A. ALGORITHMS

A.1. Mathematical Model

Tree theory converts the problem of finding an efficient origami crease pattern into one of several different types of nonlinear constrained optimization, a type of problem that has been thoroughly studied by the computer science community and for which there are many known algorithms for solution.

The key quantity here is efficiency; the goal is to make the most efficient model (i.e., largest in proportion to the starting paper). The quantity to maximize is therefore the scale, which is the size of a one-unit flap compared to the size of the square. There are two families of constraints that must be satisfied for any valid crease pattern:

- The coordinates of every vertex must lie within the square.
- The separation between any two vertices on the square must be at least as large as the scaled length of the path between their corresponding two nodes as measured along the tree.

If there are N leaf nodes in a figure, the first condition sets $4N$ linear inequality constraints while the second sets $N(N–1)/2$ quadratic inequality constraints on the $2N$ vertex coordinates. In addition to these constraints, the designer can set a number of other constraints to enforce various symmetries in the model:

- A vertex can have its position set to a fixed value.
- A vertex can be constrained to lie on a line of bilateral symmetry.
- Two vertices can be constrained to lie symmetrically about a line of symmetry.
- Three vertices can be constrained to be collinear.
- An edge can be constrained to a fixed length.
- Two edges can be constrained to have the same distortion (strain).
- A path can be forced to be active.
- A path can have its angle set to a fixed value.
- A path can have its angle quantized to a multiple of a given angle.

The problem is solved by converting the path conditions of the tree theorem and all constraints into mathematical equations, specifically, a constrained nonlinear optimization. This section enumerates the equations that define each type of optimization.

A.2. Definitions and Notation

Define U to be the set of all vector-valued *vertex* coordinates $\mathbf{u}_i$, $i \in I^n$, where I^n is a set of vertex indices: $I^n = \{1..n_n\}$. Each vertex $\mathbf{u}_i$ has coordinate variables $u_{i,x}$ and $u_{i,y}$.

Define E to be the set of all *edges* e_i, $i \in I^e$, where I^e is a set of edge indices $I^e = \{1..n_e\}$. Each edge contains exactly two nodes $n_i, n_j \in e_k$. Each edge e_i has a length l_i and a fractional distortion, called *strain*, σ_i.

Each vertex $\mathbf{u}_i$, corresponds one-to-one with the node n_i.

Define U^t to be the set of *leaf vertices*, which are those vertices that correspond to a node connected to exactly one edge. Define I^{tn} to be the set of leaf node indices. Clearly, $U^t \subseteq U$ and $I^{tn} \subseteq I^n$.

Define P to be the set of all *paths*, p_{ij}, $i, j \in I^n$, $i \neq j$. Each path is identified by the indices of the nodes at each end of the path. Each path has a length, l_{ij}, which is given by the sum of the strained lengths of the edges in the path; that is,

$$l_{ij} \equiv \sum_{e_k \in p_{ij}} (1 + \sigma_k) l_k . \tag{A-1}$$

Define P^t to be the set of *leaf paths*, which are those paths that connect two leaf nodes.

Define m to be the overall *scale* of the tree.

Define w, h to be the *width* and *height* of the paper. The paper is a rectangle whose lower left corner is the origin $(0, 0)$ and whose upper right corner is the point (w, h).

A.3. Scale Optimization

The most basic optimization is the optimization of the positions of all leaf vertices and the overall scale of the design. This is equivalent to solving the problem:

$$\text{minimize } (-m) \text{ over } \left\{m, \mathbf{u}_i \in U^t\right\} \text{ s.t.:} \tag{A-2}$$

$$0 \le u_{i,x} \le w \text{ for all } \mathbf{u}_i \in U^t \tag{A-3}$$

$$0 \le u_{i,y} \le h \text{ for all } \mathbf{u}_i \in U^t \tag{A-4}$$

$$m \sum_{e_k \in p_{ij}} \left[(1 + \sigma_k) l_k\right] - \sqrt{\left(u_{i,x} - u_{j,x}\right)^2 + \left(u_{i,y} - u_{j,y}\right)^2} \le 0 \text{ for all } p_{ij} \in P^t . \tag{A-5}$$

A.4. Edge Optimization

Edge optimization is used to selectively lengthen flaps by the same relative amount to fill out a crease pattern. The scale *m* is fixed, and a subset of the edges E^s is subjected to the same variable strain σ. A subset of the leaf vertices U^s is allowed to move. The edge optimizer solves the problem:

$$\text{minimize } -\sigma \text{ over } \{\sigma, \mathbf{u}_i \in U^s\} \text{ s.t.} \tag{A-5}$$

$$0 \le u_{i,x} \le w \text{ for all } \mathbf{u}_i \in U^s \tag{A-6}$$

$$0 \le u_{i,y} \le h \text{ for all } \mathbf{u}_i \in U^s \tag{A-7}$$

$$m\left[\sum_{\substack{e_k \in p_{ij} \\ e_k \in E^s}} \left[(1+\sigma)l_k\right] + \sum_{\substack{e_k \in p_{ij} \\ e_k \notin E^s}} \left[(1+\sigma_k)l_k\right]\right] - \sqrt{\left(u_{i,x}-u_{j,x}\right)^2 + \left(u_{i,y}-u_{j,y}\right)^2} \le 0 \text{ for all } p_{ij} \in P^t. \tag{A-8}$$

Equation (A-5) can be broken into a fixed part and a variable part:

$$\underbrace{\sigma}_{\text{variable}} \cdot \underbrace{m \sum_{\substack{e_k \in p_{ij} \\ e_k \in E^s}} \left[l_k\right]}_{\text{fixed}} + \underbrace{m \sum_{\substack{e_k \in p_{ij} \\ e_k \notin E^s}} \left[(1+\sigma_k)l_k\right]}_{\text{fixed}} - \underbrace{\sqrt{\left(u_{i,x}-u_{j,x}\right)^2 + \left(u_{i,y}-u_{j,y}\right)^2}}_{\text{fixed or variable}} \le 0. \tag{A-9}$$

Note, however, that the last term may or may not be a mixture of fixed and variable parts depending upon which vertices are moving.

A.5. Strain Optimization

Strain optimization is used to distort the edges of the tree minimally in order to impose other global constraints, e.g., symmetry, particular angles, etc., on the overall crease pattern. As with edge optimization, the overall scale is fixed, but in this case, rather than *maximizing* the same strain for all affected edges, the strain optimization should *minimize* the RMS strain for a large set of edges with each edge potentially having a different strain. As with the edge optimization, there is a set of strainable edges E^s and a set of moving vertices U^s. The strain optimizer solves the problem:

$$\text{minimize } \sum_{e_i \in E^s} \sigma_i^2 \text{ over } \{\sigma_i\big|_{e_i \in E^s}, \mathbf{u}_j \in U^s\} \text{ s.t.} \tag{A-10}$$

$$0 \le u_{i,x} \le w \text{ for all } \mathbf{u}_i \in U^s \tag{A-11}$$

$$0 \le u_{i,y} \le h \text{ for all } \mathbf{u}_i \in U^s \tag{A-12}$$

$$m\left[\sum_{\substack{e_k \in p_{ij} \\ e_k \in E^s}}\left[(1+\sigma_k)l_k\right]+\sum_{\substack{e_k \in p_{ij} \\ e_k \notin E^s}}\left[(1+\sigma_k)l_k\right]\right]-\sqrt{\left(u_{i,x}-u_{j,x}\right)^2+\left(u_{i,y}-u_{j,y}\right)^2}\leq 0 \text{ for all } p_{ij}\in P^t. \qquad \text{(A-13)}$$

Equation (A-13) can also be broken into a fixed and variable part:

$$\sum_{\substack{e_k \in p_{ij} \\ e_k \in E^s}}\left[\underbrace{\sigma_k}_{\text{variable}}\ \underbrace{ml_k}_{\text{fixed}}\right]+\underbrace{m\left[\sum_{\substack{e_k \in p_{ij} \\ e_k \in E^s}}\left[l_k\right]+\sum_{\substack{e_k \in p_{ij} \\ e_k \notin E^s}}\left[(1+\sigma_k)l_k\right]\right]}_{\text{fixed}}-\underbrace{\sqrt{\left(u_{i,x}-u_{j,x}\right)^2+\left(u_{i,y}-u_{j,y}\right)^2}}_{\text{fixed or variable}}\leq 0. \qquad \text{(A-14)}$$

A.6. Optional Conditions

In addition to the conditions specified above, which are always required to be satisfied, one will commonly impose additional conditions on the crease pattern that are typically strict equalities. These additional conditions are placed to enforce various symmetries, either symmetries of the subject (that the model is bilaterally symmetric) or symmetries of the crease pattern (that major creases, i.e., active paths, lie at fixed angles). These conditions are easily incorporated into the nonlinear constrained optimization machinery as additional equality constraints. Several conditions and related equations commonly implemented in tree theory are listed below.

1. Vertex position fixed.

A vertex $\mathbf{u}_i$ that has one or both of its coordinates fixed to a point $\mathbf{a}$ must satisfy one or both of the equations

$$u_{i,x}-a_{i,x}=0 \qquad \text{(A-15)}$$

$$u_{i,y}-a_{i,y}=0. \qquad \text{(A-16)}$$

2. Vertex fixed to corner of paper.

A vertex $\mathbf{u}_i$ that is constrained to lie on a corner of the paper must satisfy both equations

$$\left(u_{i,x}-w\right)\cdot u_{i,x}=0 \qquad \text{(A-17)}$$

$$\left(u_{i,y}-h\right)\cdot u_{i,y}=0. \qquad \text{(A-18)}$$

3. Vertex fixed to edge of paper.

A vertex $\mathbf{u}_i$ that is fixed to lie on the edge of the paper must satisfy the equation

$$\left(u_{i,x}-w\right)\cdot u_{i,x}\cdot\left(u_{i,y}-h\right)\cdot u_{i,y}=0. \tag{A-19}$$

4. Vertex fixed to line.

A vertex $\mathbf{u}_i$ that is constrained to lie on a line through point **a** running at angle α, such as a line of symmetry, must satisfy the equation

$$\left(u_{i,x}-a_x\right)\cos\alpha-\left(u_{i,y}-a_y\right)\sin\alpha=0. \tag{A-20}$$

5. Two vertices paired about a line.

Two vertices $\mathbf{u}_i$ and $\mathbf{u}_j$ that are constrained to be mirror-symmetric about a line through point **a** running at angle α, such as a line of symmetry, must satisfy the two equations

$$\left(u_{i,x}-u_{j,x}\right)\cos\alpha-\left(u_{i,y}-u_{j,y}\right)\sin\alpha=0 \tag{A-21}$$

$$\left(u_{i,x}+u_{j,x}-2a_{i,x}\right)\sin\alpha-\left(u_{i,y}+u_{j,y}-2a_{i,y}\right)\cos\alpha=0. \tag{A-22}$$

6. Three vertices collinear.

Three vertices $\mathbf{u}_i$, $\mathbf{u}_j$, and $\mathbf{u}_k$ that are constrained to be collinear must satisfy the equation

$$\left(u_{j,y}-u_{i,y}\right)\left(u_{k,x}-u_{j,x}\right)-\left(u_{k,y}-u_{j,y}\right)\left(u_{j,x}-u_{i,x}\right)=0. \tag{A-23}$$

7. Edge length fixed.

An edge e_k whose length is fixed must have its strain satisfy the equation

$$\sigma_k=0. \tag{A-24}$$

8. Edges same strain.

Two edge e_j and e_k that have the same strain must satisfy the equation

$$\sigma_j-\sigma_k=0. \tag{A-25}$$

9. Path active.

A path p_{ij} between two vertices $\mathbf{u}_i$ and $\mathbf{u}_j$ that is constrained to be active must satisfy the equation

$$m \sum_{e_k \in p_{ij}} \left[(1+\sigma_k) l_k \right] - \sqrt{\left(u_{i,x} - u_{j,x}\right)^2 + \left(u_{i,y} - u_{j,y}\right)^2} = 0. \qquad \text{(A-26)}$$

10. Path angle fixed.

A path p_{ij} between two vertices $\mathbf{u}_i$ and $\mathbf{u}_j$ that is constrained to lie at an angle α must satisfy the equation

$$\left(u_{i,x} - u_{j,x}\right)\cos\alpha - \left(u_{i,y} - u_{j,y}\right)\sin\alpha = 0. \qquad \text{(A-27)}$$

11. Path angle quantized.

A path p_{ij} between two vertices $\mathbf{u}_i$ and $\mathbf{u}_j$ that is constrained to lie at an angle of the form $\alpha_k = \alpha_0 + k \cdot \delta\alpha$. where $\delta\alpha \equiv \frac{180°}{N}$ must satisfy the equation

$$2^{N-1} \cdot \left[\left[\left(u_{i,x} - u_{j,x}\right)^2 + \left(u_{i,y} - u_{j,y}\right)^2 \right]^{-N/2} \right] \cdot \left[\prod_{k=0}^{N-1} \left[\left(u_{i,x} - u_{j,x}\right)\sin\alpha_k - \left(u_{i,y} - u_{j,y}\right)\cos\alpha_k \right] \right] = 0. \qquad \text{(A-28)}$$

This function has the property that it goes to zero when the path is quantized, goes to ±1 in between quantized paths, and has no gradient component in the direction of shortening the path (which can cause problems when there are many such constraints). However, the code for the gradient of this function is rather complicated.

A.7. Stubs

A *stub* is a leaf node and edge added to the tree—usually emanating from the middle of an existing edge—such that the new leaf node creates exactly four active paths to other nodes in the tree.

When a stub is added inside of an existing *N*-sided polygon it breaks the polygon into 4 new polygons that all have fewer than *N* sides. Thus, by addition of stubs, high-order polygons are converted to lower-order polygons, until eventually all polygons in the crease patterns are triangles and can be filled with rabbit-ear molecules. This process is called triangulation of a crease pattern. By repeatedly adding stubs, any crease pattern can be fully triangulated.

I introduce a few more definitions:

A polygon *Q* is defined by a set of leaf vertices U^{tQ} that form the vertices of the polygon and a set of leaf paths P^Q, which are all paths that span U^Q.

Define the set of vertices U^Q and edges E^Q as all vertices and edges whose nodes are contained within one or more of the paths P^Q; U^Q and E^Q constitute the *subtree* of polygon *Q*. Define U^{tQ} as the vertices in U^Q that are also leaf vertices, i.e., $U^{tQ} = U^Q \cap U^{tn}$.

Let e_{ab} be an edge of the subtree with $\mathbf{u}_a$ the vertex at one end of e_{ab} and $\mathbf{u}_b$ the vertex at the other end.

In general, for every edge e_{ab} and set of four distinct vertices $\mathbf{u}_i$, $\mathbf{u}_j$, $\mathbf{u}_k$, $\mathbf{u}_l$, there is a stub that terminates on a new vertex $\mathbf{u}_m$; the stub is defined by four quantities:

- The vertex coordinates $u_{m,x}$ and $u_{m,y}$;
- The distance d_m from node $\mathbf{u}_a$ at which the new stub emanates from edge e_{ab};
- The length l_m of the new stub.

These four variables are found by solving the four simultaneous equalities

$$m\left[\sum_{e_k \in p_{ia}} l(e_k) + \begin{Bmatrix} d_m & \text{if } e_{ab} \in p_{ia} \\ -d_m & \text{if } e_{ab} \notin p_{ia} \end{Bmatrix} + l_m\right] - \sqrt{\left(u_{i,x} - u_{m,x}\right)^2 + \left(u_{i,y} - u_{m,y}\right)^2} = 0 \qquad \text{(A-29)}$$

for each of the four vertices $\mathbf{u}_i$, $\mathbf{u}_j$, $\mathbf{u}_k$, $\mathbf{u}_l$. Although a solution can potentially be found for any four vertices, not all are valid; the only valid combinations of vertices $\mathbf{u}_i$, $\mathbf{u}_j$, $\mathbf{u}_k$, $\mathbf{u}_l$ and edges e_{ab} are those for which (a) the four vertices are distinct, and (b) *both* signs of d_m are represented among the four equations. In addition, solutions for d_m that are negative or greater than the length of e_{ab} are nonphysical and must be discarded.

Note that for these equations to be used as written above, there can be no unrelieved strain in the system. They can clearly be modified to include strain.

A.8. Universal Molecule

The universal molecule is a crease pattern that is constructed by a series of repeated reductions of polygons. The construction is carried out by insetting the polygons and constructing reduced paths and fracturing the resulting network into still smaller polygons of lower order. As with triangulation, the process is guaranteed to terminate.

We use the same polygon definitions as were used in the description of stubs. In addition:

Assume the vertices $\mathbf{u}_i \in U^Q$ are ordered by their index, i.e., the *N*-sided polygon contains the indices $\mathbf{u}_1$, $\mathbf{u}_2$,…$\mathbf{u}_N$, ordered as you travel clockwise around the polygon. Construct the following:

α_i is half of the interior angle at vertex $\mathbf{u}_i$. We define the quantity $\varsigma_i \equiv \cot \alpha_i$.

Define R_{90} as the operator that rotates a vector clockwise by 90°, i.e., $R_{90} \circ (u_x, u_y) \equiv (u_y, -u_x)$.

Define N as the operator that normalizes a vector, i.e., $N \circ \mathbf{u} \equiv \dfrac{\mathbf{u}}{|\mathbf{u}|}$.

$\mathbf{r}_i$ is the scaled bisector of the angle formed at $\mathbf{u}_i$, pointing toward the interior of the polygon with magnitude $\csc \alpha_i$. The vector $\mathbf{r}_i$ as well as ς_i can be constructed according to the following prescription:

$$\begin{aligned}
\mathbf{r}' &\equiv N \circ (\mathbf{r}_{i-1} - \mathbf{r}_i) \\
\mathbf{r}'' &\equiv N \circ (\mathbf{r}_{i+1} - \mathbf{r}_i) \\
\mathbf{r}''' &\equiv N \circ R_{90} \circ (\mathbf{r}'' - \mathbf{r}') \\
\mathbf{r}_i &= \frac{\mathbf{r}'''}{\mathbf{r}''' \cdot \left[R_{90} \circ \mathbf{r}'''\right]} \\
\varsigma_i &= \mathbf{r}_i \cdot \mathbf{r}'.
\end{aligned} \tag{A-30}$$

The inset distance h is the largest value such that

1. For every path p_{ij} of length l_{ij} between nonadjacent vertices $\mathbf{u}_i$ and $\mathbf{u}_j$,

$$\sqrt{\left(u_x + hr_x\right)^2 + \left(u_y + hr_y\right)^2} \le m\left[l_{ij} - h\left(\varsigma_i + \varsigma_j\right)\right]. \tag{A-31}$$

2. For every path p_{ij} between adjacent vertices $\mathbf{u}_i$ and $\mathbf{u}_j$,

$$h \le \frac{\mathbf{u} \cdot \mathbf{u}}{\mathbf{u} \cdot \mathbf{r}} \tag{A-32}$$

where

$$\mathbf{u} \equiv \mathbf{u}_i - \mathbf{u}_j \tag{A-33}$$

$$\mathbf{r} \equiv \mathbf{r}_i - \mathbf{r}_j. \tag{A-34}$$

Although this problem is also a nonlinear constrained optimization, since there is only one variable and the number of paths is typically small, it can be solved directly by simply solving for each equality for all possible paths p_{ij} and taking the smallest positive real value of h found. The second relation, Equation (A-32), gives the value for h by replacing the inequality by equality. The solution for equality for relation (1), Equation (A-31), which is quadratic in h, is given by the following sequential substitutions:

$$\begin{aligned}
w &= \varsigma_i + \varsigma_j \\
a &= \mathbf{r} \cdot \mathbf{r} - w^2 \\
b &= \mathbf{u} \cdot \mathbf{r} + l_{ij} w \\
c &= \mathbf{u} \cdot \mathbf{u} - l_{ij}^2 \\
h &= \frac{-b + \sqrt{b^2 - ac}}{a} \, .
\end{aligned} \qquad \text{(A-35)}$$

Obviously, negative or complex values of *h* should be ignored.

Once a solution is found, we create a set of new reduced vertices $\mathbf{u}_i'$ and reduced paths p_{ij}' of length l_{ij}' :

$$\mathbf{u}_i' = \mathbf{u}_i + h\mathbf{r}_i \qquad \text{(A-36)}$$

$$l_{ij}' = l_{ij} - h\left(\varsigma_i + \varsigma_j\right). \qquad \text{(A-37)}$$

The reduced vertices and reduced paths may then be checked for active status and subdivided into polygons, and the process repeated.