Inverse Design Tool for Asymmetrical Self-Rising Surfaces with Color Texture

Jianzhe Gu
Carnegie Mellon University
jianzheg@andrew.cmu.edu

Vidya Narayanan
Carnegie Mellon University
vidyan@cmu.edu

Guanyun Wang
Carnegie Mellon University
guanyun@zju.edu.cn

Danli Luo
Carnegie Mellon University
danlil@andrew.cmu.edu

Harshika Jain
Carnegie Mellon University
harshikj@alumni.cmu.edu

Kexin Lu
Carnegie Mellon University
kexinl@andrew.cmu.edu

Fang Qin
Carnegie Mellon University
fangq@stanford.edu

Sijia Wang
Carnegie Mellon University
sijiaw2@andrew.cmu.edu

James McCann
Carnegie Mellon University
jmccann@cs.cmu.edu

Lining Yao
Carnegie Mellon University
liningy@cs.cmu.edu

ABSTRACT

4D printing encodes self-actuating deformation during the printing process, such that objects can be fabricated flat and then transformed into target 3D shapes. While many flattening algorithms have been introduced for 4D printing, a general method customized for FDM (Fused-Deposition Modeling) printing method is lacking. In this work, we vary both the printing direction and local layer thickness; and extend the shape space to continuous-height-field surfaces without the requirement of symmetry. We introduce an end-to-end tool that enables an initially flat sheet to self-transform into the input height field. The tool first flattens the height field into a 2D layout with stress information using a geometry-based optimization algorithm, then computes printing tool paths with a path planning algorithm. Although FDM printing is the fabrication method in this work, our approach can be applied to most extrusion-based printing methods in theory. The results exemplify how the tool broadens the capabilities of 4D printing with an expanded shape space, a low-cost but precise coloring technique, and an intuitive design process.

CCS CONCEPTS

• **Computing methodologies** → Interactive simulation; *Mesh geometry models*; *Physical simulation*; • **Human-centered computing** → Graphical user interfaces; • **Applied computing** → **Computer-aided manufacturing**.

SCF '20, November 5–6, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8170-3/20/11.
https://doi.org/10.1145/3424630.3425420

KEYWORDS

4D printing, shape-changing interface, inverse design, computer-aided design, computational fabrication, geometry optimization, 3D printing

ACM Reference Format:
Jianzhe Gu, Vidya Narayanan, Guanyun Wang, Danli Luo, Harshika Jain, Kexin Lu, Fang Qin, Sijia Wang, James McCann, and Lining Yao. 2020. Inverse Design Tool for Asymmetrical Self-Rising Surfaces with Color Texture. In *Symposium on Computational Fabrication (SCF '20), November 5–6, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3424630.3425420

1 INTRODUCTION

4D printing emerged as a technique to automate the assembling process, with which objects can be printed flat and then transformed into a target 3D shape [Tibbits 2014]. It minimizes materials and shipping costs, speeds up fabrication, and saves the effort of manual assembling [Mitchell et al. 2018].

Among different additive manufacturing approaches, an FDM-based printing method is a popular option for 4D printing with unique advantages in safety, cost, and controllability [van Manen et al. 2017]. FDM printers extrude melted thermoplastic through a narrow nozzle, which stretches the material along the printing direction. The pre-stretch causes the material to shrink along the printing direction under heat. In addition to the shrinkage direction, the amount of shrinkage can be controlled through the printing thickness of each layer. A previous work Geodesy [Gu et al. 2019] has explored 4D printing of various 2.5D surfaces by changing the printing direction and layer thickness. However, the shape space of Geodesy was limited to axial symmetric shapes. One of the main challenges in this problem is the strict requirements for extrusion-based printing tool paths. Specifically, to ensure the printing quality, tool paths should be continuous, equal-spacing, and have very few sharp turns or corners. Additionally, the problem is highly non-linear and has a large parameter space including the printing

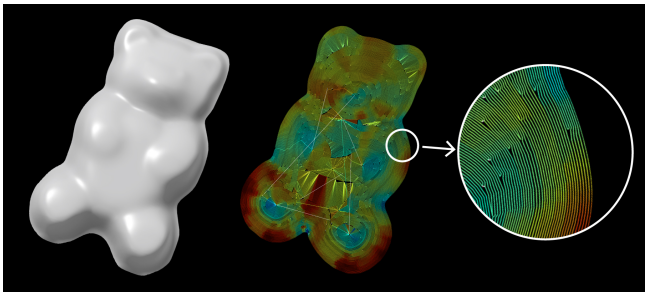

Figure 1: Inverse design of a textured Teddy bear via 4D printing. Left: flattened screen printing texture pattern. Right: flattened 3D printing tool paths. The color shows the expected local shrinkage ratio of the material during transformation. The regions with redder colors have larger shrinkage and smaller layer thickness along the tool paths.

direction and layer thickness at every location on the plane, which makes it extremely hard to directly search or optimize.

In this work, we try to push the limit of the shape space for thermoplastic-based 4D printing. To our knowledge, converting continuous 3D height fields into flat tool paths for FDM techniques has never been thoroughly considered before, and we focus on solving this problem.

As Figure 1 shows, given an input 3D surface, our inverse design tool generates continuous equal-spacing 2D printing tool paths. In addition to the shrinkage direction, the amount of shrinkage also affects the transformation, which is indicated by the color in Figure 1. Here we adopt layer thickness as the variable to tune the shrinkage ratio. We print 6 layers with the same printing paths to ensure the sheet stiffness. Eventually, the flat sheet is uniformly heated with a hot water bath at 90℃, and self-transform into the target 3D surface.

We introduce an inverse design algorithm that consists of a flattening algorithm and a tool path planner. Given an input height field, our tool computes a 2D layout of the triangle mesh with geometry-based optimization and a vertices relocation process. Inspired by the tracing procedure of automatic machine knitting[Narayanan et al. 2018], we developed a path planning method that computes continuous tool paths with varying layer thickness from the 2D layout. The algorithm finally generates the G-code for FDM printers.

In addition, this work enables the 4D printed object to have color texture or conductive circuits through screen printing. Printing full-color 3D objects or applying conductive traces usually requires a specialized setup [Baudisch et al. 2017; Zhang et al. 2015]. However, in the case of 4D printing, screen printing, which was only for 2D painting, becomes highly accessible and reliable.

We start by introducing the material mechanism and the description of our problem, then detail the user workflow and algorithms and end with a set of demo applications and performance evaluations. We summarize our contributions as follows:

- An inverse design tool: we build a design tool that computes printing tool paths and corresponding color textures in 2D, given a target colored 3D surface. A flat sheet is printed with an FDM printer, colored by screen printing, and can self-transform into a target 3D shape with heat.
- Demo artifacts: we demonstrate the use cases for our tool with various applications including touch-sensitive information display, iterative design, rapid prototyping, and texture changing facial masks.

Although it is focused on FDM printers, our algorithm might theoretically be extensible to most 4D printing systems that rely on the extrusion to tune the material anisotropy.

2 RELATED WORK

2.1 2D-to-3D Shape Changing Mechanism

Researchers have been investigating various mechanisms that enable rapid shape change from initially flat sheets to 3D shapes. These mechanisms have different fabrication processes, triggering processes, and the resulting shapes.

Pressurized air can instantaneously trigger shape transformation and hence has been widely used as a shape-changing mechanism. Researchers successfully fabricated 3D surfaces with inflatable structures [Skouras et al. 2014], cut-and-sew method [Mori and Igarashi 2007], and non-stretchable airbags [Ou et al. 2016]. Auxetic structures can transform under air pressure, gravity, or manual activation [Konaković-Luković et al. 2018; Wang et al. 2018a]. The in-plane strain of pneumatic channels can be turned into out-of-plane shape change [Siéfert et al. 2019]. However, these methods require active air supply to sustain the shape.

Buckling of kinematic grid structures is a highly automated approach to transform 2D shapes into doubly curved 2.5D shapes, yet it still requires manual control on corner angles [Pillwein et al. 2020].

The inherent elastic energy of automatic shape changing systems is another widely explored mechanism. The pre-stretch inside fabric or elastic sheets has been used to approximate 3D shapes [Guseinov et al. 2020, 2017; Perez et al. 2017]. These methods can approximate a large set of geometries, but often requires a complicated fabrication process, such as uniform pre-stretching and binding between different materials.

Responsive materials can self-transform under specific stimuli. The shape change can be programmed by locally tuning the concentration, pre-strain, or anisotropy of the materials. Nematic-to-isotropic phase transition in liquid crystal elastomers (LCEs) can be programmed on pixel level by magnetic field alignment [Aharoni et al. 2018; Spillmann et al. 2006]. The tendency of biomaterials, such as hydrogels, to absorb water and expand has also been used to enable shape transformation [Kim et al. 2012; Lee and Konst 2014; Nojoomi et al. 2018]. However, the materials are often costly or require a laborious routine to prepare.

Extrusion based 4D printing utilizes the intrinsic responsiveness of materials for shape change. The printing method itself can shear-align the microscopic polymer chains or fibers inside the material and program local anisotropic deformation. It can align ferromagnetic particles [Kim et al. 2018], microfibers mixed in polymers and hydrogels [Ge et al. 2013; Gladman et al. 2016], and Cocoon-shaped Natto cells [Yao et al. 2015]. However, these methods need specially-made materials that are accessible to hobbyist users.

Shape-memory thermoplastic is a low-cost and easy-to-get material. It shrinks along the printing direction when the residual strain of polymer chains is released [van Manen et al. 2017]. Researchers have constructed flat sheets that can self-fold into origami-like geometries [An et al. 2018a,b]. To extend the shape space, researchers have found that circular printing paths can transform disks into doubly curved surfaces [Gu et al. 2019], but the inverse design is restricted to the surface of revolutions. This work is intended to further extend the shape space of extrusion-based thermoplastic 4D printing with an inverse design algorithm. However, our algorithm can be applied to extrusion-based methods in general.

2.2 Printing Tool Path

FDM printing is a cheap and accessible fabrication method. However, it has particular limitations on the tool path. A high-quality tool path is expected to be continuous, equally spaced, and free of sharp turns [Ding et al. 2014; Jin et al. 2014]. During printing, the filament, which is usually a thermoplastic material, is melted and extruded from a nozzle. The molten material is laid onto the printing platform and quickly solidified. Disconnection in the tool path forces the nozzle to stop extrusion, travel to another location, and restart extrusion. Because of the rheology of molten material, a sudden stop or start drastically impairs the quality of the print. There is a dead zone from the valve to the nozzle tip, where material will keep extruding even when the valve is shut. This undesired oozing will contaminate the existing print when the nozzle is traveling. When the print restarts, the dead zone in the nozzle will leave holes where the material should be extruded. These defects are hard to predict or avoid once disconnection in the tool path is encountered. Furthermore, sharp turns that occur due to unavoidable acceleration and deceleration of the nozzle will affect the printing quality and precision as well, resulting in inconsistent extrusion speed and undesired density change of the print paths.

Several approaches have been introduced to alleviate this problem. Continuous curves can be evolved into maze or labyrinth patterns that fill a 2D region with fewer disconnections [Pedersen and Singh 2006]. Space-filling zig-zag curves [Gibson et al. 2014] have been widely adopted as tool paths for FDM printers. Fermat spirals can fill a 2D region with a single path with very few sharp turns only in the centers of the spirals [Zhao et al. 2016].

However, none of these methods consider the direction of the printer tool path. In this paper, we will take into account that the material will shrink along the printing direction in addition to the inherent challenges of FDM tool path planning.

2.3 Flattening Algorithm

Flattening a 3D surface is a common topic in computer graphics, used for surface parameterization and texture mapping [Mullen et al. 2008; Nehari 2012; Sawhney and Crane 2017]. However, these methods are detached from physical systems and have no constraints on the fabrication.

As mentioned before, researchers in computational fabrication introduced different methods to find a flat configuration that can transform into 3D shapes [Guseinov et al. 2017; Konaković et al. 2016; Pillwein et al. 2020]. These methods are constrained by the physical limits of the system, but none of them need to handle printing tool paths.

Several works in extrusion-based 4D printing develop algorithms to approximate arbitrary surfaces, but their printing tool paths are built on top of a simplified grid-like surface [Boley et al. 2019; Wang et al. 2018b].

The in-plane strain of pneumatic channels is used to transform a sheet into arbitrary surface [Siéfert et al. 2019]. However, the channels are allowed to change their width to fill the surface, whereas printing tool paths usually have a consistent width and need to be equispaced. Nematic alignment in LCE can be tuned by a magnetic field to transform the LCE sheet, but the magnetic field programs the alignment pixel by pixel and does not require continuity.

2.4 Painting on 3D Surface

In terms of 3D color printing techniques, various approaches have been used to color 3D printed structures, from multi-color 3D printers to computational hydrographics[Zhang et al. 2015] and vacuum forming [Schüller et al. 2016]. But these approaches usually require professional equipment. For incorporating touch input and sensing, researchers have developed a technique based on electric field tomography [Zhang et al. 2017] by applying uniform conductive spray on the 3D surface. However, they have to go through a separate surface coating process to color the object. In our work, the pre-transformation flat prints are particularly compatible with commercial and well-established approaches such as screen printing. Conductive screen printing can be directly employed for creating touch-sensitive surfaces.

3 BACKGROUND

Leveraging layer thickness to control local shrinkage has been introduced previously [Gu et al. 2019]. For people not familiar with 4D printing, we will introduce the basic mechanism behind the prior work and this work.

The fundamental mechanism behind thermoplastics-based 4D printing is the directional shrinkage effect of PLA (Polylactic acid, a type of thermoplastic for 3D printing). During printing, PLA is melted and extruded through a small nozzle onto the printing platform(Fig. 2a). In this process, the polymer chains in PLA are elongated. As the PLA is cooled down and frozen, the polymer chains are locked in the elongated status. When the printed path is reheated and melted again, it shrinks along the printing direction and retains its stress-free length (Fig. 2b). Based on our quantification tests(Fig. 2c,d), the shrinkage ratio is related to the layer thickness. The thinner the path is, the more PLA is elongated, and the more the path will shrink after reheating.

This local shrinkage can cause a global out-of-plane shape change. For example, in Figure 3a, if PLA is printed into a round disk with parallel concentric circle paths, the disk only shrinks along the azimuth direction L_1 and remains the same in the radius direction, which causes the center area to pop up. Varying the layer thickness along the radius results in various axial symmetric shapes (Fig. 3b).

The prior work [Gu et al. 2019] showed a variety of axial symmetric surfaces by only changing the layer thickness between different circular paths. In this work, we further extend the shape space by

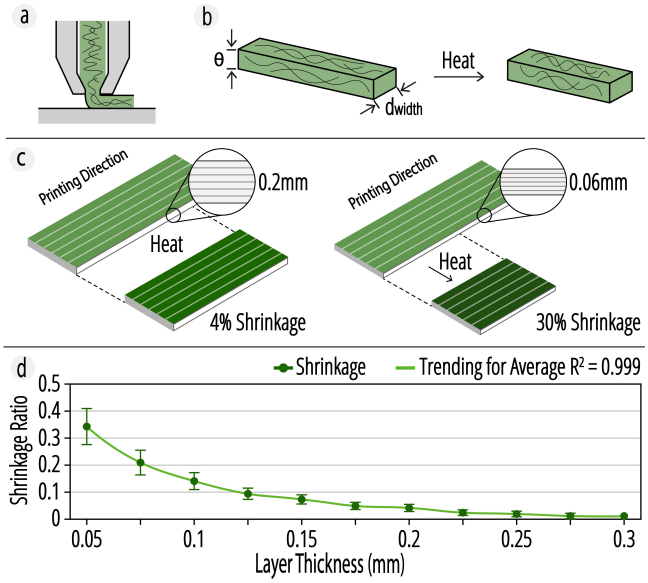

Figure 2: Layer thickness and shrinkage ratio. (a) Demonstration of the extruding process in 3D printing. (b) Shrinkage of single printing path, where θ is the layer thickness, d_{width} is the width of the printing path. (c) Illustration of measuring the shrinkage ratio of rectangle pieces with 6 parallel paths and 6 layers. (d) Mapping between layer thickness and shrinkage ratio.

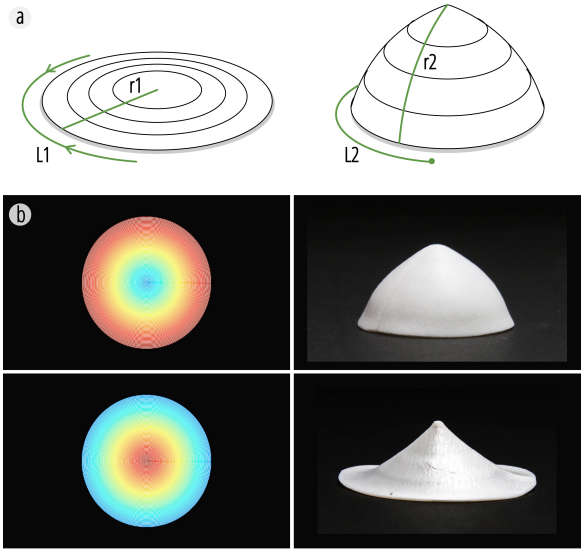

Figure 3: 4D printing Mechanism. A (a) disk printed with concentric circular paths will transform into a surface of revolution after heating (r1 = r2 and L1 > L2). (b) Disks with varying distribution of layer thickness transform into different surface of revolutions. (Red: thinner layer and larger shrinkage ratio, Blue: thicker layer and smaller shrinkage ratio

varying the **printing direction** and varying the **shrinkage ratio within printing tool paths**.

4 USER WORKFLOW

In this section, we describe the user workflow with our tool and its key functions of inverse design, while leaving the implementation details for section 5, 6. Specifically, we need to go through the following steps to create a self-transforming height field.

(1) Create a surface mesh generated from a height field with color texture and importing it into the tool (Fig. 4a).
(2) Use the "meshing" function to let the tool check the validity of the surface first and report errors if there are any invalid geometrical features, such as holes or depressing areas (elaborated in the later section). It then meshes the input surface into a triangle mesh (Fig. 4b). By hiding the edges that represent the lateral printing direction, we render the triangle mesh as a stress field in 3D. The color of each edge indicates the stress of the material along the edge.
(3) After the meshing is complete without errors, use the "flatten" function to let the tool automatically flatten the 3D triangle wireframe (Fig. 4c d) based on an optimization pipeline detailed in the next section. The animation on the screen shows that both the shape and the stress of the triangle wireframe update at each timestep, and eventually the wireframe converges onto a 2D plane with color indicating the required shrinkage ratio along the edge direction.
(4) We can switch to the "forward simulation" mode to visualize the simulated transformed shape(Fig. 4f) computed from the flattening result to validate the shape.
(5) Export the file. Based on the stress map generated in step 3, the tool generates the 2D printing tool path in G-code together with a 2D color texture (Fig. 4e, g).
(6) 3D print the flat sheet with the generated G-code according to the tool path (Fig. 4e).
(7) Screen print a colorful ink onto the sheet with the generated color texture pattern (Fig. 4g).
(8) Use hot water at 80℃ to trigger the transformation and get the final 3D surface with color (Fig. 4h).

5 PROBLEM STATEMENT

Our goal is to compute the printing tool path of a flat sheet that can transform into a target surface with a texture defined as vertex colors. To simplify the problem, the target surface is restricted to disk topology.

There are several criteria for 3D printing tool paths. First, each pair of adjacent printing paths should be equally spaced. Nonparallel or intersecting paths will cause gaps or overlaps that sacrifices the printing quality and impair the directional shrinkage. Second, because molten PLA tends to flow, it is not recommended to stop and restart extrusion once the printing has begun. Based on these constraints, the problem can be reformed as follows:

Given a height field $\mathcal{H}$, we want to find an ordered list of tool path traces $T = \{t_0, t_1, ...t_n\}$. Each trace t_i is an ordered sequence of directed segments $t_i = \{s_{i,0}, s_{i,1}, ...s_{i,n}\}$, where each segment $s_{i,j}$ is

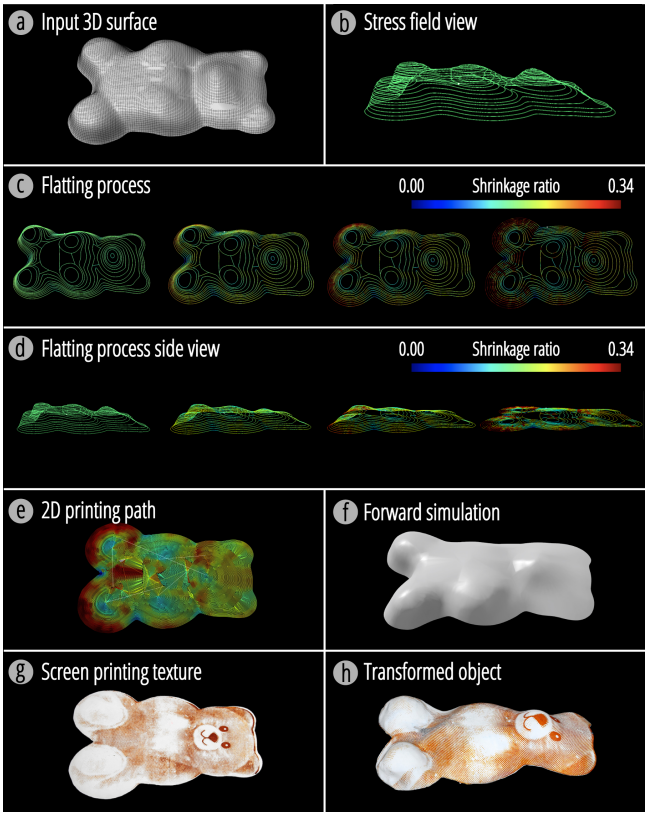

Figure 4: User Workflow.

associated with the shrinkage ratio $\sigma_{i,j}$. A flat surface composed of printing tool paths, which are extruded along T with the required local shrinkage ratio $\sigma_{i,j}$, will morph into the target height field after shrinkage.

To generate a valid G-code for 3D printing, we need to convert the shrinkage ratio $\sigma_{i,j}$ to the corresponding layer thickness $\theta_{i,j}$, according to the mapping quantified in Figure 2 (c, d), and compute a sequence of nozzle positions and associated printing parameters. Finally, we need to generate the G-code for 3D printers.

6 INVERSE DESIGN METHOD

Observation. Imagine that we color the dome in Figure 3b with equidistant black dots, and tape plenty of non-stretchable thin strips along the radius. To find its flattened layout, we can heat up and soften the dome, and uniformly press it against the plane and spread wrinkles if there are any. The dots will move farther apart along the azimuthal direction, which shows the amount of expansion during the flattening process. On the other hand, the expansion ratio indicates the required shrinkage ratio that can turn the disk to the dome. We built an inverse method inspired by this observation.

The abovementioned inverse method requires the assumption of the printing direction, which is circular paths in the dome case. Theoretically, the method can flatten any path rendered on the surface, but a wrongly chosen path may require excessively large shrinkage. Therefore, we need to find a surface-filling path that can efficiently utilize the limited shrinkage ratio of PLA (0% to 34%).

We observe that a circular pattern can effectively push the center area upwards, which is suitable for a 2.5D surface. We, therefore, choose the direction along the isolines of the target height field as guidance for the direction of the printing tool path. Exploring more efficient tool paths would be a good future work.

Method Overview. Our system takes as input a 2.5D height field with an optional texture map for color. The height field is first meshed into a quadrangle-dominated graph and then converted into a triangle mesh. Next, a 2D layout of the triangle mesh is computed through a geometry-based constraint optimization and a vertices relocation process in an iterative fashion. After that, we construct a tool path with appropriate shrinkage ratios by tracing the 2D mesh. Lastly, color and opacity values are computed from the texture map and local shrinkage factors.

Material characterization. To control the amount of shrinkage, we vary the *thickness* of the printing layer on-the-fly. Compared to previous work [Gu et al. 2019], we report a set of printing parameters that result in a larger range of shrinkage ratio. For the primitive tests, we used a nozzle of 0.5 mm in diameter, a PLA filament of 2.88 mm in diameter, and printed at 3000 mm/s. The resultant shrinkage ranges from 0.0% to 34.3%. We also tested 11 layer thickness values with 33 rectangles. These rectangles all have a length of 55 mm, a width of 20 mm, and 6 layers. We ran a 4th-order polynomial curve fitting on the collected data and got a mapping $\theta \sim \sigma$, where θ is the layer thickness and σ is the shrinkage ratio (Fig. 2).

6.1 Meshing

Graph generation. Our tool first generates a quadrangle-dominated graph over the target height field. Using the height function of the target surface, we first extract equidistant isolines, and then generate gradient lines by sampling points from each isoline and iteratively extend a line along both positive and negative gradient directions [Dong et al. 2005]. Intersection points between isolines and gradients lines form the vertices of the graph with edges connecting adjacent vertices either along isolines or gradients lines (Fig. 5a). In a regular case, each vertex in the graph is connected to its four neighbors, on its left, right, top, and bottom, respectively. The graph is dominated by quadrangles, except for three cases – at each (1) saddle or (2) cone position of the function, there are two (or more) isolines along the height direction and this appears as a polygonal region in the graph. (3) As the length of isolines gets smaller with increasing height, the number of quads decreases, thus causing some quads to merge into one quad by sharing the upper edge as a part of the lower edge of the merged quad.

Triangulation. Next, our system upsamples the graph and converts it into a triangle mesh, on which we set up the geometric constraints. Each quadrilateral in the graph is upsampled by splitting the edge that lies along the isoline such that each edge has a maximum length l_u (Fig. 5b). This discretization factor l_u can be tuned for smoothness and simulation stability. We use a value of 0.25mm for all our examples. The newly introduced vertices are linked in a greedy, iterative fashion by introducing an edge with the shortest distance to triangulate the quad as shown in Figure 5c.

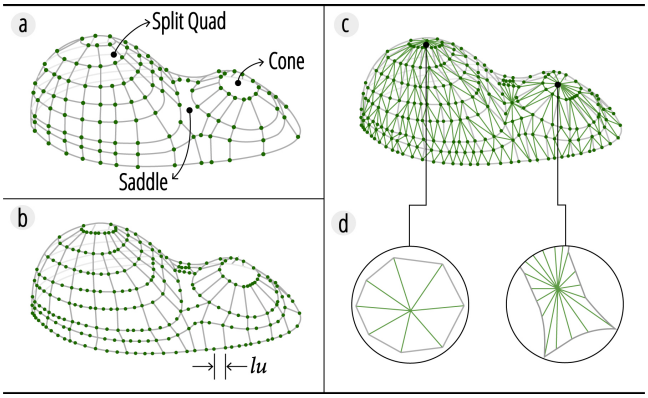

Figure 5: Meshing. (a) Graph generation. The graph is generated by extracting isolines and gradient lines from a height function and taking their intersections as vertices. (b) Upsampling. The mesh is upsampled by inserting equidistant vertices on printing edges. (c) Triangulation. The quadrangles in the mesh are triangulated by a greedy method. (d) Cone and Saddle areas are triangulated with fan triangulation, by inserting a vertex at the centroid position and connect it with vertices on the edges.

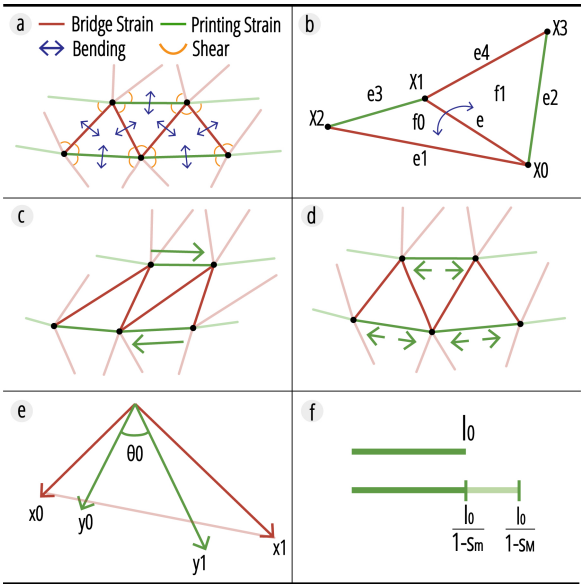

Figure 6: Flattening. (a) Soft constraints on a region of the mesh, where printing strain constraints are applied on printing edges along isoline direction and bridge strain constraints are applied on bridge edges along gradient line directions. (b) Bending constraint is set on interior edges shared by pairs of faces. (c, e) Printing edges shift when there is no shear constraint. Shear constraint is set on the inner corners of triangles. (d, f) The bridge strain constraints are set on bridge edges which tend to maintain their length. The printing strain constraints are set on printing edges, which will expand to a length within a preset range whose upper and lower limits represent the maximum and minimum shrinkage ratio.

For the polygonal region around saddles and extrema, a new vertex is introduced in the center of the polygon (Fig. 5 d). The polygon is triangulated by joining all vertices to this central vertex. The triangle mesh is represented as $M = \{V, E_p, E_b, F\}$, where V is the set of all the vertices in the graph. E_p is the set of all the edges along the isolines representing the printing direction which is called *printing edge* in this paper. E_b is the set of all the rest of edges that bridge the vertices between consecutive isolines and F is the set of triangles, which is called *bridge edge* in this paper.

6.2 Mesh Flattening

The tool flattens the triangle mesh iteratively. Each iteration includes two steps. The first step is a constraint optimization with a loss function factored into 4 geometric constraints. These soft constraints approximate the mechanical properties of the sheet under heat triggering, except that the tendency to shrink is modeled as the tendency to expand in order to inversely compute the shrinkage ratio. In the second step, we move the vertices following 3 relocation rules to mimic the external forces that flatten the mesh.

6.2.1 Geometric Constraint Optimization. 4 key mechanical properties of the material determine the shape change process and result. (1) Elastic modulus. Thermoplastic PLA material has a tendency to withstand stretching or compression. Particularly, as mentioned in the **Background** section, the sheet tends to preserve its length along directions perpendicular to the printing direction. (2) The residual strain upon elastic deformation. In addition to the elastic modulus, there is a residual strain that is formed when the material is extruded and stretched along the printing direction with rapid cooling. Once the printed sheet is triggered with heat above T_g, the residual strain will be released and cause the sheet to shrink along the print path. (3) Bending stiffness. Although the PLA sheet is softened under heat triggering, it still has elasticity and bending stiffness that prevents it from getting folded or wrinkled. (4) Shear stiffness. This property makes the sheet resistant to shear deformation, preventing a rectangle from deforming into a parallelogram without changing the lengths of sides. To capture these properties, we introduce 4 constraints respectively: *Bridge Strain Constraint*, *Printing Strain Constraint*, *Bending Constraint*, and *Shear Constraint*.

Printing Strain Constraint. As mentioned in the **Background** section, the material shrinks by $0 \sim 33\%$ after shape transformation. For printed material along the printing direction, it shows tensile stiffness with residual strain. As an inverse process, we start with the target mesh which is the user input 3D mesh with all the *printing edges* already in their final form (Fig. 6a). We try to find its initial length, the length before shrinking when the sheet is flat, by setting the target length of the edge between the largest possible initial length, in which case the edge shrinks by maximum shrinkage ratio, and the smallest possible initial length (Fig. 6d,f). Specifically, the length of each *printing edge* at each iteration should meet the

following constraint. For each $e_p \in E_p$,

$$\frac{l_0}{1-s_m} \leq l \leq \frac{l_0}{1-s_M},$$

where l_0 is the length of e_p on the target mesh, l is the length of e_p at the current iteration, s_M is the maximum shrinkage ratio, and s_m is the smallest shrinkage ratio. The energy of the *printing edges* is as follow,

$$E_{printing} = \sum_{e_p \in E_p} max(0, \frac{l_0}{1-s_M} - l)) + max(0, l - \frac{l_0}{1-s_m}).$$

In contrast to the triggering process when the *printing edges* start with the length on the flat mesh and target on shrinking by a certain amount, in the inverse design process, *printing edges* tend to elongate (Fig. 6f) and the mesh will expand and flatten.

Bridge Strain Constraint. As described in the **Background** section, expansion along the lateral direction is minor and is ignored in our model. Therefore, each $e_b \in E_b$ tends to maintain its length during triggering without pre-strain (Fig. 6a,d). The bridge strain constraint is introduced to represent the physical bonding between the material on a parallel tool path. We are trying to minimize the difference between the current edge length and the target edge length by applying the bridge strain energy as the form

$$E_{bridge} = \sum_{e_b \in E_p} ||l - l_0||_2^2,$$

where l_0 is the length of e_p at the target mesh, l is the length of e_p on the mesh at the current iteration.

Bending Constraint. The sheet has bending stiffness during transformation, which makes it hard to fold. We approximate this stiffness by minimizing the bending energy of the mesh. Specifically, the bending energy is set on each interior edge $e \in E_p \cup E_b$ that is shared by two faces f_0, f_1 (Fig. 6b). We apply bending energy of the form

$$E_{bending} = \sum_{\substack{e \in E_p \cup E_b \\ e \notin boundary}} \frac{1}{2}(x_0, x_1, x_2, x_3) Q(e) (x_0, x_1, x_2, x_3)^T,$$

where $Q(e) = \frac{3}{A(f_0), A(f_1)} K_0^T K_0$, $A(f_i)$ is the area of triangle f_i. $K_0 = c_{03}+c_{04}, c_{01}+c_{02}, -c_{01}-c_{03}, -c_{02}-c_{04}$, where $c_{jk} = cot \angle e_j, e_k$.

Shear Constraint. The shear constraint is introduced to approximate the shear stiffness. Specifically, it prevents two paths from shifting in opposite directions by restricting the in-plane angle change of corners in each triangle (Fig. 6c, e). For every corner in each triangle, $\overrightarrow{x_0}$ and $\overrightarrow{x_1}$ are the vectors of two neighboring edges, and θ_0 is the angle of the corner on the target surface. We construct $\overrightarrow{y_0}$ and $\overrightarrow{y_1}$ as the angle between them equals to θ_0. Additionally, they have the same length and the same angle difference between them and $\overrightarrow{x_0}$, $\overrightarrow{x_1}$ respectively. We are trying to minimize the following energy

$$\sum_{\alpha \in A} ||y_0 - x_0||_2^2 + ||y_1 - x_1||_2^2,$$

where α is the inner corner of a triangle, A is the set of all inner corners on the mesh.

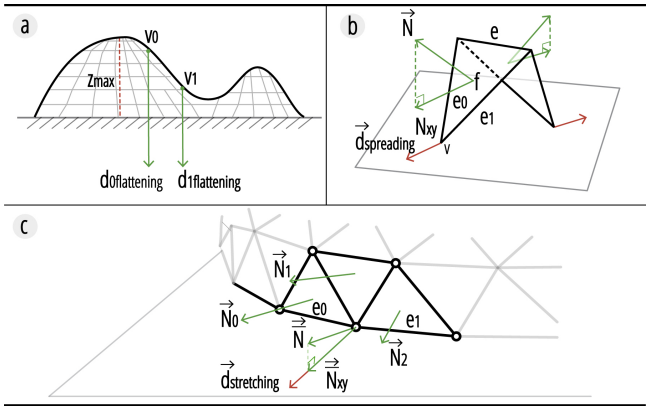

Figure 7: Vertex Relocation. (a) Flattening relocation is set on each vertex pointing downwards to press the mesh against the plane. (b) Stretching Relocation is set on boundary vertices to stretch the boundary outwards and enlarge the projection area of the mesh. (c) Spreading Relocation is set on a pair of faces sharing an edge to wipe out crumples and avoid folding.

6.2.2 Relocating Vertices. After optimizing the mesh, we introduce a vertex relocation process. This process mimics the flattening of a sheet (without crumpling) under external forces. Imagine flattening a rubber balloon by pushing it downwards over a surface. As the surface area of the balloon is larger than the projected area, simply pressing downwards will cause wrinkles to appear on the balloon. To remove wrinkles, we may stretch it from the boundary to enlarge the projected area. We may still introduce some wrinkles in the middle, and pressing might cause some area to fold. Hence, the last step is to wipe out the wrinkles appearing in the center. To mimic these three processes, we introduce a strategy to iteratively update vertex positions following the three relocation processes, *Flattening*, *stretching* and *spreading*.

Flattening Relocation. As figure 7a shows, at the end of each time step, our tool relocates each vertex towards the plane by assigning a displacement along the z-axis by the following amount,

$$d_{flattening} = -\frac{v_z}{v_{z_{max}} \sqrt{l_x l_y}},$$

where v_z is the z-axis value of the vertex, $v_{z_{max}}$ is the maximum z-axis value on the mesh, l_x and l_y are the extents of the mesh at x and y direction. The displacement is normalized according to the projected extent of the surface and the maximum distance from the surface to the plane. This value decreases as the distance gets smaller, which assures the mesh staying on the plane without fluctuating.

Stretching Relocation. Stretching relocation is introduced to stretch the boundary of the mesh, increase the mesh projection area and flatten the mesh (Fig. 7b). For each vertex v on the *boundary*, f_i is an incident face of v, $\overrightarrow{N_i}$ is the unit normal of f_i, and A_i the area of f_i. The stretching relocation displacement is normalized by the

average length of incident boundary edges of the form

$$\overrightarrow{d_{stretching}} = \frac{|e_0| + |e_1|}{2}\overrightarrow{N_{xy}},$$

where $|e_i|$ is the length of the incident boundary edge e_i, $\overrightarrow{N_{xy}}$ is the projection of averaged normal $\overrightarrow{N}$ on the xy-plane in which $\overrightarrow{N} = \frac{\sum_{i=0}^{n} A_i \overrightarrow{N_i}}{\sum_{i=0}^{n} A_i}$.

Spreading Relocation. We apply the spreading relocation to unfold pairs of triangles against the ground in order to eliminate crumples and prevent folding (Fig. 7c). For each inner edge e shared by triangle f and the other triangles, v is the vertex on f opposite to e, and $\overrightarrow{N}$ is the unit face normal of f. The displacement is normalized with average length of incident edges of the form

$$\overrightarrow{d_{spreading|xy}} = \frac{|e_0| + |e_1|}{2}\overrightarrow{N_{xy}},$$

where $|e_i|$ is the length of e_i, which is an incident edge of v, and $\overrightarrow{N_{xy}}$ is the projection of $\overrightarrow{N}$ on the xy-plane.

After flattening, our system produces an updated 2D triangle mesh $\dot{M} = \{\dot{V}, E_p, E_b, F\}$ where only the position of the vertices are updated to $\dot{V}$, and the topology remains the same. Based on location of vertices before and after flattening, we compute the expansion of each e_p as $1/(1-\sigma)$, and expected shrinkage ratio as σ. This shrinkage ratio of e_p is used to compute the shrinkage ratio of each segment on the printing tool paths in the next section.

6.2.3 Parameters. For the constraint optimization part, we use ShapeOp, a C++ optimization library [Bouaziz et al. 2012] using a projection based method to efficiently compute the equilibrium positions of vertices subjected to physical or geometric constraints. We set coefficients of constraints to $w_{bridge} = 1.0$, $w_{printing} = 1.0$, $w_{bending} = 0.025$, $w_{shear} = 0.25$, set coefficients of relocation to $w_{flattening} = 0.3$, $w_{stretching} = 1.0$, $w_{spreading} = 0.5$, and apply 20 to 50 iterations depending on the mesh size and resolution. The computing time for a mesh with 1.2k triangles is 0.8 seconds on a Macbook Pro with 3.1GHz on a single core.

6.2.4 Forward Simulation. To implement the forward simulation (Fig. 4f) from the flattened layout, we removed the relocations and change the target length of *Printing Strain Constraint* as the product of shrinkage ratio and its current length, σl.

6.3 Tool Path Generation

Now that the mesh is flattened in 2D, our system computes the final toolpath for printing. As explained earlier, 4D FDM printing requires a few tools paths that (1) have an equal distance in between to avoid overlapping and ensure the seamlessness of the sheet and (2) are continuous to avoid retracting the filament. To generate a continuous toolpath, we require that the graph be mostly Eulerian i.e., have an even degree at each node. To do so, we employ an edge doubling strategy [Narayanan et al. 2018]. We first upsample the triangles by interpolating even number of *printing edges* parallel to the original *printing edge* in the triangle, then generate a path by tracing vertices along the *printing edges* following a set of rules.

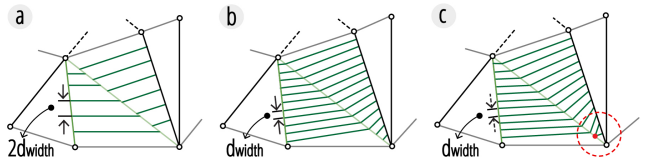

Figure 8: Trace interpolation process.

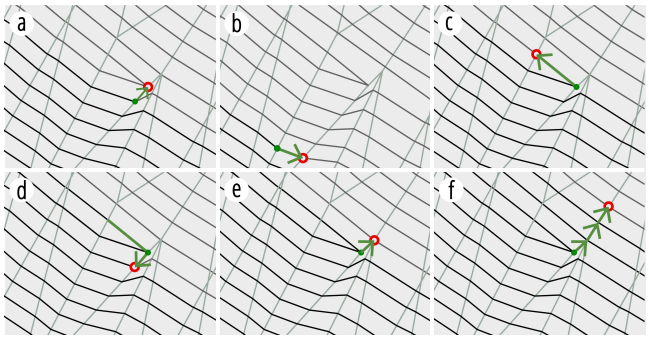

Figure 9: Tool path tracing process.

Printing edge upsampling. For each triangle, we first insert vertices on the two edges, where the segments between them are parallel and have a distance twice as the toolpath width d_{width} to each other (Fig. 8a). For each consequent pair of triangles on the same isoline, we merge the vertices on the edge that shares them, and then double the number of edges by inserting the same edge in between, such that the number of edges in each triangle is always even and the distance equals to the width of printing paths (Fig. 8b). We further move the two neighboring nodes at the end of the segments to the same location and redistribute the vertices on the same edge, such that there is less space between paths (Fig. 8c). Lastly, an isoline value is interpolated and stored in each vertex so that we know where is the rising direction of the height on the target surface. The larger the value, the higher the vertex is.

Tool path tracing. Beginning from a vertex on the boundary, our system follows a set of rules and traces our graph along the filament edges to generate traces. This procedure introduces only a few continuous filament paths $T = \{t_0, t_1, ...t_n\}$, where t_i is each path. Additional paths appear only at each saddle region. The following graphs demonstrate the rules of path tracing. The black lines are the

printing edges. whose two vertices are already visited and stored on the traced path, the grey lines are the

printing edges. to be visited, the green lines are

bridge edges. , the arrow shows the tracing direction, the green dot is the currently visiting vertex and red vertices are the vertices to be visited subsequently in the current step. Each step of tracing follows the rule of the highest order that meets the condition of the current step. The current visiting vertex is represented by v in the following rules.

Rule 1. If an unvisited vertex v_n with a larger isoline value is connected to v by a *bridge edge* and v_n is at a corner, visit v_n (Fig. 9a).

Rule 2. If there is an unvisited vertex v_n connected to v by a *printing edge*, visit v_n (Fig. 9b-c).

Rule 3. If there is an unvisited vertex v_n connected to v by a *bridge edge* and having a smaller isoline value, visit v_n (Fig. 9d).

Rule 4. If v has no unvisited left or right vertex, and there is an unvisited vertex v_n connected to v by a *bridge edge*, visit v_n (Fig. 9e).

Rule 5. If none of the above conditions are met, keep visiting the vertex (upper vertex) with larger isoline value connected by *bridge edge*, until any above condition is met. If there is the upper vertex, end the path (Fig. 9f).

Finally, our system extracts an ordered list T consisting of traces t_i, each trace $t_i = \{s_{0,1}, s_{1,2}, ...s_{n-1,n}\}$ is an ordered list of direction segments of the form $s_{i,j} = \{v_i, v_j, \sigma_{ij}\}$, where σ_{ij} is its shrinkage ratio.

6.4 G-code Generation

A G-code for 3D printing is composed of a sequence of commands, which indicates the segments that the nozzle will move along and their associated printing parameters, such as printing speed and extrusion ratio.

In our case, each command mainly includes the nozzle position at the end of the next move and the volume of material extrusion when the nozzle moving from the current position to the next position. To generate the G-code for the first layer, we first extract the x-y position of each move from v_{i+1} of each s_i. We compute the z position of each move by converting σ_i to layer thickness θ_i according to Fig. 2. We then compute the volume of extrusion ϵ_i based on the length and thickness of the current move s_i. If the nozzle is moving from the end of a trace to the start of another trace, then $\epsilon = 0$. To sum up, for each layer, we convert T into an ordered list of command parameters $C = \{c_0, c_1, ...c_n\}$, where $c_i = \{x_i, y_i, z_i, \epsilon_i\}$. Nozzle z-position $z_i = \theta_i \cdot i_l$, where i_l is the index of the current layer. We stack 6 layers with the same 2D configuration by only changing the i_l, and converts the commands into the G-code.

6.5 Color Mapping

Each vertex on the flattened mesh can be mapped to a point on the triangles of the input mesh. Therefore, the color of each pixel on the texture for the flattened surface can be assigned by interpolating the colors of the three vertices of the face with using its barycentric coordinates.

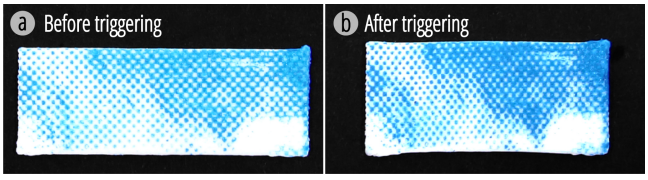

Figure 10: Density of screen printed dots increases as the sheet shrinks, and the color looks slightly darker.

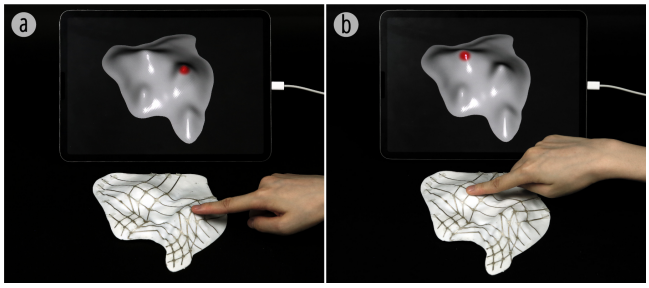

Figure 11: We implement a 3D touch pad by applying conductive screen printing, the touching position is mapped to the surface in the graphics interface.

However, because the flattened surface will be transformed by shrinking, directly applying the position color will appear incorrect after transformation (Fig. 10). To remedy this, based on the length change of the edges, our system computes the expansion area change of the face. The opacity of the color is computed as a_t/a_f, where a_t is the face area on the target surface, a_f is the face area after flattening. This opacity mapping based on area difference provides the intended target color after triggering the surface.

7 RESULTS

7.1 Interactive 2.5D Surface

By screen printing functional inks onto the sheet, we can add additional interactive functionalities. We implemented a capacitive touch-sensing pad on top of a landscape (i.e., a height field map of three Hawaii islands). We were able to apply the color mapping technique introduced earlier to accurately touch-sense on the 2.5D surface (Fig. 11).

A 6 x 6 matrix of conductive traces was printed on top of the height field. A small disk of insulation tape (3M VHB tape cut with a CNC plotter) was applied at each junction to insulate the horizontal and vertical crossovers. 3M VHB tape was chosen experimentally with different insulation glue and spray options, for its resistance to water, heat, and organic solvent inside the silver ink, as well as its elasticity to have the least effect on the 4D transformation. Different interactive modes can be implemented on top of the landscape. Users can draw on the physical surface, and the interpolated traces can be rendered on a virtual model. We also implemented a geographic information display. By touching different spots of the surface, users can gain the coordinates and other geographical information of the chosen locations.

7.2 Facial Texture Changes

By leveraging color mapping and screen printing, we can apply color-changing ink with a designed pattern on the flat surface and transform it into a 3D shape. The color pattern changes with temperature change. We envision this technique may be adapted by hobbyists to make color- and texture-changing facial masks. We chose human-like masks as an example to demonstrate the complexity achieved by our technique. However, other shapes (e.g., animal masks) of masks can be 4D printed as well (Fig. 12).

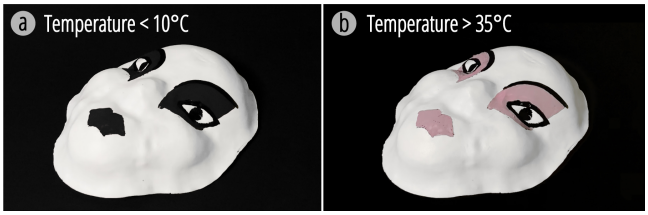

Figure 12: Color changing face.

7.3 Iterative Design and Rapid Prototyping

When prototyping physical artifacts, designers often go through multiple iterations, and each iteration may only have very small modifications. Compared to conventional 3D printing methods to achieve 2.5D height fields, where each iteration will take a similar amount of time, our technique is more rapid, as both quick modification on the virtual model and 4D printing of a flat sheet are time-efficient. Although vacuum forming can potentially produce an exact replica of the mold beneath with thermoplastics, a new 2.5D mold has to be fabricated for each step of an iteration.

We were able to 4D print a collection of 2.5D models. It shows a diverse context, ranging from different landscapes to sea creatures. In the next section, we will further elaborate the shape space and accuracy of our technique (Fig. 13).

8 PERFORMANCE

Flattening quality. PLA has a maximum shrinkage ratio of 34.3% as we tested in our paper. A dramatic shape change often requires larger shrinkage of the material and shows more mismatch after transformation. The colored stress visualization (Fig. 4e) indicates the shrinkage ratio of the edges. To evaluate the quality of the flattening process we report the percentage of edges that fall into the invalid shrinkage range of each demo (Table 1). This happens when the required shrinkage ratio is larger than the maximum shrinkage ratio or is negative (expanding). These invalid values indicate an inadequate shape change and eventually cause a shape mismatch.

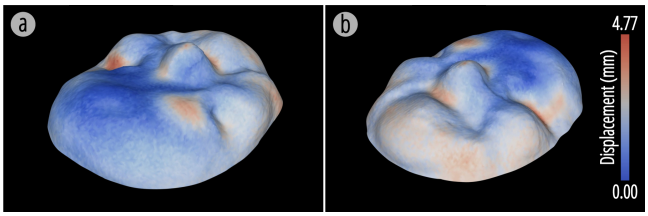

Figure 14: Visualizing shape displacement of a 3D scanned Noh mask from its simulated shape.

Shape accuracy. In order to evaluate the accuracy of the fabrication and transformation, we measured the discrepancy between the five simulated shapes and transformed shapes. As Figure 14 shows, we first 3D scanned the transformed shapes using ScandyPro app with iPhone X, and then quantified the average and maximum point-wise displacement with the iterative closest point (ICP) method. The results are reported in Table 1.

Table 1: We report the information of five models, including longest diam(mm), maximum height(mm), average displacement(mm), maximum displacement(mm), percentage of invalid edges(%), computing time(sec), and 3D printing time(min).

Model	L.D	M.H.	A.E.	M.E.	Inv.	Com.	Prt.
Starfish	175.2	33.4	2.38	5.60	0.4	58.05	92
Hawaii	159.0	34.7	3.61	8.39	3.6	48.52	99
Mask	146.4	42.3	3.70	7.22	4.6	55.05	118
Glacier	181.4	37.5	3.47	6.84	4.7	.53.28	125
Bear	176.8	26.3	2.54	6.13	4.2	55.73	141

Invalid shapes. Due to the limitation of 3D printing and our path planning algorithm, surfaces with extra holes or depressed areas are not feasible. The 3D printed surface is topologically equivalent to a disk. In this case, a boundary is needed in the shape after transformation. Therefore, closed surfaces are not feasible. The generated tool path must be closed loops for the 3D printer to continuously print without retracting and re-extruding material. A hole on the surface will cut off the print path and leave the boundary of the hole as the tool head turns sharply. The accumulated effect of the clustered sharp turns causes undesired bending of the hole boundary. Additionally, the printed sheet is symmetric along the thickness and has no bias towards any direction. When pushing into the water, the resistance of the water usually push the surface in one direction. Therefore, the surface tends to pop in the same direction and an area of depression on the tip is not valid in this work.

9 LIMITATIONS, DISCUSSION AND FUTURE WORK

Material property assumption. We used edge strain constraints to describe the behavior of the material along printing paths. Such a constraint assumes that the material is linearly elastic and does not reach its yield strain. First, based on solid mechanics, elasticity is dominant in thermoplastics when the load frequency is low, and hyper-elasticity is dominant when the frequency is higher. As the triggering time is about 20 seconds, the frequency here is low enough to assume the elasticity and elastic assumptions are reasonable.

The second assumption is that the shape change does not exceed the yield strain. To more accurately characterize this feature, we can potentially investigate other effects, such as Mullins effect [Diani et al. 2009], to describe and understand the shape change behavior.

Fabrication and triggering limits. There are uncertainties in the environment of printing and triggering. The height of the first layer on each and every print is auto-calibrated by the printer, but it could vary between different print jobs. This variation is usually negligible for standard FDM print jobs with reasonable height. However, since our prints are thin sheets, this variation has a significant impact on the performance. A clean calibration mechanism or a sacrificial layer may eliminate this discrepancy.

We also experimented with various triggering temperatures for the same layer thickness. Higher temperatures can cause a larger

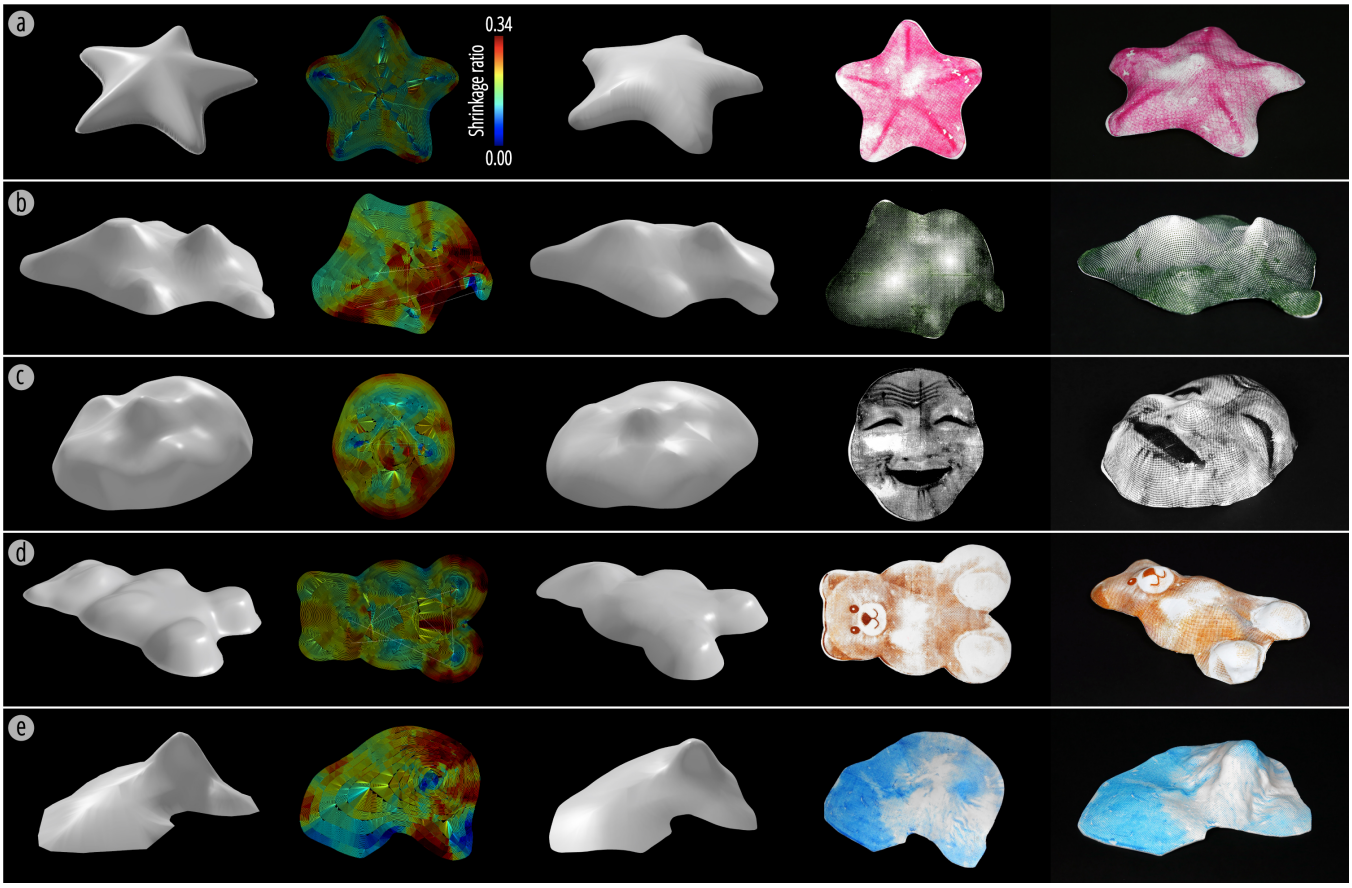

Figure 13: A library of printed objects. (a) Sea star. (b) Island. (c) Face. (d) Bear. (e) Glacier. From left to right, input geometry, printing tool path with stress visualization, simulated transformed shape, 3D printed sheet with screen printing pattern before transformation and after transformation.

shrinkage ratio leading to more dramatic shape change. Calibration of the shrinkage ratio and temperature settings is important for repeatable performance.

Extensibility. Extrusion-based 4D printing shares the same challenges of the printing tool path and has a similar mechanism that is to control the direction deformation to change the global shape. Although this work focuses on the 4D printing of thermoplastics, this inverse design method has the potential to be applied to extensive 4D printing techniques that rely on the extrusion to tune the material's anisotropic properties.

10 CONCLUSION

In this paper, we presented a novel inverse design algorithm to promote the 4D printing of thermoplastics. To our knowledge, this is the first algorithm of its kind to flatten the 3D height field for thermoplastic-based 4D printing. In the future, it can be extended to other materials and mechanisms that rely on continuous deposition to accumulate shrinkage energy. We envision in the future, if we employ a material that has reversible bending behaviors, we can leverage the same method to design and simulate repeatable, robotic, and shape-changing behaviors. We hope the method can help promote and democratize extrusion-based 4D printing in the short term and inspire computational design of shape-changing systems.

REFERENCES

Hillel Aharoni, Yu Xia, Xinyue Zhang, Randall D Kamien, and Shu Yang. 2018. Universal inverse design of surfaces with thin nematic elastomer sheets. *Proceedings of the National Academy of Sciences* 115, 28 (2018), 7206–7211.

Byoungkwon An, Shuhei Miyashita, Aaron Ong, Michael T Tolley, Martin L Demaine, Erik D Demaine, Robert J Wood, and Daniela Rus. 2018a. An End-to-End Approach to Self-Folding Origami Structures. *IEEE Transactions on Robotics* 34, 6 (2018), 1409–1424.

Byoungkwon An, Ye Tao, Jianzhe Gu, Tingyu Cheng, Xiang'Anthony' Chen, Xiaoxiao Zhang, Wei Zhao, Youngwook Do, Shigeo Takahashi, Hsiang-Yun Wu, et al. 2018b. Thermorph: Democratizing 4D printing of self-folding materials and interfaces. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 260.

Patrick Baudisch, Stefanie Mueller, et al. 2017. Personal fabrication. *Foundations and Trends® in Human–Computer Interaction* 10, 3–4 (2017), 165–293.

J William Boley, Wim M van Rees, Charles Lissandrello, Mark N Horenstein, Ryan L Truby, Arda Kotikian, Jennifer A Lewis, and L Mahadevan. 2019. Shape-shifting structured lattices via multimaterial 4D printing. *Proceedings of the National Academy of Sciences* 116, 42 (2019), 20856–20862.

Sofien Bouaziz, Mario Deuss, Yuliy Schwartzburg, Thibaut Weise, and Mark Pauly. 2012. Shape-up: Shaping discrete geometry with projections. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 1657–1667.

Julie Diani, Bruno Fayolle, and Pierre Gilormini. 2009. A review on the Mullins effect. *European Polymer Journal* 45, 3 (2009), 601–612.

Donghong Ding, Zengxi Stephen Pan, Dominic Cuiuri, and Huijun Li. 2014. A tool-path generation strategy for wire and arc additive manufacturing. *The international journal of advanced manufacturing technology* 73, 1-4 (2014), 173–183.

Shen Dong, Scott Kircher, and Michael Garland. 2005. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Computer aided geometric design* 22, 5 (2005), 392–423.

Qi Ge, H Jerry Qi, and Martin L Dunn. 2013. Active materials by four-dimension printing. *Applied Physics Letters* 103, 13 (2013), 131901.

Ian Gibson, David W Rosen, Brent Stucker, et al. 2014. *Additive manufacturing technologies*. Vol. 17. Springer.

A Sydney Gladman, Elisabetta A Matsumoto, Ralph G Nuzzo, L Mahadevan, and Jennifer A Lewis. 2016. Biomimetic 4D printing. *Nature materials* 15, 4 (2016), 413.

Jianzhe Gu, David E Breen, Jenny Hu, Lifeng Zhu, Ye Tao, Tyson Van de Zande, Guanyun Wang, Yongjie Jessica Zhang, and Lining Yao. 2019. Geodesy: Self-rising 2.5 D Tiles by Printing along 2D Geodesic Closed Path. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, 37.

Ruslan Guseinov, Connor McMahan, Jesús Pérez, Chiara Daraio, and Bernd Bickel. 2020. Programming temporal morphing of self-actuated shells. *Nature communications* 11, 1 (2020), 1–7.

Ruslan Guseinov, Eder Miguel, and Bernd Bickel. 2017. CurveUps: Shaping Objects from Flat Plates with Tension-Actuated Curvature. *ACM Transactions on Graphics (SIGGRAPH 2017)* 36, 4 (2017).

Yu-an Jin, Yong He, Jian-zhong Fu, Wen-feng Gan, and Zhi-wei Lin. 2014. Optimization of tool-path generation for material extrusion-based additive manufacturing technology. *Additive manufacturing* 1 (2014), 32–47.

Jungwook Kim, James A Hanna, Myunghwan Byun, Christian D Santangelo, and Ryan C Hayward. 2012. Designing responsive buckled surfaces by halftone gel lithography. *Science* 335, 6073 (2012), 1201–1205.

Yoonho Kim, Hyunwoo Yuk, Ruike Zhao, Shawn A Chester, and Xuanhe Zhao. 2018. Printing ferromagnetic domains for untethered fast-transforming soft materials. *Nature* 558, 7709 (2018), 274–279.

Mina Konaković, Keenan Crane, Bailin Deng, Sofien Bouaziz, Daniel Piker, and Mark Pauly. 2016. Beyond developable: computational design and fabrication with auxetic materials. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 89.

Mina Konaković-Luković, Julian Panetta, Keenan Crane, and Mark Pauly. 2018. Rapid Deployment of Curved Surfaces via Programmable Auxetics. *ACM Trans. Graph.* 37, 4, Article 106 (July 2018), 13 pages. https://doi.org/10.1145/3197517.3201373

Bruce P Lee and Shari Konst. 2014. Novel hydrogel actuator inspired by reversible mussel adhesive protein chemistry. *Advanced Materials* 26, 21 (2014), 3415–3419.

A Mitchell, U Lafont, M Hołyńska, and C Semprimoschnig. 2018. Additive manufacturing-a review of 4D printing and future applications. *Additive Manufacturing* (2018).

Yuki Mori and Takeo Igarashi. 2007. Plushie: an interactive design system for plush toys. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 45.

Patrick Mullen, Yiying Tong, Pierre Alliez, and Mathieu Desbrun. 2008. Spectral conformal parameterization. In *Computer Graphics Forum*, Vol. 27. Wiley Online Library, 1487–1494.

Vidya Narayanan, Lea Albaugh, Jessica Hodgins, Stelian Coros, and James Mccann. 2018. Automatic Machine Knitting of 3D Meshes. *ACM Trans. Graph.* 37, 3, Article 35 (Aug. 2018), 15 pages. https://doi.org/10.1145/3186265

Zeev Nehari. 2012. *Conformal mapping*. Courier Corporation.

Amirali Nojoomi, Hakan Arslan, Kwan Lee, and Kyungsuk Yum. 2018. Bioinspired 3D structures with programmable morphologies and motions. *Nature communications* 9, 1 (2018), 3705.

Jifei Ou, Mélina Skouras, Nikolaos Vlavianos, Felix Heibeck, Chin-Yi Cheng, Jannik Peters, and Hiroshi Ishii. 2016. aeroMorph-heat-sealing inflatable shape-change materials for interaction design. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 121–132.

Hans Pedersen and Karan Singh. 2006. Organic labyrinths and mazes. In *Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*. 79–86.

Jesus Perez, Miguel A Otaduy, and Bernhard Thomaszewski. 2017. Computational design and automated fabrication of kirchhoff-plateau surfaces. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 62.

Stefan Pillwein, Kurt Leimer, Michael Birsak, and Przemyslaw Musialski. 2020. On Elastic Geodesic Grids and Their Planar to Spatial Deployment. *arXiv preprint arXiv:2007.00201* (2020).

Rohan Sawhney and Keenan Crane. 2017. Boundary First Flattening. *ACM Transactions on Graphics (TOG)* 37, 1 (2017), 5.

Christian Schüller, Daniele Panozzo, Anselm Grundhöfer, Henning Zimmer, Evgeni Sorkine, and Olga Sorkine-Hornung. 2016. Computational thermoforming. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 43.

Emmanuel Siéfert, Etienne Reyssat, José Bico, and Benoît Roman. 2019. Bio-inspired pneumatic shape-morphing elastomers. *Nature materials* 18, 1 (2019), 24.

Mélina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. 2014. Designing inflatable structures. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 63.

Christopher M Spillmann, Jawad Naciri, Mu-San Chen, Amritha Srinivasan, and Banahalli R Ratna. 2006. Tuning the physical properties of a nematic liquid crystal elastomer actuator. *Liquid crystals* 33, 04 (2006), 373–380.

Skylar Tibbits. 2014. 4D printing: multi-material shape change. *Architectural Design* 84, 1 (2014), 116–121.

Teunis van Manen, Shahram Janbaz, and Amir A Zadpoor. 2017. Programming 2D/3D shape-shifting with hobbyist 3D printers. *Materials Horizons* 4, 6 (2017), 1064–1069.

Guanyun Wang, Tingyu Cheng, Youngwook Do, Humphrey Yang, Ye Tao, Jianzhe Gu, Byoungkwon An, and Lining Yao. 2018a. Printed Paper Actuator: A Low-cost Reversible Actuation and Sensing Method for Shape Changing Interfaces. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 569.

Guanyun Wang, Humphrey Yang, Zeyu Yan, Nurcan Gecer Ulu, Ye Tao, Jianzhe Gu, Levent Burak Kara, and Lining Yao. 2018b. 4DMesh: 4D Printing Morphing Non-Developable Mesh Surfaces. In *The 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, 623–635.

Lining Yao, Jifei Ou, Chin-Yi Cheng, Helene Steiner, Wen Wang, Guanyun Wang, and Hiroshi Ishii. 2015. BioLogic: natto cells as nanoactuators for shape changing interfaces. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1–10.

Yang Zhang, Gierad Laput, and Chris Harrison. 2017. Electrick: Low-cost touch sensing using electric field tomography. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1–14.

Yizhong Zhang, Chunji Yin, Changxi Zheng, and Kun Zhou. 2015. Computational hydrographic printing. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 131.

Haisen Zhao, Fanglin Gu, Qi-Xing Huang, Jorge Garcia, Yong Chen, Changhe Tu, Bedrich Benes, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2016. Connected fermat spirals for layered fabrication. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–10.