**Folding any 3D shape.**

Saul Griffith*5, James McBride, Benjamin Su, Bobby Ren and Joseph M. Jacobson

*Center for Bits and Atoms, MIT Media Laboratory, Massachusetts Institute of Technology. 20 Ames St, Cambridge, MA, 02139

$Squid-Labs LLC, 1467 Park Ave, Emeryville, CA, 94608.$

Contact: saul@media.mit.edu
The folding of linear chains into 3D structure forms a crucial underpinning for a wide range of fields including protein folding, polymer packing, cellular robotics and aspects of self assembly and analytical geometry. It has long been appreciated empirically from the study of proteins that a diverse set of 3D structures may be folded from 1D strings. Here we show that such an approach, the mapping of 1D sequence to 3D structure, is complete, namely that arbitrary 2D or 3D structures may be created from a 1D string of a finite set of building blocks in a series of sequential, non-intersecting, folds. In addition we elucidate a general algorithm specifying such a string. The work suggests routes for the de novo design and generation of complex micro, nano and molecular scale structures. A built example is given in 2D.

Understanding how linear chains fold into 2D and 3D shapes has been a long-sought goal in diverse fields such as protein folding and design, polymer packing, materials fabrication, cellular robotics\(^1\), self assembly\(^2\)\(^3\)\(^4\)\(^5\), analytical geometry\(^6\)\(^7\), microelectromechanical systems\(^8\), and even children’s toys\(^9\). Biology has shown that encoding structural information in one dimension with a limited set of parts, and then folding to three dimensions is a powerful and general method of construction that enables both error correction and replication. While the diversity of three dimensional structures accessible by such an approach has long been appreciated\(^10\), this paper seeks to demonstrate the completeness of this approach. Here we show that a finite set of polygonal and polyhedral components can programmatically generate any 2D or 3D space-filling structure by folding according to simple rules embedded in the component order. Specifically we demonstrate that any 2-dimensional shape can fold from a linear string of not more than 4 types of vertex-connected squares in a series of sequential, non-intersecting folds. Similarly any 3D shape can be folded from an edge-connected string of not more than 2 types of right-angled tetrahedrons.

In order to be general a program to create 2D or 3D structure from 1D strings should enable a simple connected series of building blocks, (polyhedra in 2D, polyhedron in 3D) connected at vertices (2D, figure 1) and edges (3D, figure 4b) to fold sequentially (starting from one end) to form any pixelated(2D) or voxelated(3D) object. The resulting object should be space-filling (solid) when desired and should be guaranteed to be foldable in a finite number of sequential non-intersecting folds – folding deterministically according to the sequence embedded in the series of building blocks. Finally for the resulting structure to be useful we impose a final criterion that all of the bonds in the resulting

---

structure should be ‘strong’, that is that nearest neighbors in the final folded structure have opposite bonding characteristics such as N/S magnetic pole pairing, or a +/- attractive charge pairing.

Figure 1. A string of vertex-connected squares 1, 2, 3…n. Folding occurs sequentially from tile 1 to tile n. Folds are determined by polarities assigned to the faces of connected tiles (for example 1:2 = counterclockwise (CC), 2:3 clockwise (CW), 3:4(CC), 4:5(CC), 5:6(CC), 6:7(CC), etc…)

Geometric Proof in 2 Dimensions

The 1D sequence of parts required to create an arbitrary 2D shape can be determined by viewing the collection of pixels which make up the desired shape as a graph. The nodes of the graph are the centers of the pixels and the vertices in the graph connect adjacent pixels. In order to construct the shape by folding, it must be possible to connect all the nodes in the graph through a single non-intersecting path. Such a path is known as a Hamiltonian Path. Procedures for finding such a path when they exist are known, but not all graphs possess them.

As an example consider the dog shape of Figure 2a. If we begin tracing a path at the nose of the dog (pixel 1 - yellow), at the first branch (pixel 3 - roughly the eye of the dog) the path continues on to the ear above the head, but there is no return path to complete the body, legs and tail. As such, Figure 2a does not contain a Hamiltonian Path.

A solution to this problem can be achieved at the cost of increasing the number of pixels used by a factor of 4. To see why this helps it is necessary to introduce a final idea from graph theory. A spanning tree for a graph is a subgraph of the original graph that contains all the nodes and a subset of the vertices. Enough vertices must be included in this set to keep the original graph connected. Unlike Hamiltonian Paths every graph has a spanning tree (which is not necessarily unique). If a Hamiltonian Path does not exist, the spanning tree will be branched. If each original pixel is replaced by 4 sub-pixels, a one pixel wide perimeter is created around the original spanning tree (Figure 2b). This perimeter is a Hamiltonian Path (now explicitly a circuit) on the enlarged graph that takes the new sub-pixels as its nodes.

This construction can also be viewed inductively and it is this perspective that allows the procedure to be extended into 3D. Instead of laying out the shape, finding a spanning tree and quadrupling the pixel count to create the path, the path can be constructed by repeatedly adding sets of 4 sub-pixels until the desired shape is constructed. With each addition the path is extended to encompass the new sub-pixels. To demonstrate this, a whiskey barrel (yellow pixel) is added around the dog’s neck in figure 2b. The new node has a 4 sub-pixel Hamiltonian around it, similar to the zero path connected node in figure 1d. This new Hamiltonian will be joined to the original Hamiltonian circuit of the dog by severing both paths around the new connecting branch of the spanning tree. These two paths are highlighted in bold. The two new connecting paths are shown as green dashed lines and the new branch of the tree thus formed as a red dashed line. By similar constructions any additions can be made to the dog, or indeed any object can be built additively by combination of Hamiltonian circuits in this manner.

Figure 2. (a) A dog comprising square tiles without a Hamiltonian path. A spanning tree is shown in red lines connecting the nodes (red dots) at the centroid of each ‘pixel’. (b) The same dog where each pixel has been divided into 4 ‘sub-pixels’ enabling a Hamiltonian path or circuit one pixel wide around the perimeter of the spanning tree. The yellow tile shows the construction by addition of new tiles.

(c) The six possible ‘pixel’ configurations and their ‘sub-pixels’ demonstrating edge connectivity of (a = 0, b=1, c=2, d=2, e=3, f=4).

The minimal tile set for programmatically determining the folds according to the embedded sequence is presented in figure 3a. Tiles must always be presented head to tail according to the vector indicated. The polarities of the faces always guarantee one attractive and one repulsive force (e.g. N/S magnetic or +/- electrostatic) on either side of the joining vertex to enforce the folding rule. In 2D, 4 pixel types as defined by polarity of edges are required to produce a ‘strong’ or attractive bond at every face. These tiles, and the order in which they need to be placed in order to fold clockwise or counter-clockwise, are shown in Pink, Blue, Yellow, and Green.

Figure 3. (a) Magnetic poles of 4 different tile types allow clockwise and counter-clockwise folds at each vertex depending on the sequence of tiles: Pink(P), Green(G), Yellow(Y), Blue(B). P,G, G,Y, Y,B, B:P produce CW folds about the joining vertex, P:B, B:Y, Y,G, G:P ordering produce CC folds about the joining vertex. Any similar attractive / repulsive forces may be used eg: +/- charge or hydrophobic/hydrophylic. (b) The first 9 sequential folds of the ‘sequence’ for the dog of 3c starting at point A. (c) Any object (eg. The dog) can be superimposed on a regular plane tiling of the 4 tile types and such a construction implies the resulting ‘gene’ or tile sequence for folding that object. Gene is denoted by (B)lue, (P)ink, (Y)ellow, (G)reen tiles. The ‘gene’ sequence starts at tile A. For convenience a space is included in the sequence to denote corners.

Figure 3c. is the same dog superimposed on a regular tiling of the 4 tiles from figure 3a. It will be noted that all faces in the 2D tiling are a pairing of ‘strongly’ attractive polarities (N/S or S/N) guaranteeing a strongly cohesive structure. The ‘gene’ sequence for such a dog can be determined by this superposition and the gene for the dog starting at the tile denoted (A) is presented. It can be noted that more and less optimal positions for the start of the chain exist. (A) is more optimal than (B) as when starting at (B) the last string of tiles will need to fold down a deep trough in the structure, requiring some compliance of the structure to allow the folds. Starting at (A) allows a sequence that always adds tiles around the perimeter of a growing structure.

Geometric Proof in 3 Dimensions

Now we will extend the proof to 3 dimensions. The simplest case to consider is a cube constructed from 8 sub-cubes. This construction works, but the resolution is not minimal (8 sub-voxels required per voxel) and there is no guarantee that the resulting edge-connected string would lie flat in a plane amenable to micro-manufacture. The details of this construction may be found elsewhere, but are omitted here for the sake of brevity, in favor of a more optimal solution that has higher resolution and a planar, linear string irrespective of sequence.

First we need to determine the voxel in 3D. To reach all 3 dimensions by translation from each voxel, positive and negative translations in all 3 dimensional axes are required. One entry and exit sub-face per face is required for the Hamiltonian construction used thus far, as demonstrated in the 4 sides (+/- in two axes) of the square pixels in the 2D proof. The desirable ‘voxel’ (where voxel is the primitive unit of space, similar to pixel in 2D) therefore has 6 faces - one for positive, and one for negative translation in each axis. The voxel also requires 12 sub-faces derived from the sub-units that comprise it, two on each of the 6 faces analogous to the 2 subpixel faces on each pixel face in the 2D ease. The right-angled tetrahedron of figure 4a satisfies our constraints for a sub-voxel; it fills space, can be edge-connected linearly as shown in figure 4b, and 6 of them combine to make the rhombic hexahedral voxel shown in figure 4c. The Hamiltonian circuit for the voxel (equivalent to fig.2c.(a) is overlayed in red. The two blue edges of the tetrahedron separate faces at 90 degrees. The 4 red edges separate faces by 60 degrees. The rhombic hexahedron is space-filling with 6 pairs of sub-faces on 6 faces with normals to those faces on 3 axes.

Figure 4. a) Right angled tetrahedron. Faces subtend a 90 degree angle about blue edges and 60 degree angles about red edges. b) A string of edge connected right angled tetrahedra. c) A rhombic hexahedral voxel comprised of 6-right angled tetrahedral sub-voxels with Hamiltonian loop of connectivity shown in red. d)Fully face connected voxel (centre) is connected to 6 other voxels enabling a connected path to all surrounding voxels from any given voxel.
By analogy we can use the additive construction technique demonstrated for 2D to show that these tetrahedra can fold from a string to fill any voxelated 3D object where the voxels are rhombic hexahedron. The fully face connected case is presented in figure 4d where it is demonstrated that indeed, the return paths to 6 additional hexahedrons – 1 connected to each of the 6 faces of the original (central) hexahedron are possible. Figure 5a shows a simple 3D voxelated dog comprising 17 rhombic hexahedrons with the 17 Hamiltonians of each voxel in red. Figure 5b demonstrates how by a similar technique of splitting Hamiltonian circuits, one can grow the loop by adding other loops until a single loop for the entire object is available, where all connections are made about foldable edges. The construction in fig.5b. has been manually reproduced in laser-cut acrylic model right-tetrahedrons, and the 3D dog indeed folds from a linear string of edge-connected right-tetrahedra.

Interestingly, as is shown in Figure 6a only two sub-voxel types are required for 3D. The polarities for the faces in projection are shown as (top surface polarity) / (bottom surface polarity). Similarly to the 2D case, these will fold such that all faces in the 3D crystal will be strongly bound. The ‘gene’ for the 3d dog is shown in figure 6b and implies the methodology by which the sequence is defined by the construction.

**Compact Mapping Using Space Filling Curves**

The constructions which have been detailed so far require, as an upperbound, a multiple of four times as many tiles as there are pixels in the original shape. An interesting question is whether one can create such 1D to 2D or 3D mappings with fewer tiles. Here we show briefly that more compact mappings are possible using space filling curves.

Over a century ago, Peano\(^ {12}\), and soon after Hilbert\(^ {13}\), showed that an infinite recursive curve, called a space-filling curve, could completely fill space.

As an example consider a slightly more complicated 2D dog superimposed on a Hilbert curve (figure 7a). By severing the connections at the perimeter of the object and reconnecting them as is seen in figure 7b a more compact solution is found. We note that in this construction the resolution cost decreases as the size of the object increases as a greater number of pixels lie in the interior of the shape.

---

Materials and Methods
The folding described in both the 2D and 3D cases as developed requires sequential folding to proceed to a specified structure. This was implemented for the 2D case with acrylic parts as per figure 8a floating on a 2D air bearing described elsewhere. These parts were programmed with the magnetic poles indicated in figure 3a. They were ejected sequentially from a constraining tube which allowed each to fold before pushing the next pixel in the string out the tube. The folding of the letter ‘M’ is shown in figures 8b)i-iv. The letters M, I, & T, as folded, are shown in figure 8c. Supplementary video 1 demonstrates the process.

A perforated acrylic plate had pressurized air pass through the perforations in the manner of an ‘air hockey table’. Parts placed on this surface floated somewhat frictionlessly and were unconstrained. The linear sequence for the letters constructed was loaded into a plastic channel on this surface and the sequences were pushed through the channel by hand with an attempt to ensure that the previous tile (and folded chain before it) had folded into position before the next part was pushed through. All parts were connected by passive tether pieces that kept them in the string and acted as hinges for the folds. NdFeB magnets (amazinemagnets.com), 1x3mm were placed with polarities defined in figure 3 on the 4 sides of each ‘pixel’. The parts were laser cut from 3mm acrylic sheet (mcmaster carr) and magnets were press fitted into flexure-based clamps on each face. The folding experiments were recorded with a SONY

Figure 8. (a-b) Laser-cut acrylic parts used for physical implementation of the folding construction scheme. Tiles are connected by the connector piece illustrated, which allows plane filling packing. Magnets that determine the folding direction are held in the magnet holder flexure. (c-f) Screenshots of video of the folding process for the letter M. The tiles are ejected sequentially from the constraining tube at left. The tile sequence determines the final structure exactly. (g) Screenshot of M.I.T. folded from 3 different string sequences.
digital video camera and still frames were taken from the video with Apple Imovie.

Figure 9. Radiusing of edges about 2 axes allows a structure that neither self-intersects, nor self-interferes during folding.

Results and Discussion.

In summary, the above constructions prove that at a resolution cost of $4n$ in 2d and $6n$ in 3d, any space-filling structure can be built of a string of connected geometric primitives. This structure can be folded without intersecting itself by sequential folds from one end of the string. The important point is that the primitive components of these constructions are simple to make, either through macroscale printing processes, MEMS, or potentially even chemically or biologically. This implies an interesting potential technique for building complex micro- and nano-structures. This also implies the importance of molecules such as chaperonin in protein folding, in that they might limit the folding order to a sequence that gives deterministic structure rather than global instantaneous folding. This might therefore also be an interesting model to assist in protein design and in understanding how complex structure is formed in nature. Furthermore, this implies a route to micro-robotics where a single string with simple binary actuators could fold from any one configuration to any other in order to grasp an object or for locomotion.

While the construction presented creates only pure geometry, it is of course true in biological systems such as protein folding that it is function that is important. The folding system as described here could have functionality superimposed on the different pixels or voxels and the sequencing would allow the positioning of those functional components at any desired location in the global 3D structure. This naturally would increase the number of component types.

Although the construction shown here indicates no problem with self-intersection (the folding of the string through the spatial position of previously folded components), it says nothing of self-interference (the interference of corners of the component when sweeping about an edge fold). To fold into corners, the pure tetrahedra of figure 4a. will interfere and require compliance of the global structure to allow them to settle to the desired structure. The radiused tetrahedra of Figure 9 allow all folds with neither self-intersection nor self-interference.

Finally, we note that in a related piece of work the author, building on the work of Penrose$^{14}$, has implemented state machines capable of autonomously replicating strings of components similar to those presented here$^{15}$. Combing both pieces of work represents the beginning steps in synthetic, self-replicating, 3 dimensional systems.

Acknowledgements

Thanks to Erik and Martin Demaine of MIT for constructive discussions towards this solution.

---
