A Computational Algorithm for Origami Design

Robert J. Lang
7580 Olive Drive
Pleasanton, CA 94588
rjlang@aol.com

1. Introduction

1.1 Background

Origami is the Japanese name for the centuries-old art of folding paper into representations of birds, insects, animals, plants, human figures, inanimate objects, and abstract shapes. In the purest form of origami, the figure is folded from a single uncut square of paper. In the past 50 years, the complexity of origami figures has increased dramatically, evolving from simple, stylized shapes to highly detailed representations of crustacea [1], insects [2, 3], dinosaurs [4, 5], and other multi-legged, multi-winged, and generally multi-appendaged creatures. As the range of origami subject matter has expanded in complexity, the problem of folding a base — a geometric shape with flaps for each of the appendages of the subject — has assumed greater and greater importance. Indeed, the problem of folding a multi-pointed base constitutes the bulk of that branch of the art known as origami *sekkei*, or "technical folding."

Throughout the history of origami, most origami design has been carried out by a combination of trial and error and/or heuristic techniques based on the folder's intuition. In this paper, I present for the first time a complete algorithm for the design of an arbitrary origami figure, specifically, for the solution of a crease pattern that folds flat into a base with any desired number of flaps of arbitrary length, which become the arms, legs, wings, etc., of the origami figure. The algorithm is based on a set of mathematical conditions on the mapping between the crease pattern and a tree graph representing the base. In this fashion, the problem is transformed into a nonlinear constrained optimization, which as it turns out, is closely related to existing circle packing and triangulation algorithms. I have implemented the algorithm in a computer program written in C++ that is available on the Internet; with it one can compute the crease pattern for origami designs of unprecedented complexity and sophistication.

1.2 Previous Work

The problem of designing an origami shape based on the number of appendages has been recognized for years [6] and several workers have addressed various aspects of origami design [7–9], albeit typically at a conceptual rather than an algorithmic level. In recent years, several authors, notably Meguro [10], Maekawa [11], Kawahata [12], and Kawasaki [13] have described useful geometric algorithms for the design of a base in which the target shape is represented as a stick figure, and flaps of the base are represented by nonoverlapping circles and/or circular contours. I have also described a geometric algorithm for origami design in [14]. The present work builds upon and generalizes concepts presented in and implicit in [10–14], mathematically proves the underlying theory, and completes the algorithm begun in [14] in a form suitable for computer implementation.

1.3 Outline of the Approach

The basis of many complex origami designs is a geometric shape called a base, which is a folded configuration of the original square that has the same number of flaps of the same length as the appendages of the subject. Once the folder has a base with the right number of flaps, it is relatively easy to transform it into a recognizable representation of the subject, so for complex subjects, at least, most effort is focused on folding the base to begin with.

I begin in section 2 by establishing a mathematical definition of a base and several useful measures of its properties, including the tree graph that forms an abstraction of the base and that serves to define the target design. In section 3 I establish a set of necessary conditions relating key vertices of the crease pattern to properties of the desired base. These necessary conditions help identify "active paths" — primary valley creases that subdivide the crease pattern into convex polygons, which correspond to distinct portions of the base. Sections 4 and 5 introduce the "universal molecule," an algorithm for filling in the creases in each polygon. Section 6 describes the mathematical and computer implementation of the algorithm and gives an example of its usage in the construction of an origami base.

2. Sufficient Conditions

2.1 What is a Base?

To start, I will establish some definitions:

A **crease pattern** is a division of the unit square into a finite set of polygonal regions by a set of straight line segments. Each segment is called a **crease**. Each polygon, which is bounded by a combination of creases and the edge of the square, is called a **facet** of the crease pattern.

A **base** is a non-stretching transformation of the unit square into 3-space such that the facets remain flat, i.e., all folding occurs only on creases. A base can therefore be fully defined by the locations of the creases, their angles, and the orientation and location of a facet.

A **flat-foldable** crease pattern is any crease pattern that can be folded into a base so that all layers of the base lie in a common plane.

A base may have several properties:

1. Projectability. A **projectable** base can be oriented in 3-space so that all facets of the base are perpendicular to the xy plane. The **projection** of any base or element of a base is the projection of that element into the xy plane. That is, the projection of a point (x,y,z) is the point (x,y). Since the facets of a projectable base are perpendicular to the xy plane, the projection of any facet of a projectable base is a line segment.

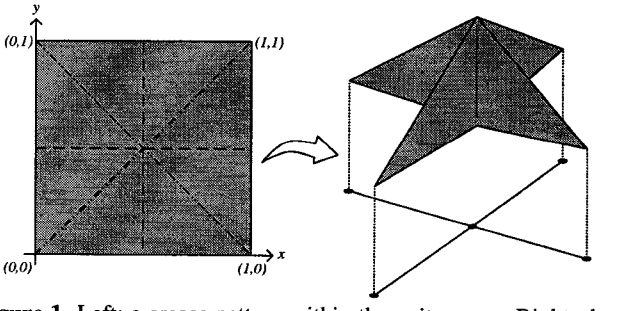

Figure 1. Left: a crease pattern within the unit square. Right: the crease pattern embedded in 3-space and transformed into a base. The projection of the base lies in the xy plane.

The projections of facets that meet along a crease are line segments that touch at a point or overlap along a line segment.

Computational Geometry'96, Philadelphia PA, USA

Since the facets of a crease pattern are connected, the set of all line segments formed by projecting individual facets into the xy plane is connected as well and forms a planar embedding of a **tree graph**. The line segments thus formed are the edges of the graph; points where two edges come together are nodes of the graph. Nodes that have exactly one edge connected to them are **terminal nodes**; the remaining nodes are **internal nodes**.

Definition: a **flap** is a group of facets in a base that project to a common edge of the tree graph such that:

1. Every facet in the flap projects to the same edge;

2. Any facet in the base that projects to the edge and is connected to a facet of the flap is itself a part of the flap.

3. The boundary of the facets that make up the flap projects to one or two nodes that are the endpoint(s) of the edge.

Therefore each flap in a base is associated with an edge of the tree graph. In many cases, the flaps of a projectable base can be positioned at different angles, in 3-space giving projected tree graphs that differ in the arrangement of their edges. Note that two side-by-side flaps may project to overlapping line segments in the xy plane; we still distinguish different flaps by associating them with distinct edges of the tree graph. To avoid ambiguity, a tree graph will always be drawn with nonoverlapping edges, but it should be understood that such a graph could represent a base with side-by-side flaps or even one flap wrapped around another, as shown in figure 2, whose projections include overlapping edges.

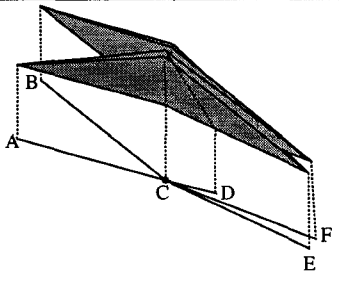

Figure 2. A base with wrapped flaps. Flaps D, E, and F, although wrapped around each other, correspond to distinct edges of the tree graph.

Each edge of a tree graph has a length associated with it, which is the length of the edge in the projection. Two tree graphs are considered to be identical if they are topologically equivalent and corresponding edges have the same weights. The act of projecting a base onto a tree graph provides a mapping from the crease pattern to the base and thence to the tree graph.

A few more properties:

2. Completeness. A **complete** base is a base in which every facet belongs to a unique flap. Note that in a complete projectable base, all "hinges" between connected flaps are parallel to the z axis.

3. Connectedness. A **simply connected base** is a base whose tree graph is simply connected (i.e., contains no closed loops).

4. Orientability. An **oriented** base is a base that lies in the half-space $z \geq 0$ such that the intersection of the base with the xy plane is identical to its projection in the xy plane. Put differently, an oriented base is a base that precisely covers its shadow.

Finally, a **uniaxial base** is an oriented, complete, simply connected, projectable base. These definitions provide a mathematical way of characterizing a uniaxial base in terms of the length and number of its constituent flaps by constructing the tree graph from the base. The length of the flaps are simply defined by the lengths of their corresponding edges on the tree graph. The algorithm in this work addresses the inverse problem: given a tree graph, construct a uniaxial base whose projection is the desired graph.

2.2 Comment

Throughout the history of origami, most but not all origami bases have been uniaxial bases. The four classic bases of traditional Japanese origami — the Kite, Fish, Bird, and Frog bases — may be configured to be uniaxial, as can many more complicated and sophisticated bases exploited by modern technical folders [3,4,6–9]. It should be noted, however, that there are many bases that are not uniaxial (notably Montroll's Dog Base [2] and its derivatives [5]) as well as curved and/or three-dimensional structures, all of which require different design strategies.

3. Necessary Conditions

3.1 Tree Conditions

The transformation from the two-dimensional crease pattern into the three-dimensional base can be expressed by a mapping $F(P) \rightarrow Q$, where P is a point in the unit square and Q is the corresponding point in the 3-space representation of the base. According to the conventions of origami transformations (no stretching, no self-intersection) [15–16], F is one-to-one and I will call F the transformation operator.

Similarly, a mapping $G(Q) \rightarrow R$ expresses the projection of the point Q onto the tree graph. G, which I will call the projection operator, is obviously not one-to-one, since all points with the same x and y coordinates map onto the same point on the tree graph. If S is the crease pattern in the unit square, the transformation of S into a base B and thence into a tree graph T can be written $T=G(F(S))$ when F is known. The design of a uniaxial base boils down to the inversion of this problem: given T, find the mapping F and crease pattern S that transform and project S into T. As I will show, knowledge of T places some restrictive conditions on the crease pattern S that are key to the solution of the problem.

Definition: A **path** $L(P_1,P_2)$ is a straight line between two points (P_1, P_2) in the unit square.

Theorem 1: For any two points P_1 and P_2 on the crease pattern, the length of path $L(P_1,P_2)$ is at least the length of the transformed path $G(F(L))$ in the embedded tree.

Proof: Consider such a path L, which in S consists of a collinear set of line segments l_i, one for each facet crossed by the path. The length of this path is given by $\sum_i l_i$. Upon transformation into the base, the path is transformed into a piecewise linear line $F(L)$ of the same length $\sum_i l_i$ since, within each facet, F preserves distances. However, upon taking the projection of the transformed path $G(F(L))$, each segment will be reduced in length by a factor $\cos \alpha_i$, where α_i is the angle between the transformed segment and the xy plane. Therefore, the length of the projection of the path is $\sum_i l_i \cos \alpha_i$. Since all of the multipliers are 1 or less, the length of the projection must be less than or equal to the length of the original path.

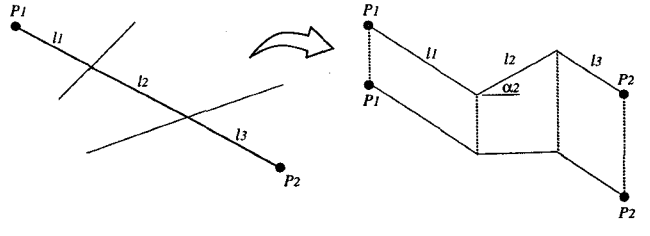

Figure 3. Left: a path crossing facet boundaries in the unit square. Right: the transformed path is piecewise linear. The length of the original path is greater than or equal to the length of its projection on the embedded tree.

Corollary 1: If P_1 and P_2 are two points in the crease pattern S and $F(P_1)$ lies in the xy plane, the length of the path $L(P_1,P_2)$ is equal to the length of $G(F(L))$ if and only if $F(L)$ lies in the xy plane.

Proof: By theorem 1, equality holds only when each of the cosines is equal to 1 and the angles α_i are zero. Therefore each of the line segments of the transformed path is parallel to the xy plane. Since one endpoint of the path transforms to the xy plane and the path is parallel to the xy plane, the entire path must lie in the xy plane. Conversely, if the transformation lies in the xy plane, then all of the angles α_i are zero by definition.

Definition: A **vertex** is a point in the unit square where two or more creases come together.

Definition: A **terminal vertex** is a vertex whose transformation lies in the xy plane and whose projection is a terminal node of the tree graph.

Lemma 1. The length of the projection of a path between two terminal vertices is greater than or equal to the sum of the lengths of the edges between the two corresponding terminal nodes on the tree graph.

Proof: Since the tree graph is simply connected, the projection of the path must at a minimum include the edges between the two corresponding terminal nodes. Consequently, its length is greater than or equal to the sum of the lengths of the edges.

These definitions and results lead to the fundamental theorem underlying the algorithm of this paper:

Theorem 2: For any uniaxial base, the distance between any two terminal vertices in the crease pattern is greater than or equal to the sum of the lengths of the edges connecting the two corresponding terminal nodes in the tree graph.

Proof: Consider the path between two terminal vertices. According to theorem 1, the length of the projection of the path is less than or equal to the length of the path, which is the distance between the two vertices. But since the projection of the path must lie on the tree graph, and the tree graph is simply connected, the edges that lie between the corresponding terminal nodes must be a subset of the edges that comprise the projection of the path. Consequently, the sum of the lengths of the connecting edges is less than or equal to the length of the projection, which is less than or equal to the length of the original path.

3.2 Discussion

Theorem 2 establishes a set of necessary conditions on the terminal vertices of a crease pattern that folds into a base that projects to a specified tree graph. Given a tree with N terminal nodes, the terminal vertices $\mathbf{u}_i$ in the unit square must satisfy the $N(N-1)/2$ conditions $\|\mathbf{u}_i - \mathbf{u}_j\| \geq l_{ij}$ where l_{ij} is the distance between the terminal nodes as measured along the tree graph. I will call these conditions the **tree conditions**. Note that while the conditions of theorem 1 apply to every possible pair of points in S, the tree conditions apply only to terminal vertices. In the next sections, I will show that the tree conditions are by themselves sufficient for the existence of the desired crease pattern.

4. Subtrees and Active Paths

Definition: an **active path** is a path between two terminal vertices whose length is equal to the sum of the lengths of the edges connecting the two corresponding terminal nodes in the tree graph.

Theorem 3. Every active path is either a crease or the edge of the square.

Proof: According to corollary 1 above, since the length of the active path equals the length of its projection, the transformed active path must lie in the xy plane. But since facets touch the xy plane only at their boundaries, the active path must lie along facet boundaries. Since the boundaries of facets are either creases or the edge of the square, every active path must be composed of creases or the edge of the square. Since both paths and the edges of the square are straight lines, if any portion of a path is the edge of the square, the entire path must be the edge of the square. Consequently, the path is either wholly crease or wholly square edge.

Thus, given a tree graph, one can begin to design a crease pattern that folds into the corresponding uniaxial base by identifying a set of terminal vertices that satisfy the tree conditions; then the active paths between terminal vertices are a subset of the creases of the base. It remains to identify the remaining creases.

It can be shown that active paths cannot cross; they touch only at terminal vertices. Thus, active paths do more than define creases; they break up the unit square into regions that may be investigated independently.

Definition: an **active polygon** is a region of the unit square containing no terminal vertices in its interior whose boundary is formed exclusively by active paths.

Definition: a **subbase** is the transformation of an active polygon into a portion of the base.

Definition: a **subtree** is the projection of a subbase.

Observe that if every terminal vertex lies on the boundary of at least one active polygon, then the union of all of the subtrees corresponding to active polygons is the complete tree graph. Also note that the edges of an active polygon, being active paths, must lie in the xy plane after transformation. Thus, if one can identify the creases within each active polygon that transform it into a subbase, the individual subbases can figuratively be stitched together at their edges (the active paths) to form the overall base whose projection is the desired tree graph.

5. Molecules

5.1 The Universal Molecule

Within origami, there has been a recognition that certain crease patterns show up again and again in origami bases and the term *bun-shi* (**molecule**) was coined by Meguro [10, 11] to describe various regular patterns. I will adopt this terminology in this work to apply to the creases that fill an active polygon. The goal is to identify creases along which the active polygon can be folded into its corresponding subbase, which is itself a uniaxial base. It is easy to show that any convex polygon can be folded into some uniaxial base using a construction analogous to the Maekawa molecule (see figure 7). However, we need something stronger:

Theorem 4. Any active polygon that satisfies the tree conditions for a given subtree can be folded into a uniaxial base whose projection is the subtree.

Proof: The proof is by construction. Consider an active polygon with a corresponding subbase and subtree. By definition, the edges of the active polygon are active paths and the transformed edges of the subbase lie in the xy plane. All paths between adjacent terminal vertices are active; all paths between nonadjacent terminal vertices are not active, i.e., none are at their minimum length.

Suppose we inset the boundary of the polygon by a distance h, as shown in figure 4. If the original vertices of the polygon were A_1, A_2,... then we will label the inset vertices A_1', A_2',.... I will call the inset polygon a **reduced polygon** of the original polygon. The boundary of the reduced polygon is formed by the intersection of a plane parallel to the xy plane at height $z=h$ with the subbase.

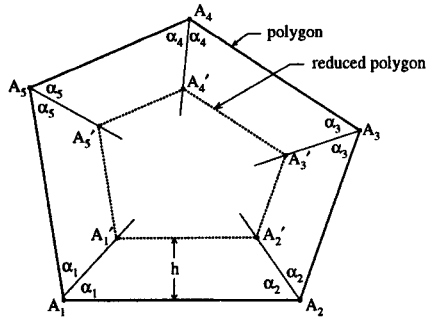

Figure 4. A reduced polygon is inset a distance h inside of an active polygon. The inset corners lie on the angle bisectors (dotted lines) emanating from each corner.

Note that the points A_i' lie on the bisectors emanating from the points A_i for any h. Consider first a reduced polygon that is inset by an infinitesimally small amount. In the subbase, the sides of the reduced polygon all lie in a common plane, just as the sides of the original active polygon all lie in a common plane; however, the plane of the sides of the reduced polygon is offset from the plane of the sides of the active polygon by a perpendicular distance h. As we increase h, we shrink the size of the reduced polygon. Is there a limit to the shrinkage? Yes, there is, and this limit is the key to the universal molecule. Recall that for any polygon that satisfies the tree conditions, the path between any two vertices satisfies a path length constraint

$$\left|A_i - A_j\right| \geq l_{ij}, \tag{1}$$

where l_{ij} is the path length between nodes i and j measured along the tree graph. There is an analogous condition for reduced polygons; any two vertices of a reduced polygon must satisfy the condition

$$\left|A_i' - A_j'\right| \geq l_{ij}', \tag{2}$$

where l_{ij}' is a **reduced path length** given by

$$l_{ij}' = l_{ij} - h\left(\cot\alpha_i + \cot\alpha_j\right) \tag{3}$$

and α_i is the angle between the bisector of corner i and the adjacent side. I call equation (2) the **reduced path constraint** for a reduced polygon of inset distance h. Any path for which the reduced path constraint becomes an equality is, in analogy with active paths between nodes, called an **active reduced path**.

So for any distance h, there is a unique reduced polygon and a set of reduced path constraints, each corresponding to one of the original path constraints. The reduced path constraints are parameterized by the inset distance h. We have already assumed that all of the original path constraints are met; thus, we know that all of the reduced path constraints are met for the h=0 case (no inset distance). It can also be shown that there is always some positive nonzero value of h for which the reduced path constraints hold. On the other hand, as we increase the inset distance, there comes a point beyond which one or more of the reduced path constraints is violated. Suppose we increase h to the largest possible value for which every reduced path constraint remains true. At the maximum value of h, one or both of the following conditions will hold:

(1) For two adjacent corners, the reduced path length has fallen to zero and the two inset corners are degenerate; or

(2) For two nonadjacent corners, a path between inset corners has become an active reduced path.

These two situations are illustrated in figure 5.

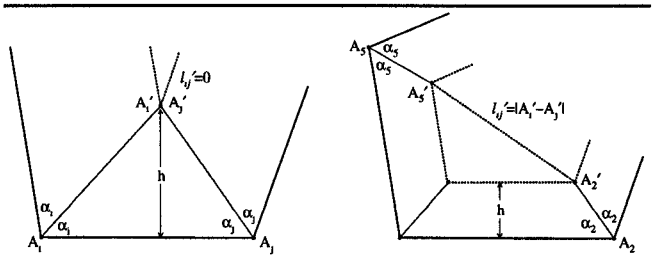

Figure 5. (Left) Two corners are inset to the same point, which is the intersection of the angle bisectors. (Right) Two nonadjacent corners inset to the point where the reduced path between the inset corners becomes active.

As I said, one or the other or both of these situations must apply; it *is* possible that paths between both adjacent and nonadjacent corners have become active simultaneously or that multiple reduced paths have become active for the same value of h. In either case, the reduced polygon can be simplified, thus reducing the complexity of the problem.

In a reduced polygon, if two or more adjacent corners have coalesced into a single point, then the reduced polygon has fewer sides (and paths) than the original active polygon. And if a path between nonadjacent corners has become active, then the reduced polygon can be split into separate polygons along the active reduced paths, each with fewer sides than the original polygon had. In either situation, you are left with one or more polygons that have fewer sides than the original. The process of insetting and subdivision is then applied to each of the interior polygons anew, and the process repeated as necessary.

If a polygon (active or reduced) has three sides, then there are no nonadjacent reduced paths. The three bisectors intersect at a point, and the polygon's reduced polygon evaporates to a point, completing the desired crease pattern.

Four-sided polygons can have the four corners inset to a single point or to a line, in which case no further insetting is required, or to one or two triangles, which are then inset to a point. Higher-order polygons are subdivided into lower-order ones in direct analogy.

Since each stage of the process absolutely reduces the number of sides of the reduced polygons created (although possibly at the expense of creating more of them), the process must necessarily terminate. Since each polygon (a) can fold flat, and (b) satisfies the tree conditions, then the entire collection of nested polygons must also satisfy the tree condition. Consequently, *any* active polygon that satisfies the tree conditions can be filled with a crease pattern using the recursive procedure outlined above and collapsed into a base on the resulting creases.

Each polygon so divided consists of two parts: the **core** is the reduced polygon (which may be crossed by active reduced paths); the border around the core is the **ring**. The angle bisectors that cross the ring become mountain folds of the crease pattern. Active reduced paths that cross the core such as $A_2'A_5'$ in figure 5 become valley folds. In addition, there are creases that emanate from each vertex of the crease pattern that are perpendicular to active paths; these may be mountain, valley, or flat (uncreased), depending on the desired orientation of the flaps. The same rules for the assignment of creases apply to each level of the recursive construction.

For triangular active polygons, there is no recursion; the three angle bisectors of the triangle meet at a point. The bisectors form mountain folds and the perpendiculars from the intersection of the bisectors to the three edges become valley folds. The crease pattern thus obtained is well known in the origami literature. The

procedure is called a rabbit ear and the crease pattern is thus called the rabbit ear molecule. It is shown in figure 6.

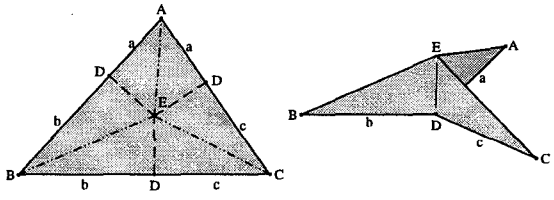

Figure 6. Crease pattern for a rabbit ear molecule and resulting subbase. Note that the three flaps of the subbase have the same lengths as the three edges of the planar graph.

The construction of the rabbit ear molecule is straightforward. If $\mathbf{p}_A$, $\mathbf{p}_B$, and $\mathbf{p}_C$ are the vector coordinates of the three corners A, B, and C, and $\mathbf{p}_E$ is the coordinate of the bisector intersection, then $\mathbf{p}_E$ is given by the simple formula

$$\mathbf{p}_E = \frac{\mathbf{p}_A(b+c)+\mathbf{p}_B(c+a)+\mathbf{p}_C(a+b)}{2(a+b+c)}.$$

That is, the location of the bisector intersection is simply the average of the coordinates of the three corners with each corner weighted by the length of the opposite side.

For quadrilaterals and higher-order polygons, one or more levels of recursion may be required. Figure 7 illustrates the four possible results of this procedure applied to quadrilateral active polygons. The patterns for both triangles and quadrilaterals have been previously identified in the origami literature as types of molecules; however, it is clear that the process described above can be applied to polygons of arbitrary complexity. I call this general construction, which to my knowledge has not been previously described, the **universal molecule** algorithm.

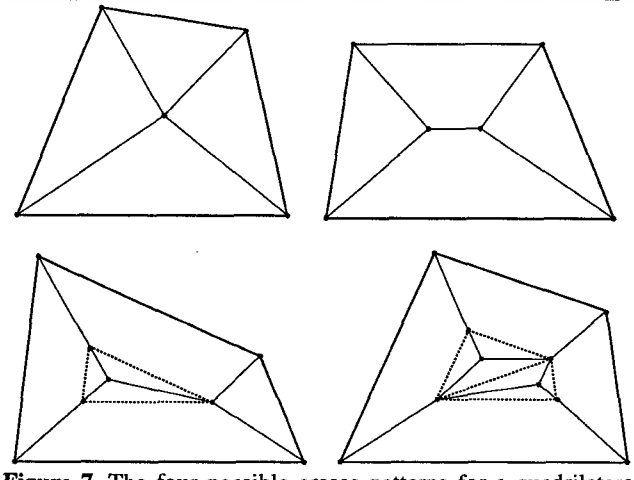

Figure 7. The four possible crease patterns for a quadrilateral active polygon. If all four corners are inset to the same point, the result is the Husimi molecule (top left). If adjacent pairs of corners are inset to two points, the Maekawa molecule is obtained (top right). If the inset polygon is a triangle, it is filled in with a rabbit ear molecule, which also results in a Maekawa molecule (bottom left). Finally, if the inset polygon is a quad crossed by an active reduced path, the result is the "gusset quad" molecule.

5.2 Fracture Algorithm

An alternative strategy to filling in a large active polygon using the universal molecule algorithm is to break it into smaller active polygons. This can be accomplished by adding a new edge — a **stub** — somewhere in the subtree of such length and location that the new terminal vertex forms four active paths with existing vertices of the polygon. This has the effect of breaking up the polygon into polygons of lower order. Such addition is always possible; there are four path conditions that must be satisfied and four degrees of freedom: the x and y coordinates of the terminal vertex, the length of the new edge, and the location on the tree where the new edge joins the original tree. This procedure is a generalization of Meguro's method of overlapping circles for filling in quadrilateral molecules [10, 11]. I call it the **fracture algorithm** for polygons.

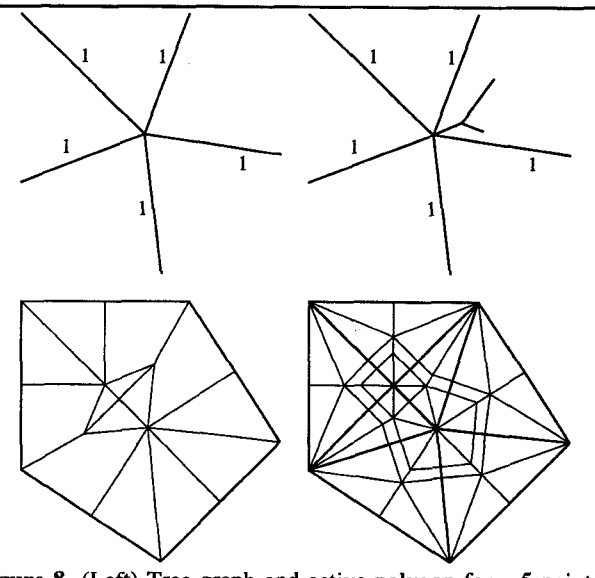

Figure 8. (Left) Tree graph and active polygon for a 5-pointed base, filled in with the universal molecule. (Right) By adding stubs to the tree, the polygon is fractured into all triangular active polygons.

Adding a vertex to the interior of an order-N polygon with four active paths creates four new polygons. However, the largest new polygon can have at most N–1 sides (in which case the other three polygons are triangles). By recursively applying the fracturing algorithm to a set of active polygons, any crease pattern can be reduced entirely to rabbit ear molecules.

5.3 Crease Assignment

In a valid crease pattern, crease lines will be either valley folds, mountain folds, or uncreased (0° fold angle). The algorithms described above identify the creases of the base; however, they must still be assigned. As mentioned earlier, the angle bisector folds in the universal molecule are mountain folds and active reduced paths are valley folds. Between active polygons, the active paths, which form the boundaries between active polygons, are all valley folds for terminal vertices that lie on the boundary of the square. For terminal vertices in the interior, the situation gets more complicated. Maekawa's theorem [13] states that for an interior vertex, the number of mountain and valley creases around the vertex differ by ±2; thus, one of the active paths connected to an interior terminal vertex must be converted to a mountain fold along at least a portion of its length.

In addition to active paths, active reduced paths, and angle bisectors, there is a family of creases that must be constructed that radiate from every vertex of the crease pattern and are perpendicular to the active paths and reduced active paths. Most of these creases are in fact uncreased; however, some also become mountain and/or valley folds. The assignment of these remaining creases depends upon the specific configuration chosen for the flaps of the base. If a flap is turned one way, some creases will be valley, others will be mountain, and some will be uncreased; turn the flap the other way, and the assignments change.

Based on the foregoing algorithms, there remains some ambiguity in the assignment of creases for the base to fold flat. Bern and Hayes recently showed that the problem of determining a flat-foldable assignment of mountain and valley folds for an arbitrary crease pattern is *NP*-complete [17]. In a crease pattern constructed by the tree conditions/universal molecule, quite a bit of information is already known about the identity of most of the creases, which may reduce the complexity of the problem. However, a complete algorithm for crease assignment is still not yet known. As a practical matter, I find that once I know the location of all of the creases, I can find a valid crease assignment by empirical means.

6. Computational Implementation

6.1 Optimization Algorithm

The tree conditions must hold for every pair of terminal vertices and their corresponding terminal nodes on the tree. Thus, if there are N nodes, there are $N(N-1)/2$ inequality conditions that must be satisfied. Obviously the ease of satisfying all path conditions depends on the size of the tree graph relative to the size of the square. Since the longest possible path on the unit square is the diagonal with length $\sqrt{2}$, the tree cannot have any path longer than this. In practice, no path can be considered by itself, since the location of any terminal vertex affects the lengths of several different paths, and the longest possible path is often not even this long.

Generally, the origami designer seeks the largest possible base for a given size square. The most efficient base (having the fewest layers of paper) is obtained for the largest possible tree graph. The size of the tree graph can be related to the square by a scale factor m; that is, each edge of the tree graph is given a length that is a multiple of m. The most efficient base is that with the largest numerical value of m. Then the solution of a node pattern for a base corresponding to a tree graph can be found by solving a nonlinear constrained optimization:

> Given a weighted tree graph P with nodes P_i and P_j, define l_{ij} as the length of the path between P_i and P_j on the tree. Let $\mathbf{u}_i$ be the vector coordinates of node i in the unit square with x and y components $u_{i,x}$ and $u_{i,y}$. Then an optimally efficient crease pattern is found by maximizing m over all $\mathbf{u}_i$ subject to the constraints:
>
> (a) $m\, l_{ij} - \left[\left(u_{i,x}-u_{j,x}\right)^2 + \left(u_{i,y}-u_{j,y}\right)^2\right]^{1/2} \le 0$ for all i, j
>
> (b) $u_{i,x} \le 1$, $u_{i,x} \ge 0$, $u_{i,y} \le 1$, $u_{i,y} \ge 0$ for all i.

This nonlinear constrained optimization may be solved using the well-known Augmented Lagrangian Multiplier algorithm (ALM) [18]. Due to the simplicity of the merit function and constraints, the gradient of the augmented Lagrangian may be explicitly evaluated, which allows the use of a first-order algorithm, e.g., a conjugate-gradient algorithm [19], for the outer minimization.

In addition, other constraints to implement esthetic conditions are readily added to the minimization framework. For example, if the desired base has a line of symmetry, one might require that two nodes be each other's reflection about a symmetry line of the square. To avoid terminal vertices in the interior of the square (which give rise to wrapped flaps as shown in figure 2), the coordinates of the terminal vertices may be constrained to lie on the boundary of the square. These and similar additional constraints may be simply mixed into the augmented Lagrangian.

Another benefit to ALM is that at the end of the optimization, active constraints — inequalities that are in fact equalities — are identified by having nonzero Lagrangian multipliers. Active constraints correspond to active paths; thus nonzero Lagrangian multipliers identify primary valley folds of the crease pattern.

Once a pattern of nodes and active constraints is found that forms a network of active polygons, each polygon may be subdivided according to the universal molecule algorithm. This, too, is a constrained optimization; each level of the solution is found by maximizing the inset distance h subject to the reduced path constraints:

$$l_{ij} - h\left(\cot\alpha_i + \cot\alpha_j\right) - \left|A_i'(h) - A_j'(h)\right| \le 0 \text{ for all } i,j.$$

While this problem could be solved by ALM, the constraints are quadratic in h; thus, each equality can be solved analytically. Since the number of nodes in a given active polygon is usually rather small (polygons above order-6 are rare in practice), one can simply evaluate the expression for h for each pair of nodes in turn and select the largest value that still satisfies the other inequalities.

Alternatively, active polygons can be fractured into smaller polygons by adding a stub to the subtree of size, length, and location that create four active paths. In this case, we have four equalities with four unknowns (the x and y coordinates of the new node, the length of the stub, and the distance between the point of attachment and the nearest node), which may be solved by Newton-Raphson or equivalent algorithm.

After solving for the positions of all nodes, creation of additional nodes (if desired) and identifying active paths, valley and mountain creases are assigned as discussed in section 5.3. A simple example will illustrate this process.

Figure 9 illustrates a very simple tree graph for a typical mammal with four legs, a body, a head, and a tail. For simplicity in this example, I have chosen all edges to have the same length.

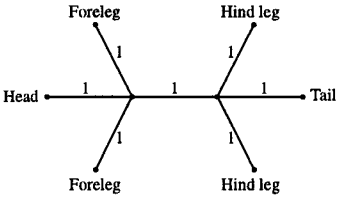

Figure 9. Planar graph for a six-flap base. Each edge of the graph has a length of 1 unit in this example.

Since there are 6 nodes, there are 15 possible paths between nodes; combining these constraints with four constraints for each node (restricting the node to lie within the square), there are a total of 39 inequality constraints. Application of ALM (or, in this case, simple algebra) reveals that an optimum distribution of nodes is the pattern shown in figure 10, which has a scale of $1/2\sqrt{((121+8\sqrt{179})/65)} \approx 0.267$.

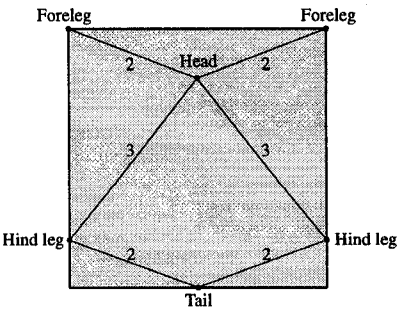

Figure 10. Node pattern that satisfies the tree conditions for the six-legged tree graph.

The active paths are illustrated by lines in figure 10, identifying active polygons. Each polygon corresponds to a subgraph of the original tree graph, as shown in figures 11 and 12. Note that

interior nodes of the tree graph correspond to vertices along active paths; these interior vertices, as well as those created within universal molecules radiate creases perpendicular to the active paths.

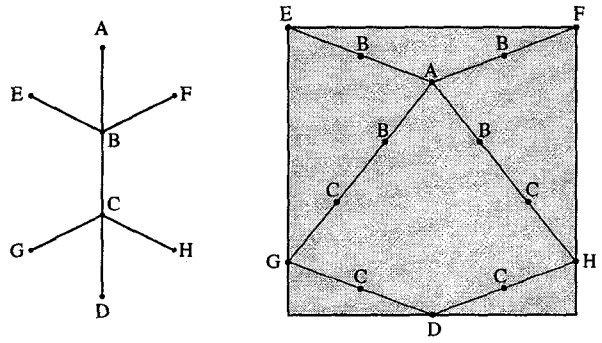

Figure 11. (Left). Planar graph with all nodes lettered. (Right) Crease pattern with terminal vertices, internal vertices, and active paths.

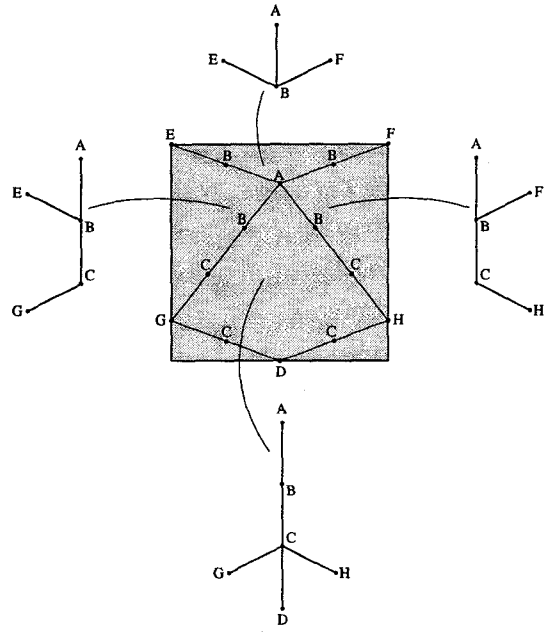

Figure 12. The four active polygons for the six-legged base and the planar graphs corresponding to each subbase.

Now the individual active polygons must be filled with creases according to the universal molecule algorithm.

When all of the active polygons are filled with creases, one obtains the pattern shown on the left in figure 13. Folding it up, one obtains the base on the right, which has the same configuration of flaps as the tree graph from which I began.

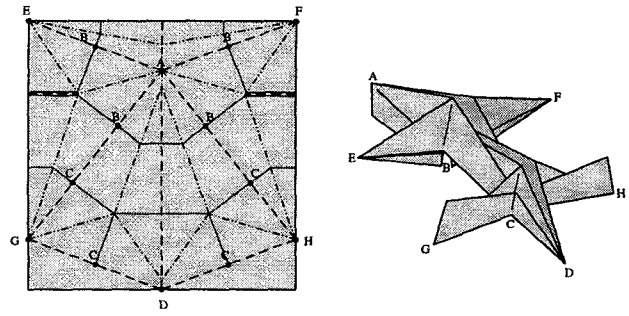

Figure 13. Full crease pattern and six-legged base.

It should be emphasized that the outcome of this algorithm is an origami base, not the final figure. The base has the right number and size flaps, but it is still up to the origami designer to shape the flaps into arms, legs, head, tail, et cetera. The techniques to do such shaping are, however, widely known and available in the origami literature.

6.2 Computational Implementation

The optimization of the terminal nodes and subsequent computation of creases are readily amenable to computer implementation using standard algorithms and conventional data structures. I have implemented this algorithm in C++ in a program called TreeMaker, using the ALM algorithm to perform the constrained optimization with the Polak-Ribiere conjugate gradient algorithm [19] for the outer minimization. Using this program, which is available via ftp on the Internet [20], one can construct origami bases of unprecedented complexity and sophistication. Although the example above was relatively simple, Figure 14 shows a real-world example: the tree graph for a six-point buck deer. Application of the tree algorithm results in the crease pattern shown on the right; folding on the crease lines gives a base which may be shaped into the deer shown in the photograph in figure 15. Performing the optimization of the terminal nodes required less than 1 minute on an 80 MHz Power Macintosh 601; computation of the creases was essentially instantaneous. Considering that conventional trial-and-error origami design of complex models can take anywhere from hours to years, the potential effect of this algorithm on the field is significant.

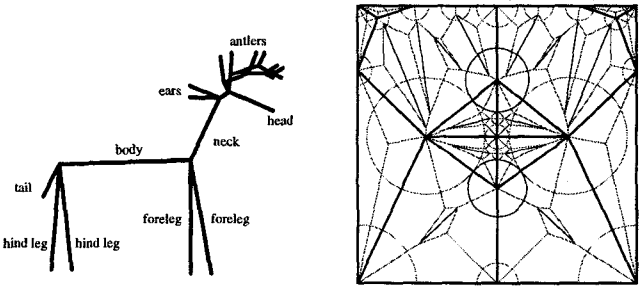

Figure 14. (left) Stick figure for a deer. (Right) computed crease pattern.

Once the creases are identified and assigned, there remains the very practical problem of how to transform the crease pattern into the base, that is, what is the order of folding? Although most origami designers go to some effort to develop and present the folding method as an ordered sequence, there is nothing inherent in the algorithms presented here that would give the resulting base a step-by-step folding sequence; and generally, the bases derived by this method cannot be folded one crease at a time. Instead, the paper must be precreased and then all creases brought together at once.

Furthermore, there is rarely a transformation from the square to the base that preserves facet flatness throughout the transformation. There are moves in the origami repertoire that, topologically speaking, cannot be completed with a finite number of creases, such as the "closed sink" (a move plentiful in [3]), which is equivalent to inversion of a cone — a known impossibility. In practice, the transformation of a computed crease pattern into an origami base, and thence into a figure as shown in figure 15, remains a challenging endeavor.

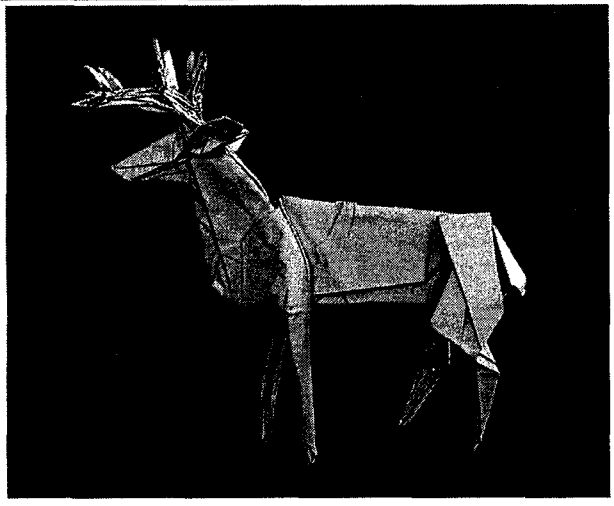

Figure 15. Folded deer.

6.3 Comment

As a final note, a special class of tree graphs consists of those in which all edges emanate from a single interior node. For such a graph, the tree conditions simplify: the distance between two terminal nodes must be greater than or equal to the sum of the lengths of the edges connected to each node. This is equivalent to the condition describing a dense packing of nonoverlapping circles of varying size where the radius of each circle corresponds to the length of the edge in the tree graph and the centers of the circles must lie within a square. This similarity has been exploited by several folders who have described origami design algorithms by representing flaps with circular arcs and contours [10–12]. Even for tree graphs with multiple interior nodes, the use of circles to represent terminal flaps aids in the visualization of the base. To show this relationship, figure 13 superimposes circles on the crease pattern where each circle represents a terminal flap of the base.

If in addition to emanating from the same node, the edges of a tree graph are all the same length, then the problem of designing a base with N flaps is equivalent to the densest packing of N equal circles, all of whose centers lie within a square. Since any such circle packing can be circumscribed by a square that touches all of the boundary circles, the problem is also equivalent to the densest packing of N circles wholly within a square, which is a problem that has seen some recent activity in its own right [21]. Implicit within every optimal circle packing, there is an optimal origami base and vice versa. It is interesting to note that only a handful of the known densest circle packings have been realized as origami designs — at least, to date.

Acknowledgements

The algorithms described here evolved out of the published works cited in the references and fruitful direct exchanges with Toshiyuki Meguro, Tom Hull, and Alex Bateman. The presentation in this paper was further refined by many helpful suggestions from Marshall Bern of Xerox PARC.

References

[1] John Montroll and Robert J. Lang, *Origami Sea Life*, Dover, 1990.

[2] John Montroll, *Origami for the Enthusiast*, Dover, 1979.

[3] Robert J. Lang, *Origami Insects*, Dover, 1995.

[4] Fumiaki Kawahata, *Fantasy Origami*, Gallery Origami House (Tokyo), 1995.

[5] John Montroll, *Prehistoric Origami*, Dover Publications, 1989.

[6] Kunihiko Kasahara, *Creative Origami*, Japan Publications, 1967.

[7] Kunihiko Kasahara and Toshie Takahama, *Origami for the Connoisseur*, Japan Publications, 1987.

[8] Kunihiko Kasahara, *Origami Omnibus*, Japan Publications, 1988.

[9] Peter Engel, *Origami from Angelfish to Zen*, Dover , 1994.

[10] Toshiyuki Meguro, private comm.

[11] Jun Maekawa, in *Oru*, vol. 2, no. 2, 1994.

[12] Fumiaki Kawahata, "The technique to fold free angles of formative art 'origami'," abstract in proceedings of *The Second International Meeting on Origami Science and Scientific Origami*, Otsu, Japan, 1994.

[13] Toshikazu Kawasaki, "On the relation between mountain-creases and valley-creases of a flat origami," in *Origami Science and Technology*, H. Huzita, ed., 1989, p. 229–237.

[14] Robert J. Lang, "Mathematical algorithms for origami design," *Symmetry: Culture and Science*, vol. 5, no. 2, pp. 115–152, 1994.

[15] Tom Hull, "On the mathematics of flat origamis," *Congressus Numerantium* 100, (1994), pp. 215–224.

[16] D. A. Huffman, *Curvatures and creases: a primer on paper*, IEEE Trans. on Computers, Volume C-25, (1976), pp. 1010–1019.

[17] Marshall Bern and Barry Hayes, "On the Complexity of Flat Origami," ACM Symposium on Discrete Algorithms, 1996.

[18] G. N. Vanderplaats, *Numerical Optimization Techniques for Engineering Design*, McGraw-Hill, 1984

[19] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, *Numerical Recipes in C, 2nd Ed.*, Cambridge, 1994.

[20] The program *TreeMaker* runs on Macintosh computers and is available with documentation via anonymous ftp at rugcis.rug.nl in directory origami/programs.

[21] Martin Gardner, "Tangent Circles," *Fractal Music and Hypercards*, W. H. Freeman, 1992, pp. 149–166, and references therein.