Ondulé: Designing and Controlling 3D Printable Springs

Liang He[1], Huaishu Peng[2], Michelle Lin[1], Ravikanth Konjeti[1], François Guimbretière[3], Jon E. Froehlich[1]

[1]Paul G. Allen School of Computer Science
University of Washington
{lianghe, mlin88, jonf}@cs.uw.edu

[2]Computer Science
Univ. of Maryland, College Park
huaishu@cs.umd.edu

[3]Information Science,
Cornell University, Ithaca
francois@cs.cornell.edu

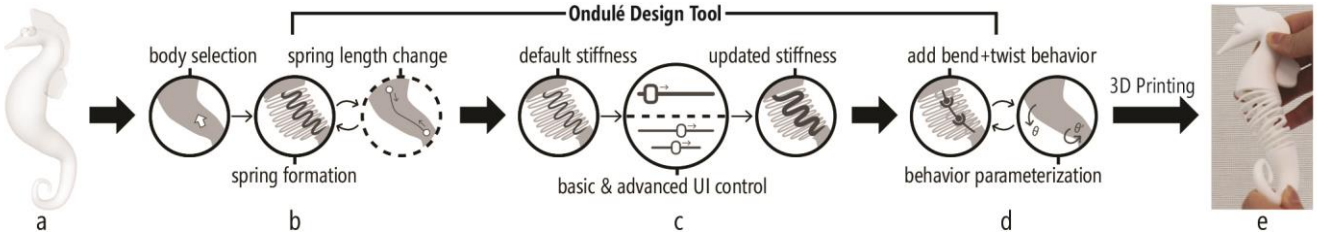

Figure 1. We introduce *Ondulé*, an interactive tool that allows designers to create and control deformable objects with embedded springs and joints. Above, a workflow shows how to make a solid seahorse body bendable and twistable: (a) select a seahorse body; (b) change the spring length and regenerate the spring directly on the model; (c) control spring stiffness; (d) parameterize spring deformation behaviors by adding additional joints; and (e) print the deformable seahorse with a consumer-grade FDM 3D printer.

ABSTRACT

We present *Ondulé*—an interactive design tool that allows novices to create parameterizable deformation behaviors in 3D-printable models using helical springs and embedded joints. Informed by spring theory and our empirical mechanical experiments, we introduce spring and joint-based design techniques that support a range of parameterizable deformation behaviors, including *compress*, *extend*, *twist*, *bend*, and various combinations. To enable users to design and add these deformations to their models, we introduce a custom design tool for Rhino. Here, users can convert selected geometries into springs, customize spring stiffness, and parameterize their design with mechanical constraints for desired behaviors. To demonstrate the feasibility of our approach and the breadth of new designs that it enables, we showcase a set of example 3D-printed applications from launching rocket toys to tangible storytelling props. We conclude with a discussion of key challenges and open research questions.

Author Keywords

Digital fabrication; 3D printing; spring; parameterizable deformation behaviors; parametric design tool; CAD.

ACM Classification Keywords

• **Human-centered computing~Human computer interaction (HCI)**; *Interactive systems and tools*.

UIST '19, October 20–23, 2019, New Orleans, LA, USA

ACM ISBN 978-1-4503-6816-2/19/10...$15.00
DOI: https://doi.org/10.1145/3332165.3347951

INTRODUCTION

3D printing techniques and tools have traditionally focused on supporting static designs (*e.g.*,[13, 19]). Recent work, however, has introduced new methods for creating dynamic objects, including: new 3D printer designs with non-plastic materials to print interactive and functional objects [17, 18] as well as new CAD and algorithmic techniques to support 3D-printable metamaterials [9, 11, 23], linkages [15], hinges [12, 20, 22], telescoping structures [26], and joints [3, 7]. These methods open new design spaces for creating interactive and functional devices on demand. In this paper, we present a new fabrication technique for incorporating one of the most extensively used mechanical structures in traditional manufacturing into fused deposition modeling (FDM) 3D-printing: *helical springs*.

Compared to other mechanisms such as hinges, joints, or metamaterials, helical springs offer several benefits. First, they support a wide range of deformation behaviors, including *compressing*, *stretching*, *bending* and *twisting* [16]. Second, because springs can endure large-scale deformations and store energy [16], they are ideal mechanisms to produce the driving force for motion. Finally, high-density spring structures have small gaps between each coil, making it possible to create complex deformable surface with a standard FDM 3D printer. While helical springs have compelling potential, two key challenges prevent them from being widely used in 3D printing. First, due to the anisotropic characteristic of additive manufacturing, the performance and mechanical properties of 3D-printed helical springs have not been extensively studied [8]. Second, although helical springs support a wide range of deformations, designing, customizing, and controlling the deformation is complex.

To address these challenges, we first investigate the mechanical properties of 3D-printed helical springs with a series of controlled mechanical experiments. The results

indicate that 3D-printed springs perform similarly to theoretical predictions. We then present a set of joint designs (*prismatic joints, revolute joints, knuckle joints*) embedded inside a 3D-printed spring, which can be used to customize and parameterize spring behaviors. Figure 1e shows one spring example based on our exploration. The helical spring is 3D printed with an internal chain of ball joints that allows the seahorse to bend and twist.

Based on our experimental findings and custom joint designs, we developed *Ondulé*—a new interactive design plugin for Rhino that allows novices and makers to rapidly prototype 3D deformable behaviors with spring structures. With *Ondulé*, a user can select arbitrary shapes (Figure 1a), convert them to springs (Figure 1b), control and customize the desired deformation by parameterizing the spring and its internal joints (Figure 1c, 1d), and print the deformable object with a consumer-grade FDM 3D printer (Figure 1e). *Ondulé* also provides real-time feedback about the spring's behavior (*e.g.*, its full compression position); however, a full dynamic simulation remains open work. To highlight the potential of *Ondulé*, we present a series of 3D-printed artifacts designed with our tool: see Figure 17-Figure 21 and our supplementary Video Figure.

In summary, this paper contributes: (i) a set of novel spring deformation techniques with intrinsic mechanical joints that allow for spring behavior customization; (ii) an interactive design tool that allows designers to rapidly convert a static 3D model to a deformable and printable object by controlling spring stiffness and parameterizing additional joints; and (iii) a series of example applications created with *Ondulé* demonstrating feasibility and an initial design space.

RELATED WORK

We build upon prior work in mechanical springs, 3D-printed deformable objects, and computer-aided design (CAD) tools for the parameterization of 3D printable deformations.

Mechanical Springs

Springs are elastic structures that can harness mechanical energy and support a wide range of deformation behaviors. In general, springs are classified as wire springs (*e.g.*, helical springs), flat springs (*e.g.*, cantilever), and special-shaped springs [16]. Compared to other spring types, helical spring can resist and deflect under tensile, compressive, or torsional loads. These properties have been studied in the field of mechanical engineering, material science, and physics [5, 20]. In this paper, we focus on imbuing 3D-printable CAD models with parameterizable *helical* springs.

3D-Printed Deformable Objects

Recently, researchers have explored converting static 3D-printed artifacts into dynamic objects with various mechanisms [3, 9, 15], including 3D printable joints [2, 7], linkages [15], hinges [21, 22], metamaterials [9, 10], and telescoping structures [26]. For example, Calì *et al.* [3] converts static 3D models into articulated ones using 3D printed posable joints. *Coded Skeleton* [12] uses repetitive slit patterns to make planar objects stretchable, bendable and twistable. These works often require high-resolution 3D printers or other fabrication processes (*e.g.*, SLA and SLS) rather than FDM. Recently, various spring structures have also been used as part of deformable mechanisms [8, 11, 16] in 3D-printed artifacts. *FlexMaps* [14], for example, constructs deformable 3D surfaces from 2D flat spiraling microstructures. *Bend-it* [25] adds kinetic properties to 3D-printed models using wire bending techniques. Unlike prior work, our techniques use a combination of 3D printed helical springs with custom joints as the core building block to achieve various deformation behaviors.

CAD Tools for Parameterizing Printable Deformations

Our work also relates to the literature of parametric CAD tools for 3D model customization. One common type of these tools allow users to use predefined modular structures to create deformable artifacts from scratch [3, 10, 11, 15]. The editor in *Metamaterial Mechanisms* [9], for example, allows the user to replace 3D shapes with predefined shear cells. Another common approach for 3D model customization is to directly edit an existing model [22, 26]. For example, the design tool in [26] allows a user to directly work on the skeleton of a 3D model to customize telescoping structures. In *Ondulé*, we employ a similar design approach to [26], which allows designers to directly customize and control a selected shape's deformation by embedding various springs and joints configurations. We differentiate our design tool from traditional CAD tools that support the generation of helix structures, such as *Autodesk Inventor* [27] and *Solidworks* [28], by not only allowing the *automation* of spring+joint behavior generation but also supporting the interactive customization of the deformation with a preview of the performance (*e.g.*, maximum compression, load, *etc.*).

HELICAL SPRING THEORY

Our approach is based on *helical springs* [22], which have three basic configurations—*compression*, *extension*, and *torsion* (*p.* 626 in [4]). Helical spring behaviors (Figure 2) are determined by two interrelated factors: *spring parameters* and *material properties.* We use both in our tool.

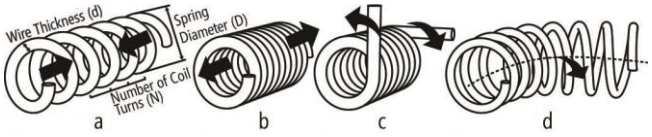

Figure 2. Basic helical spring deformation behaviors: (a) compress, (b) extend, (c) twist, and (d) laterally bend.

Spring parameters. The *compression* and *extension* behaviors of helical springs can be modeled using *Hooke's Law* (Eq. 1) and *Castigliano's theorem* (*p.* 502 in [16]), where a spring's stiffness k is determined by wire thickness d, diameter D, number of coil turns N, and shear modulus G. Similarly, to model the torsion (*i.e.*, twisting) behavior of helical springs, we use the angular form of Hooke's Law (Eq. 2) and Castigliano's theorem (*p.* 534-535 in [16]), where a spring's torsion rate k' is determined by spring parameters and Young's modulus E.

Linking material properties and spring parameters for compression and extension

$$k = \frac{F}{x} = \frac{d^4 G}{8D^3 N} \quad (1)$$

Linking material properties and spring parameters for torsion

$$k' = \frac{\tau}{\theta} = \frac{d^4 E}{64DN} \quad (2)$$

Given that a material's properties are constant, we can manipulate *d*, *D*, and *N* in our design tool to parameterize spring behaviors.

Material properties. There are two relevant material properties to control a helical spring's behavior: *Young's modulus* (*E)* and *shear modulus* (*G)*. (See Appendix A for the formal definitions of *E* and *G* and how we derived *G)*. While *E* is typically listed in filament datasheets (*e.g.*, p. 2 [30]), this *E* is for a single, unextruded portion of filament, and *G* is not listed. Thus, to obtain these values, we need to measure them experimentally for each filament type (*e.g.*, ABS, PLA). We do so for one filament type below.

MECHANICAL EXPERIMENTS

Our mechanical experiments have two primary goals: first, to study the effect of 3D printing on the material properties *E* and *G* for our selected filament type; and second, to explore whether 3D-printed helical springs perform similarly to theoretical predictions. Towards the first goal, we conducted material property tests (Experiment 1) using a load frame to empirically measure *E* and *G* (Figure 3a), based on *American Society for Testing and Materials* (ASTM) standards [1]. For the second goal, we evaluated *tensile* (Experiment 2) and *twisting* (Experiment 3) performance of 3D-printed helical springs using a load frame and a torque sensor, respectively (Figure 3b and 3c). Experimental results are used to inform *Ondulé*'s parameter space and for spring preview. The experiments focus solely on the performance of 3D-printed helical springs and do not include joints.

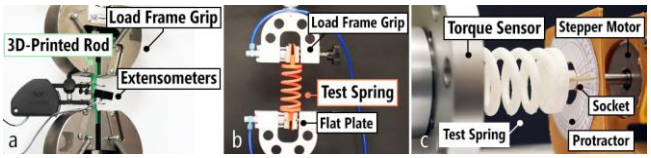

Figure 3. Mechanical experiment setups: (a) the load frame stretches a 3D-printed rod; (b) the load frame stretches a 3D-printed helical; and (c) the motor rotates a helical spring and torque is measured.

All test samples were printed with tough PLA (TPLA [29]) and dissolvable PVA [30] using an Ultimaker 3 printer. PVA was used for support material and fully removed before the experiments. We used Ultimaker Cura 3.6.0 [31] with default print settings and varied infill density, infill pattern, and printing orientation (depending on the experiment). Our experiments were run under the supervision of and consultation with a mechanical testing lab engineer.

Experiment 1: 3D-Printed Rod Material Property Tests

To derive *E* and *G* for 3D-printed TPLA and to explore the effect of 3D printing on these properties, we directly

measured *E* and the Poisson ratio *v* for different 3D printer settings (Figure 4). The experiment setup is shown in Figure 3Figure 3a and detailed in Appendix A.

Condition	Wire Thickness (mm)	Diameter (mm)	Length (mm)	Turn Number
Wire Thickness	2, 3.4, 4.8, 6.2, 7.6	32	50	5
Diameter	4	25, 30, 50, 60	50	5
Spring Length	4	32	25, 45, 65, 85	5
Turn Number	4	32	50	4, 6, 8, 10

Table 1. The conditions of spring parameters used in Experiment 2 and Experiment 3.

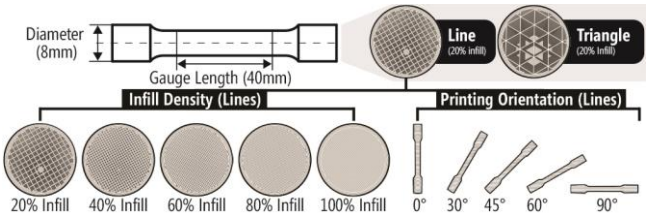

Figure 4. The 3D-printed solid rods in Experiment 1 and three varied test conditions: *infill density*, *infill pattern*, and *print orientation*.

Experiment 1 Results

The experimental results show that (i) stiffness increases as infill density increases, (ii) tensile strength orthogonal to the printing direction is highest, and (iii) shear stress is highest at a 45° angle (Figure 5). See explanations of the results in Figure 5 in Appendix A.

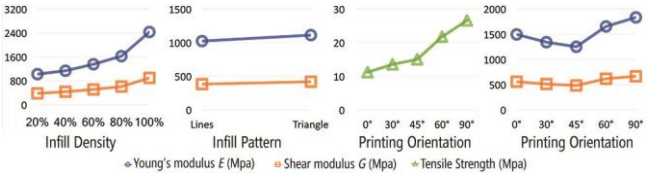

Figure 5. Experiment 1 results showing that *E* and *G* increase with infill density as well as more robust infill patterns. Tensile strength increases as printing angle increases; however, shear stress is highest at 45°.

Experiment 2: 3D-Printed Spring Tensile Tests

To empirically explore how the tensile performance of 3D-printed springs compares to theoretical predictions, we conducted controlled stretching experiments again using the load frame. We varied four spring parameters: wire thickness *d*, diameter *D*, number of coil turns *N*, and spring length *L* (Table 1). While spring theory [16] suggests that length *L* has no effect on tensile performance, we varied this parameter as well for verification. In all, we created and tested 17 helical springs with the same FDM specifications: *100% infill*, *lines infill pattern*, and *90° printing angle*. For the experiments, we followed a similar procedure to Experiment 1 (see the setup in Figure 3b and Appendix A). If our experimental results find that 3D-printed helical springs behave similarly to theoretical predictions, we can then operationalize spring theory in the *Ondulé* tool (*e.g.*, by allowing the user to control thickness *d*, diameter *D*, and number of coil turns *N*).

Experiment 2 Results

To investigate how 3D-printed spring behaves, we compared the empirically measured spring stiffness *k* of each spring to a *k* derived from Eq 2. For *d*, *D*, and *N*, we simply use our experimental conditions (Table 1) as input values. For *G*, we use the value derived from Experiment 1 for 100% infill, lines infill pattern, and 90° printing angle. Using a paired (two-tailed) t-test, we found no significant difference (Figure

6) between the empirically measured k and the theoretical prediction (t_{32}=0.0097, p=0.99).

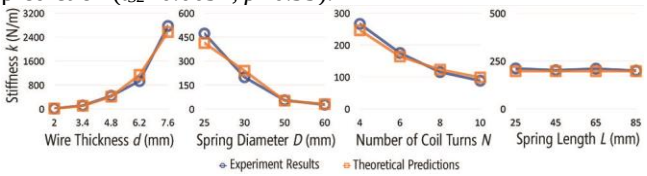

Figure 6. Experiment 2 results showing that 3D-printed helical springs perform similarly to theoretical predictions as measured by a load frame with different *d*, *D*, *N*, and *L* values.

Experiment 3: 3D-Printed Spring Torsion Tests

Finally, for our last experiment, we investigated the torsion (twisting) performance of 3D-printed springs. We reprinted the same springs used in Experiment 2 (Table 1) with the same FDM specifications (100% infill, lines infill pattern, and 90° printing angle); however, we used a different experimental setup (see Figure 3c and Appendix A).

Experiment 3 Results

Similar to Experiment 2, we compared our empirical results—in this case, the measured torsion rate k' for each spring—to theoretical predictions (Eq. 2). As before, we can input the experimental condition values for d, D, and N into Eq. 2 as well as the material property E from Experiment 1 to derive the theoretical prediction k' for each spring. Figure 7 shows our measurement for k' closely mirror the theoretical prediction based on a paired t-test (t_{32}=0.0236, p=0.98).

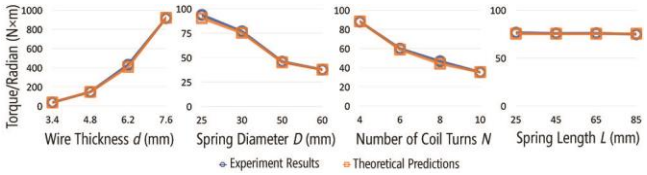

Figure 7. Experiment 3 results showing that 3D-printed helical springs have similar twisting performance to theoretical predictions with varied *d*, *D*, *N*, and *L* values.

Summary of Findings and Design Implications

In summary, our experiments derived E and G for 3D-printed TPLA, demonstrated the feasibility of 3D printing helical springs using FDM 3D printers, and showed that 3D-printed helical springs perform similarly to theoretical predictions. Assuming that friction forces in the joint mechanisms are negligible, we use these results to provide a preview of spring deformations (*e.g.*, the max load necessary for compressing a given spring) in our design tool.

THE ONDULÉ SYSTEM

Informed by spring theory, the above experiments, and our own extensive testing of 3D-printed helical springs, we created *Ondulé*—an interactive design tool for rapidly prototyping and fabricating models with embedded springs. With *Ondulé*, novices can select arbitrary solid geometries and convert them to parameterizable springs that can be printed with consumer-grade FDM 3D printers. *Ondulé* provide controls for directly manipulating and controlling a spring's shape, deformation behavior (*e.g.*, compressing *vs.* twisting), and parameters d, D, N, and L. Below, we first describe our custom spring deformation techniques, which uses mechanical joints as constraints for enabling our approach, before describing the design tool itself.

Spring Deformation Techniques

To enable users to create parameterizable springs, we introduce a set of deformation techniques that combine auto-generated helical springs with embedded mechanical joints. For some deformations, we also enable users to auto-generate a *lock* mechanism that allows the spring to be locked/unlocked in a fixed position (currently supported by *linear-only* and *twist-only* deformations).

Controlling Linear Deformations Using a Prismatic Joint

To create a spring that can *only* compress or extend, we embed a prismatic joint consisting of a shaft, a rail guide, and an embedded slider (Figure 8). When the spring is compressed or stretched, the slider can only move along the predefined rail, preventing the spring from being bent or twisted. The user can specify the *amount* of compression and/or extension, which we control by positioning the default location of the slider. Consequently, this joint design can be used to support *compression-only*, *extension-only*, or *both*. The user can also auto-generate a locking mechanism, which can be used to lock a spring at its maximum compression and/or extension states. We do this by generating a 'latching' groove at the endpoints of the guide rail (Figure 8).

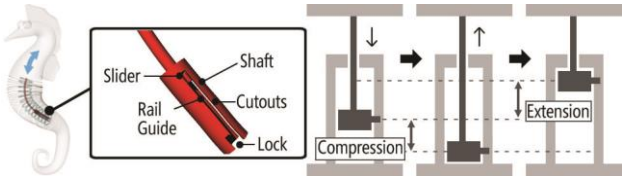

Figure 8. A prismatic joint is used for a *linear-only* deformation.

Controlling Twisting Deformations Using a Revolute Joint

To create a spring with the *twist-only* behavior and to control the maximum angle of rotation, we embed a revolute joint using a bearing socket and a circular disc (Figure 9). This revolute structure allows the circular disc to revolve concentrically to the bearing socket while preventing bending or linear deformations. For controlling the angle of rotation, we embed a custom arc sliding rail in the socket, which confines the rotation of the disc within the maximum twisting angle. Again, we generate a 'latching' groove at the position where the disc rotates to the maximum twisting angle, so as to lock a spring at its maximum angle.

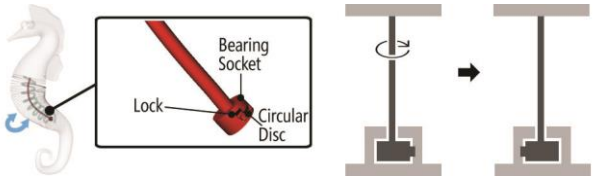

Figure 9. A revolute joint is used for a *twist-only* deformation.

Controlling Bending Deformations Using a Knuckle Joint

To support *bend-only* behaviors, we use a chain of knuckle joints. A single knuckle joint contains a cylindrical rod located inside a cylindrical socket (Figure 10). The

cylindrical rod revolves concentric to the axis of a cylindrical socket. Multiple knuckle joints can be chained nose to tail, with the first and the last one fixed to the two ends of the helical spring. This structure prevents linear and twisting deformations and offers flexible bending. However, knuckle joints confine the bending deformation in one plane. An omni bend-only deformation is also possible, which we describe below.

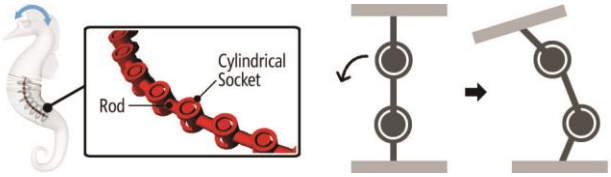

Figure 10. A chain of knuckle joints used for *bend-only* deformations.

Supporting Compound Behaviors

While the above techniques support individual deformation behaviors, we also support compound behaviors in three ways: first, via traditional freeform springs (unconstrained by internal joints), second, via additional joint designs and finally, by combining multiple springs in serial or parallel.

Free-form springs. Users can select geometries and convert them to free-form springs (without embedded joints). These springs can inherently *compress*, *extend*, *bend*, and *twist*. Here, users can only control spring stiffness (via parameters *d*, *D*, *N*, and *L*) and shape.

Additional joint designs. Using custom joint designs, we support two compound behaviors: *linear+twist* and *twist+bend*. For *linear+twist*, we adapt the *linear-only* design by replacing the slider with a circular disc in the shaft, which is a *cylindrical joint* (Figure 11a). This enables the disc to glide and twist along the rail guide. For *twist+bend*, rather than embedding a chain of knuckle joints, we embed a chain of ball joints. This allows the spring to twist and bend at any angle (Figure 11b).

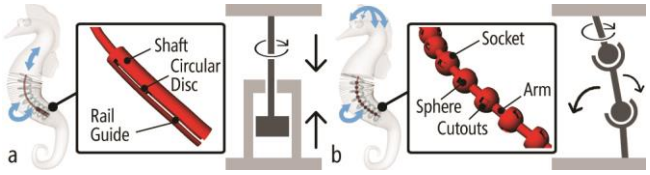

Figure 11. (a) A cylindrical joint is used for *linear*+twist deformations and (b) a chain of ball joints is used for *twist+bend* deformations.

Serial/parallel combinations. Multiple springs can be combined in serial or parallel to further produce more complex deformations. For example, our snake design in Figure 21 combines different spring types serially; our hand exerciser (Figure 19) uses *linear-only* springs in parallel. However, note that *Ondulé* does not yet support previewing these combined behaviors (see Discussion).

Interactive Spring Design Tool

The above techniques are integrated into our custom interactive design tool, *Ondulé*, via a plugin for Rhino 5. *Ondulé* enables novices to rapidly create deformable 3D-printed objects using embedded springs. To use *Ondulé*, the user (1) models geometries (*i.e.*, 3D bodies) in the traditional Rhino CAD environment; (2) selects specific bodies and converts them to springs using *Ondulé*; (3) then parameterizes spring stiffness; (4) and specifies the spring deformation behavior (*e.g.*, *linear-only* or *twist*). Stages 2-4 are supported via a side panel in Rhino (Figure 12). Below, we describe stages 2-4 before providing details on the implementation and underlying algorithms. See the accompanying video figure for a full demonstration.

Generating Springs

After creating a 3D model in Rhino, the user can use our tool to generate spring structures. To generate a spring, the user selects a 3D body in Rhino and then clicks on the '*Convert to spring'* button. If the selected body is cylindrical with a consistent diameter, the 3D shape is converted into a deformation spring automatically. If the selected body is determined to be a non-cylinder geometry, our tool will

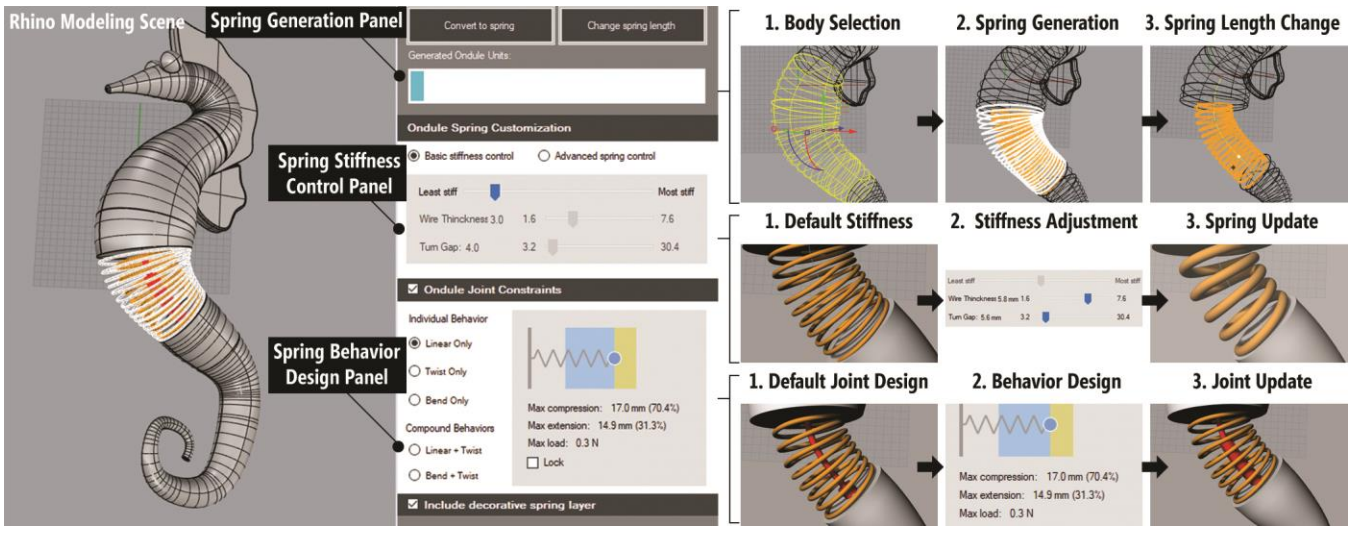

Figure 12. The *Ondulé* spring design tool interface (left) has four parts: Rhino modeling environment, a spring generation panel, a spring stiffness control panel, and a spring behavior design panel. The workflow for each design panel is shown on the right.

convert it into two springs: an internal *deformation* spring that follows the medial axis of the selected geometry (Figure 12) as in the cylindrical case, and an extra outer *decorative* spring that follows the body's geometric form and is created with dense and thin layers of coil (d=1.6mm), which has a minimal effect on the overall stiffness (see Discussion). The decorative spring maintains the complex topology of the selected geometry while the internal spring serves as the functional spring for deformations. By default, the decorative spring (if generated) is hidden to reduce visual clutter, though it can be turned on via a checkbox.

Controlling Spring Stiffness

By default, the system automatically generates the spring diameter D and its length, but these parameters can be adjusted. The users can change the spring stiffness either using a simple slider or by directly modifying the spring wire thickness d and the number of turn N (Figure 12).

Specifying Deformation Behavior

Finally, the user can specify the spring's deformation behavior: *linear-only*, *twist-only*, *bend-only*, *linear+twist*, or *twist+bend*. For each deformation, we provide a custom UI panel. For the compound behaviors we combine the UI from their respective individual panels.

Linear-only. For *linear-only*, the user specifies the *maximum compression* and *extension* points of the spring. We provide real-time feedback about the spring's displacement (shown in millimeters and percentage of L) as well as the estimated force (in Newtons) required for that displacement. The user can also click on the *'Lock'* checkbox to auto-generate a lock mechanism at the maximum compression and extension points.

Twist-only. For *twist-only*, the user can control the *maximum twisting angle* up to 90°, which we found is a safe maximum angle preventing the spring from buckling in our torsion test. As the user drags the angle selector, the UI shows the selected angle (in degrees) as well as the estimated force (in Newtons) required to reach that angle. Similar to the linear-only UI panel, the user can add a lock mechanism at the maximum twisting point by clicking on the *'Lock'* checkbox.

Bend-only. For *bend-only*, the user first specifies the *bending direction* via an angle selector, which overlays a 3D direction indicator on top of the model in Rhino, and then specifies the maximum *bending angle* using a second angle selector (shown in degrees). Unlike the other UI panels, we do not show force estimates. As noted in the Theory, modeling *bending* behavior is an open area of research.

Implementation

We implemented *Ondulé* in C# using Rhino 5's plugin architecture (*RhinoCommon API* [32]). Below, we describe how we computationally generate the springs, the embedded joints, and the locking mechanism.

Springs Generation

To generate deformation springs, we first compute the *medial axis* of the selected 3D shape using a *mean curvature flow* (MCF) algorithm [24]. The medial axis is a skeletal curve at the center of the selected 3D body. It is used to evaluate if a spring structure can be successfully converted and if a decorative spring is needed to preserve the 3D shape's appearance.

To determine printability, we compute the average minimal distance dis_{crv} from the medial axis curve to the 3D shape surface using fixed sampling (Figure 13). If the distance is larger than the minimal diameter of a printable spring (3.6mm for D based on our mechanical experiments) the selected shape can be converted to the spring structure. Using dis_{crv} we also evaluate if the selected shape has complex surface topology that is worth preserving with a decorative spring. This is done by comparing the actual variance of the samples with respect to the mean. If the variance is above a certain threshold, we generate the decorative spring.

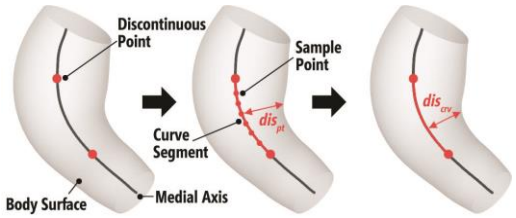

Figure 13. Generating the medial axis, calculating the size of the selected body, and evaluating the printability of an embedded spring.

To generate the deformation spring (Figure 14), we use the *RhinoCommon* `spiral` function. This function produces a spiral curve following the medial axis. The spiral curve is then used as a parameter for the `sweep` function to create the final springs. All spiral solids are concatenated to construct a spring (highlighted in yellow in Figure 14).

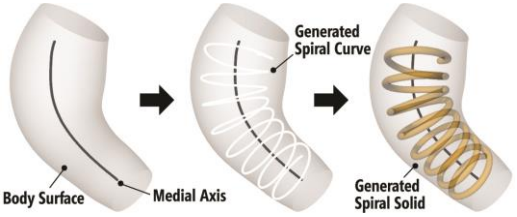

Figure 14. Generating the deformation spring using the generated medial axis and *RhinoCommon* functions.

Finally, if needed, we generate a decorative spring to preserve the original 3D shape appearance. Note that the `spiral` function cannot be used directly for the decorative spring generation, as it can only produce a helix with a consistent radius. Instead, we first reuse the spiral curve generated for the deformation spring as described above and project it onto the 3D shape surface using a 300-sampling point (Figure 15). We then generate a new set of points by retracing all intersecting points toward the median axis curve by a fixed decorative spring wire thickness (*i.e., 1.6mm*) and create a new `curve` object by interpolating these points. Using this curve, the `sweep` function is applied to create the final decorative spring.

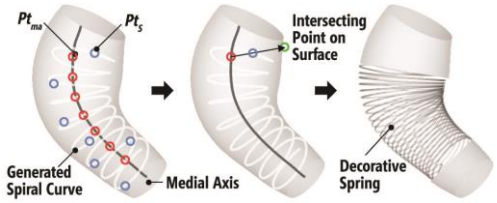

Figure 15. Generating the decorative spring.

Generating Embedded Joints

We compute the embedded joints using the medial axis generated from the previous steps. For *linear-only* joint, we first decide the slider's starting position on the medial axis and calculate its possible extension and compression distances based on the user's input. We then extrude the slider rod, the shaft, and the rail guide sweeping along the medial axis (Figure 16a). *Twist-only* and *linear+twist* joints share a similar procedure, except that the slider position varies in joint design.

Bend-only and *twist+bend* behaviors use chained joints design. To generate these joints, we first calculate the number of joints that are needed in the selected body. For each joint (*i.e.,* a knuckle joint or a ball joint), we determine the position and the length of the inner bearing stud on the medial axis. Next, we extrude the solid stud along the medial axis and generate a cylinder (for *bend-only*) or a sphere (for *twist+bend*) at the endpoint of the stud. Finally, we generate the outer bearing socket for each of the joint and then connect it to one end of the bearing stud of the next joint, which results in a chain structure. Figure 16b shows an exploded view of the construction of the ball joint chain.

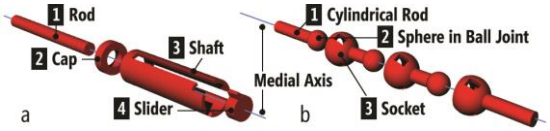

Figure 16. An exploded view of (a) a prismatic joint and (b) a chain of ball joints, which are used for *linear-only* and *twist+bend* respectively.

Generating Locks

We use latching grooves as the locking mechanisms for *linear-only* and *twist-only* deformation behaviors. For the locking mechanism in *linear-only* design, we first locate the endpoints of the shaft and then generate a groove next to the rail guide by executing a *Boolean difference* operation from the shaft with a `cubic` object. To avoid the slider to slip out of the groove, we also generate a fence on the edge of the groove (Figure 8). Similarly, for the revolute joint in *twist-only* design, we generate the groove at the position where the disc rotates at the maximum twisting angle (Figure 9).

EXAMPLE APPLICATIONS

To illustrate the potential of our approach and highlight an initial design space, we created five examples, each emphasizing one or more features of *Ondulé.* See the supplementary video for full demonstrations.

Jack-in-the-box

The jack-in-the-box is one of the most well-known helical spring-based toys. Here, we showcase how conventional spring-based mechanism can be enhanced using the *Ondulé* design tool. Figure 17 showcases the jack-in-the-box design, where instead of using an unconstrained helical spring, the center spring is built to follow a predefined curve and with a *compress+twist* deformation behavior, which is achieved by using a freeform spring and a spring with a cylindrical joint. After cranking, a cat figure pops out of the box up with a turning motion. The spring and the cranking components were 3D printed and the box is made by laser cutting.

Figure 17. A jack-in-the-box spring mechanism generated by *Ondulé*: (a) two spring designs are embedded; (b) the cat can be fully compressed and locked inside a laser-cut box; and (c) the cat pops out following a path as a surprise.

Launching Rocket

Another set of spring-based applications is to use the deformation to produce the driving force for mechanical motion, such as the proper shooting motion in our launching rocket example (Figure 18). We first select the smoke shape and convert it into a *compress-only* spring by adding a prismatic joint. We then created an additional latch structure, which when released will push the rocket fly straight-up.

Figure 18. A launching rocket application: (a) a rocket sits on top of a compressed "smoke" spring, which is locked by an external latch; (b) the user can launch the rocket by pulling the latch; and (c) the smoke is in its full extension.

Note that though a simple example, such application will be difficult to create without *Ondulé*. First, by adding a prismatic joint, we can ensure that the rocket will be pushed in the right direction without twisting or bending during the launching motion. Second, the prismatic joint resides inside the conical smoke shape, serving the linear constraint function with minimal effect on the 3D shape appearance.

Hand Exerciser

The previous two examples showcase how we can use *Ondulé* for single spring deformation behavior. Here, we demonstrate how multiple springs can be created and customized through a set of hand exercisers (Figure 19). Commercial hand exercisers are widely used for hand rehabilitation or Arthritis therapy. Figure 19a is a commercial design, where multiple springs are located in parallel for finger exercises. Figure 19b is our *Ondulé* replication using

four freeform springs in parallel for each of the fingers and three extra springs for the palm. One limitation of the commercial hand exercisers is that springs for all fingers have the same stiffness. As such, it would be impossible for a user to exercise different fingers with different strength. To address this limitation, we designed a custom hand exerciser (Figure 19c) in which for each finger the user can customize the spring stiffness by adjusting the spring parameter *d* and *N*. Finally, users can also create their own designs in arbitrary geometries, for example, a blowfish shape (Figure 19d).

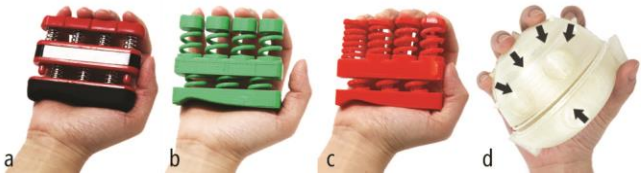

Figure 19. Replicated and custom hand exercisers: (a) an off-the-shelf hand exerciser (b) a replication with 3D-printed springs, (c) a custom design that includes springs with different stiffnesses and custom prismatic joints for compress-only behavior, and (d) a blowfish version.

Tangible Prop for Storytelling Authoring

Custom 3D-printed deformable springs can be further combined with external electrical components to create expressive interactions. In this example, we created a digital storytelling authoring tool with a tangible prop made with a 3D-printed animal and low-cost sensors (Figure 20).

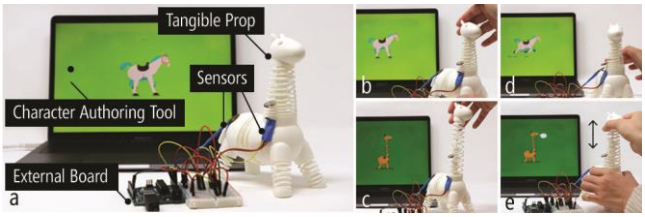

Figure 20. The setup of a tangible storytelling prop.

The 3D-printed prop is composed of a stretchable neck (with a prismatic joint design and a lock mechanism), a bendable body (with a knuckle joint design), and four freely deformable legs. With only one physical design, two animal characters (*i.e., horse* and *giraffe*) with two separate actions (*i.e., talking* and *walking*) can be mapped to the prop due to the spring design. For example, a digital horse will show up in our authoring tool when the prop's neck part stays unstretched (Figure 20b). When extended, the physical prop has a longer neck thus a giraffe will show up on the screen accordingly (Figure 20c). To detect the length change of the neck, a linear hall effect sensor is used and attached to the bottom of the neck with a magnet fixed above it (Figure 20a). The sensor is connected to an Arduino, which communicates to the authoring tool developed with Processing. While either character is activated, moving the head up and down can trigger the character's talking action (Figure 20e). We can also attach a piezo sensor to the prop body to detect vibration, which can be used as a walking action trigger (Figure 20d). The repetitive tap can be robustly detected by the sensor due to the converted bendable body and springy legs.

Other Applications

Here we showcase other applications designed using *Ondulé*. We collaborated with a mechanical engineering team and created an accessible cutting device for people with muscle weakness (Figure 21a). With the *Ondulé* design tool, we can rapidly make custom springs that can fit in the cutting device and offer the exact amount of stiffness needed for the patient. Figure 21b shows a Halloween mask as an example of wearable fashion. The trunk of the elephant is designed with a chain of ball joints so that it is flexible, bendable, and lightweight to wear. Finally, various joints can be incorporated in a snake body in a serial arrangement (Figure 21c). We envision a robotic snake controlled by an external control system and actuators.

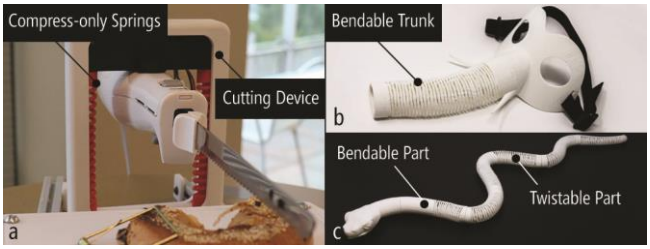

Figure 21. Other applications that *Ondulé* can support: (a) an accessible cutting device, (b) an elephant mask with a bendable trunk, and (c) a snake body with multiple spring deformation behaviors.

DISCUSSION AND FUTURE WORK

Geometry Printability

A main objective of *Ondulé* is to support deformation customization with 3D-printed springs. We achieve this by inserting custom joint inside helical springs, so that spring behaviors become modular without requiring external supporting structures. However, the size of an object is limited to the minimum joint size that we can print (currently a minimum 0.4mm tolerance is needed between the moving and the stationary parts of joint) and the minimal spring wire thickness (1.6mm in our current setup) that is printable. As a result, it is difficult to convert geometries with smaller diameter (*i.e.*, smaller than 3.6mm) with our current tool. However, joint size and spring wire thickness could be further reduced with a higher resolution printer or alternative printing process (*e.g.*, SLA).

Another limitation of our approach is that it may affect the degree of deformation. For instance, compression with our current prismatic joint design cannot surpass half the length of the original spring. Similarly, a stretching behavior with the prismatic joint cannot exceed two times its original length, even though the helical spring solely could be further extended. One solution to this is to consider alternative joints design. For instance, we can replace the rail with a prismatic telescoping structure [26].

Influence of Friction Force and Decorative Spring

Our mechanical experiments focused solely on the behavior of 3D printed helical springs. With the embedded mechanical joints, we further understand its persistent friction force with an additional experiment. Here, we compare the oscillating

behavior of two springs printed with and without a prismatic joint (*S1* and *S2*, respectively). We observed the relaxation behavior of these springs when stretched to the same length in three orientations (vertical—0°, 45°, and horizontal—90°). The results show that S1 comes to rest 48%, 50%, 70% faster, respectively. As expected, the horizontal setting has the most friction. These preliminary results indicate that friction force exists in 3D-printed joints, which impede normal spring behavior. Future work should explore methods to reduce friction. For example, we found that joint-based friction can be diminished by using a filament material with a lower friction coefficient (*e.g.*, ABS) or adding grease. We confirmed that an ABS spring reduces friction by 24% in the oscillation test setting.

For some 3D models, *Ondulé* generates both a *deformation* spring and a *decorative* spring—the latter enables us to approximate organic surface topology while maintaining form. The decorative spring is not intended to influence overall spring behavior, only the aesthetic. To examine the effect of the decorative spring on spring performance, we compared the overall stiffness *k* of springs with and without an added decorative spring. As desired, we found a minimal impact: an increase of 0.02% in the jack-in-the-box application and 0.21% in the rocket application. When analyzing the current *Ondulé* parameter space, the expected impact could be up to 4.42%. Note that in that case, we could modify the design of the deformation spring to compensate for this effect. These preliminary results indicate that the decorative spring has minimal effects on the overall deformation of a 3D-printed object.

Simulation for the Combined Behavior

Ondulé allows a user to design the spring+joint deformation with a preview of its starting and end positions. However, our tool does not currently have the capability to simulate *combined* deformation motions. In the future, we plan to provide a more realistic validation for the spring+joint behaviors by simulating the effect of the intrinsic spring weight (with specific printing settings), the friction force of the internal joints and external forces (*e.g.*, the applied user interaction force). One approachable solution is to use a physical engine such as *Kangaroo* [33], which can be integrated into Rhino to build an interactive simulation environment that offers designers a realistic motion and mechanical preview.

Spring Robustness

As discussed in Mechanical Experiments, printing orientation will affect the spring's *E* and *G*, where 45° results in minimum values for both and 90° yields the highest. As such, all models presented in this paper are printed with the spring perpendicular to the 3D printer's Z-direction for robust printing results. However, when multiple springs with varied orientations need to be printed at once, it would be challenging to find an ideal printing orientation that works for all springs. As the simplest solution to this problem, we can take a print-and-assembly approach, where certain springs can be printed separately with the optimal printing angle and then affix to the original model. Another approach is to involve alternative 3D printing method. The 5-DOF 3D printing method can be used in this case to always repose the 3D model to ensure the spring can be aligned with the Z axis. Since we characterized the impact of orientations on strength, a constant overall stiffness can be achieved.

CONCLUSION

In this paper, we present *Ondulé*, an interactive design tool that allows the user to rapidly design and build deformable plastic objects with parameterizable spring and mechanical joints. To design *Ondulé*, we first studied the feasibility of 3D-printed spring with a series of controlled mechanical experiments. We then proposed a set of spring and joint design which enables the customization of a spring deformable behavior. With *Ondulé*, novices can quickly add a wide range of spring structures and deformation behaviors to an existing model. We showed the breadth of our approach with a set of examples.

ACKNOWLEDGMENTS

This work was supported in part by NSF Awards IIS1422106. We acknowledge Prof. Mark D. Fuge for his advice. We thank William F Kuykendall and Joshua Land for their assistance with the mechanical experiments, Ruofei Du for his support on the MCF implementation, Sophie Tian for her help with the mask application, Dhruv Jian, Hanzi "Lotus" Zhang, Orson Xu, and Qisheng Li for their time and efforts with video shooting. We also thank reviewers for their valuable feedback.

REFERENCES

[1] ASTM. 2018. ASTM A370: Standard Test Methods and Definitions for Mechanical Testing of Steel Products. *ASTM International*: 1–50. https://doi.org/10.1520/A0370-17A.2

[2] Moritz Bächer, Bernd Bickel, Doug L. James, and Hanspeter Pfister. 2012. Fabricating articulated characters from skinned meshes. *ACM Transactions on Graphics* 31, 4: 1–9. https://doi.org/10.1145/2185520.2185543

[3] Jacques Calì, Dan A. Calian, Cristina Amati, Rebecca Kleinberger, Anthony Steed, Jan Kautz, and Tim Weyrich. 2012. 3D-printing of non-assembly, articulated models. *ACM Transactions on Graphics* 31, 6: 1. https://doi.org/10.1145/2366145.2366149

[4] Peter R.N. Childs. 2004. *Mechanical Design Engineering Handbook*.

[5] Rouhollah D. Farahani, Kambiz Chizari, and Daniel Therriault. 2014. Three-dimensional printing of freeform helical microstructures: A review. *Nanoscale* 6, 18: 10470–10485. https://doi.org/10.1039/c4nr02041c

[6] Miguel Fernandez-Vicente, Wilson Calle, Santiago Ferrandiz, and Andres Conejero. 2016. Effect of Infill

Parameters on Tensile Mechanical Behavior in Desktop 3D Printing. *3D Printing and Additive Manufacturing* 3, 3: 183–192. https://doi.org/10.1089/3dp.2015.0036

[7] Mark Fuge, Greg Carmean, Jessica Cornelius, and Ryan Elder. 2015. The MechProcessor: Helping Novices Design Printable Mechanisms Across Different Printers. *Journal of Mechanical Design* 137, 11: 111415. https://doi.org/10.1115/1.4031089

[8] Jacob Roth and Sharon Gerbode. 2018. Springs with Bones: Several new discoveries in the creation and use of nonlinear springs. *Unarchived.*

[9] Alexandra Ion, Johannes Frohnhofen, Ludwig Wall, Robert Kovacs, Mirela Alistar, Jack Lindsay, Pedro Lopes, Hsiang-Ting Chen, and Patrick Baudisch. 2016. Metamaterial Mechanisms. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16*: 529–539. https://doi.org/10.1145/2984511.2984540

[10] Alexandra Ion, Robert Kovacs, Oliver S Schneider, Pedro Lopes, and Patrick Baudisch. 2018. Metamaterial Textures. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*: 1–12. https://doi.org/10.1145/3173574.3173910

[11] Alexandra Ion, Ludwig Wall, Robert Kovacs, and Patrick Baudisch. 2017. Digital Mechanical Metamaterials. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*: 977–988. https://doi.org/10.1145/3025453.3025624

[12] Miyu Iwafune, Taisuke Ohshima, and Yoichi Ochiai. 2016. Coded Skeleton: Programmable Deformation Behaviour for Shape Changing Interfaces. *SIGGRAPH ASIA 2016 Emerging Technologies*, p. 1. https://doi.org/10.1145/2988240.2988252

[13] Rubaiat Habib Kazi, Tovi Grossman, Cory Mogk, Ryan Schmidt, and George Fitzmaurice. 2016. ChronoFab: Fabricating Motion. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*: 908–918. https://doi.org/10.1145/2858036.2858138

[14] Luigi Malomo, Jesús Pérez, Emmanuel Iarussi, Nico Pietroni, Eder Miguel, Paolo Cignoni, and Bernd Bickel. 2018. FlexMaps: computational design of flat flexible shells for shaping 3D objects. *ACM Transactions on Graphics* 37, 6: 1–14. https://doi.org/10.1145/3272127.3275076

[15] Vittorio Megaro, Jonas Zehnder, Moritz Bächer, Stelian Coros, Markus H. Gross, and Bernhard Thomaszewski. 2017. A Computational Design Tool for Compliant Mechanisms. *ACM Trans. Graph* 36, 4. https://doi.org/10.1145/3072959.3073636

[16] Budynas-Nisbett. 2008. Shigley's Mechanical Engineering Design (8th Edition). *McGraw-Hill Science*: 499–548.

[17] Huaishu Peng, Jennifer Mankoff, Scott E. Hudson, and James McCann. 2015. A Layered Fabric 3D Printer for Soft Interactive Objects. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*: 1789–1798. https://doi.org/10.1145/2702123.2702327

[18] Huaishu Peng, François Guimbretière, James McCann, and Scott E. Hudson. 2016. A 3D Printer for Interactive Electromagnetic Devices. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST '16:* 553-562. https://doi.org/10.1145/2984511.2984523

[19] Romain Prévost, Emily Whiting, Sylvain Lefebvre, and Olga Sorkine-Hornung. 2013. Make It Stand : Balancing Shapes for 3D Fabrication. *ACM Transactions on Graphics* 32, 4: 81. https://doi.org/10.1145/2461912.2461957

[20] Dan Raviv, Wei Zhao, Carrie McKnelly, Athina Papadopoulou, Achuta Kadambi, Boxin Shi, Shai Hirsch, Daniel Dikovsky, Michael Zyracki, Carlos Olguin, Ramesh Raskar, and Skylar Tibbits. 2014. Active printed materials for complex self-evolving deformations. *Scientific Reports* 4: 1–8. https://doi.org/10.1038/srep07422

[21] Michael L. Rivera, Melissa Moukperian, Daniel Ashbrook, Jennifer Mankoff, and Scott E. Hudson. 2017. Stretching the Bounds of 3D Printing with Embedded Textiles. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*: 497–508. https://doi.org/10.1145/3025453.3025460

[22] Andrew O. Sageman-Furnas, Nobuyuki Umetani, and Ryan Schmidt. 2015. Meltables: Fabrication of Complex 3D Curves by Melting. *SIGGRAPH Asia 2015 Technical Briefs*, Figure 2: 14:1--14:4. https://doi.org/10.1145/2820903.2820915

[23] Christian Schumacher, Bernd Bickel, Jan Rys, Steve Marschner, Chiara Daraio, and Markus Gross. 2015. Microstructures to control elasticity in 3D printing. *ACM Transactions on Graphics* 34, 4: 136:1-136:13. https://doi.org/10.1145/2766926

[24] Andrea Tagliasacchi, Ibraheem Alhashim, Matt Olson, and Hao Zhang. 2012. Mean curvature skeletons. *Eurographics Symposium on Geometry Processing* 31, 5: 1735–1744. https://doi.org/10.1111/j.1467-8659.2012.03178.x

[25] Hongyi Xu, Espen Knoop, Stelian Coros, and Moritz Bächer. 2018. Bend-it: Design and Fabrication of Kinetic Wire Characters. *ACM Transactions on Graphics* 37, 6: 1–15. https://doi.org/10.1145/3272127.3275089

[26] Christopher Yu, Keenan Crane, and Stelian Coros. 2017. Computational Design of Telescoping Structures. *ACM Transactions on Graphics* 36, 4: 83:1--83:9. https://doi.org/10.1145/3072959.3073673

[27] Autodesk Inventor. https://www.autodesk.com/products/inventor/overview

[28] Solidworks: https://www.solidworks.com/

[29] Tough PLA. https://ultimaker.com/en/products/materials/tough-pla

[30] PVA. https://ultimaker.com/en/products/materials/pva

[31] Ultimaker Cura. https://ultimaker.com/en/products/ultimaker-cura-software

[32] RhinoCommon API. https://developer.rhino3d.com/api/

[33] Kangaroo physics from https://www.food4rhino.com/app/kangaroo-physics

[34] Cura infill from https://ultimaker.com/en/resources/52670-infill

APPENDIX A

Terminology and Concepts in Helical Spring Theory

Young's modulus (*E)* is a measure of an object's resistance to being deformed elastically (*i.e.,* non-permanently) with applied stress. *E* is defined as the ratio of *tensile stress σ* (the stress state leading to expansion) to *tensile strain ε* (the relative length of deformation under tensile force)—see Figure 22a and Eq. 3.

Shear modulus (*G)* measures an object's tendency to shear when acted upon by opposing forces. *G* is defined as the ratio of *shear stress* (the stress state leading to shear parallel to the cross section of the material) to *shear strain* (the relative length of deformation under shear force)—see Figure 22b and Eq. 4.

Poisson ratio (*v*) measures how much a material expands perpendicular to the direction of compression or extension. The relationship between *E* and *G* can be derived using *v*—see Figure 22c and Eq. 5.

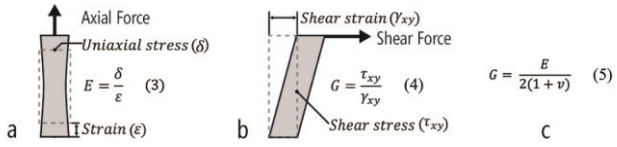

Figure 22. Material properties (a) Young's modulus *E* and (b) shear modulus *G*. *G* can be derived using *E* and Poisson ratio *υ* (c).

Mechanical Experiment 1 Setup

Following ASTM guidelines, we printed 60 solid test rods (Figure 3). We created two specimens for each combination and varied *infill density*, *infill pattern*, and *printing orientation.* Although material properties are not significantly impacted by varying infill patterns [6], we included this condition for completeness and compared line *vs* triangle infills with 20% densities.

For the experiment itself, we used an *Instron 5585H 250kN* electro-mechanical load frame (Figure 3a), which works by gripping a test specimen and performing computer-controlled mechanical operations like stretching and compressing. In our case, we loaded individual test rods and performed a controlled tensile (stretching) operation, which separated the grips at 30mm/minute. The load frame's data logger recorded the applied load, grip displacement, tensile stress *σ,* and tensile strain *ε* at 10Hz. Using these measurements, the load frame automatically calculates *E* (from Eq. 3). To measure *v*, each test rod was also instrumented with two additional digital sensors: an *Instron 2630-106* axial extensometer and an *Instron 2640-008* transverse extensometer. Under tensile stress, the axial extensometer measured changes in rod length (axial elongation) and the transverse extensometer measured changes in rod diameter (transverse elongation). Both were also logged at 10Hz.

Mechanical Experiment 1 Results

For infill density, we expect that as density increases, the Young's modulus *E* and the shear modulus *G* will also increase. That is, as the 3D-printed object becomes more solid, the force required to stretch or shear increases. Indeed, this is what we found: Figure 5 shows that regardless of printing orientation, *E* and *G* grow large as the infill density increases. In terms of infill pattern, because the triangle is a more robust fill compared to lines [28], we expect that the triangle pattern will have a comparatively higher *E* and *G* at all printing orientations. Our results (Figure 5) confirm this prediction: *E* and *G* are higher for all printing orientations with triangle infills *vs.* lines. Finally, for the printing orientation tests, it is well known that FDM printers create 3D models with anisotropic properties—models are stronger in the X and Y direction compared to the Z direction (*i.e.,* a tensile load orthogonal to the FDM layers is the weakest). As expected, Figure 5 shows that as the printing orientation shifts from 0° (rod is printed vertically) to 90° (rod is printed horizontally), the tensile strength increases. In terms of the printing orientation's effect on *E* and *G*, we found that 45° results in minimum values for both Figure 5.

Mechanical Experiment 2 Setup

We followed a similar procedure to Experiment 1 but without extensometers: springs were placed into the load frame grips and stretched at 30mm/minute. To fit the springs into the grips, we added two flat grip plates to the ends of our spring models (Figure 3b). As a spring is stretched, it begins to elastically deform—a state which is reversible. This continues until an elastic limit is reached—the *yield point*—a threshold where the spring is permanently deformed or can even break apart. As before, the load frame software recorded the applied load and the grip displacement at 10Hz. In addition, the software

automatically marked the yield point. From this data, for each spring, we can calculate k using Hooke's Law and derive G from Eq. 1.

Mechanical Experiment 3 Setup

In this experiment, to twist the 3D-printed spring and measure torque, we added a base plate and a socket to the spring, which was connected to a *NEMA 23* stepper motor and a *FUTEK Model TFF400* torque sensor (max 1130 N-mm)—see Figure 3c. We incremented the stepper motor angle by 2.8125° at 1Hz and recorded the torque sensor at 10Hz. Tests concluded when either the spring buckled due to overtwisting or slipping occurred. For analysis, we used readings from the stepper motor, the torque sensor, and a protractor to measure rotation angles (Figure 3c).