mechanik*, 61:605–615, 1981.

[33] T. Lozano-Perez. A simple motion-planning algorithm for general robot manipulators. *IEEE Journal of Robotics and Automation*, RA-3(3):224–238, June 1987.

[34] L. Lu and S. Akella. Folding cartons with fixtures: A motion planning approach. In *Proceedings of the IEEE International Conference on Robotics and Automation*, May 1999.

[35] L. Lu and S. Akella. Folding cartons with fixtures: A motion planning approach. *IEEE Transactions on Robotics and Automation*, 16(4):346–356, Aug. 2000.

[36] R. J. Milgram. personal communication, Nov. 2002.

[37] R. J. Milgram and J. Trinkle. The geometry of configurations spaces for closed chains in two and three dimensions. *Submitted to Homology, Homotopy, and Applications*, forthcoming.

[38] M. Minsky. Manipulator design vignettes. Technical report, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1972.

[39] S. Miyazaki, T. Yasuda, S. Yokoi, and J. Toriwaki. An interactive simulation system of origami based on virtual space manipulation. In *Proceedings of the IEEE International Workshop on Robot and Human Communication*, pages 210–215, 1992.

[40] H. Nakagaki, K. Kitagaki, T. Ogasawara, and H. Tsukune. Study of deformation and insertion tasks of a flexible wire. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2397–2402, Apr. 1997.

[41] D. K. Pai. STRANDS: Interactive simulation of thin solids using cosserat models. In *Eurographics*, 2002.

[42] P. Painlevé. Sur les lois du frottemont de glissement. *C. R. Acad. Sci.*, 121:112–115, 1895.

[43] H. Pottmann and J. Wallner. Approximation algorithms for developable surfaces. *Computer Aided Geometric Design*, 16:539–556, 1999.

[44] P. Redont. Representation and deformation of developable surfaces. *Computer Aided Design*, 21(1):13–20, 1989.

[45] M. Rubin. *Cosserat Theories: Shells, Rods, and Points*. Kluwer Academic Publishers, 2000.

[46] G. Song and N. M. Amato. A motion planning approach to folding: From paper craft to protein folding. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 948–953, May 2001.

[47] D. Stewart and J. Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal of Numerical Methods in Engineering*, 39:2673–2691, 1996.

[48] D. E. Stewart. Existence of solutions to rigid body dynamics and the paradoxes of Painlevé. *Comptes Rendus de l'Acad. Sci., Sér. I*, 1997. To appear.

[49] M. Sun and E. Fiume. A technique for constructing developable surfaces. In *Graphics Interface*, pages 176–185, May 1996.

[50] K. R. Symon. *Mechanics*. Addison-Wesley, third edition, 1971.

[51] J. Thorpe. *Elementary Topics in Differential Geometry*. Springer-Verlag, 1979.

[52] J. Trinkle and R. J. Milgram. Motion planning for planar n-bar mechanisms with revolute joints. *Submitted to the International Journal of Robotics Research*, forthcoming.

[53] J. Trinkle, J. Pang, S. Sudarsky, and G. Lo. On dynamic multi-rigid-body contact problems with coulomb friction. *Zeitschrift für Angewandte Mathematik und Mechanik*, 77(4):267–279, 1997.

[54] T. Wada, S. Hirai, T. Hirano, and S. Kawamura. Modeling of plain knitted fabrics for their deformation control. In *Graphics Interface*, pages 176–185, May 1996.

[55] T. Wada, S. Hirai, S. Kawamura, N. Kamiji, and K. Tsukamoto. Robust manipulation of deformable objects by a simple pid feedback. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 85–90, May 2001.

[56] T. Wada, S. Hirai, S. Kawamura, N. Kamiji, and K. Tsukamoto. Robust manipulation of deformable objects by a simple position feedback. In *3rd Asia-Europe Congress on Mechatronics*, pages 112–117, Oct. 2001.

[57] T. Wada, B. J. McCarragher, H. Wakamatsu, and S. Hirai. Modeling of shape bifurcation phenomena in manipulations of deformable string objects. *International Journal of Robotics Research*, 15(8):833–846, 2001.

[58] H. Wakamatsu, S. Hirai, and K. Iwata. Static analysis of deformable object grasping based on bounded force closure. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3324–3329, 1996.

[59] H. Wakamatsu, T. Matsumura, S. Hirai, and E. Arai. Dynamic analysis of rod-like object deformation towards their dynamic manipulation. In *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 196–201, 1997.

[60] G. Weiss and P. Furtner. Computer-aided treatment of developable surfaces. *Computers and Graphics*, 12(1):39–51, 1988.