A Series of Tubes: Adding Interactivity to 3D Prints Using Internal Pipes

Valkyrie Savage †⋆
valkyrie@eecs.berkeley.edu

Ryan Schmidt †
ryan.schmidt@autodesk.com

Tovi Grossman †
tovi.grossman@autodesk.com

George Fitzmaurice †
george.fitzmaurice@autodesk.com

Björn Hartmann ⋆
bjoern@eecs.berkeley.edu

† **Autodesk Research**
⋆ **UC Berkeley EECS**

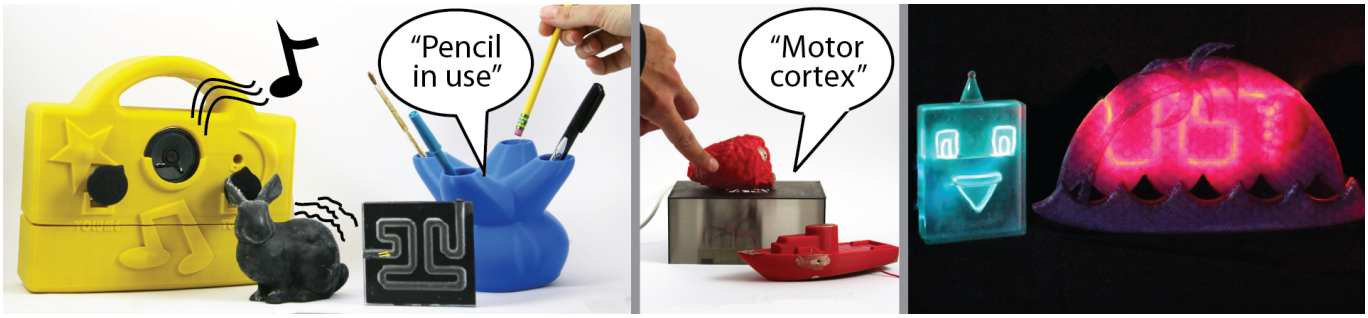

Figure 1. A collection of interactive objects designed using our tool. All have electronic sensing or actuation components routed through their *interior*. Left: a radio, haptic feedback rabbit, maze, and presence-aware pen holder. Center: touch-sensitive toys. Right: custom neon signs.

ABSTRACT

3D printers offer extraordinary flexibility for prototyping the shape and mechanical function of objects. We investigate how 3D models can be modified to facilitate the creation of *interactive* objects offering dynamic input and output. We introduce a general technique to support rapidly prototyping interactivity by removing interior material from 3D models to form internal pipes. We describe the design space of pipes for interaction design, where variables include openings, path constraints, topologies, and inserted media. We then present PipeDream, a tool for routing internal pipes through 3D models, integrated within a 3D modeling program. We use two distinct routing algorithms. The first has users define pipes' terminals and uses path routing and physics-based simulation to minimize pipe bending energy, allowing easy insertion of media post-print. The second lets users supply a desired internal shape to which it fits a pipe route: for this we developed a novel graph-based routing algorithm. We present prototypes created using our tool showing its flexibility and potential.

Author Keywords

Fabrication; 3D Printing; Interactive Objects; Design Tools

ACM Classification Keywords

H.5.2 [User Interfaces (D.2.2, H.1.2, I.3.6)]: Prototyping.

INTRODUCTION

Makers, researchers, and professional designers (jointly "makers" in this paper) increasingly leverage 3D printers as tools for design work. A wide array of objects, from toy figurines to video game controllers and jet engine blades, are now prototyped or even manufactured using these machines. Most models fabricated on 3D printers today are passive: they express the form, but not the interactivity, of an object.

Recently, human-computer interaction researchers have begun to explore how to prototype interactive objects using additive 3D printing processes. Existing approaches, including Printed Optics [30] and Sauron [22], exclusively focus on optical sensing and visual displays — we seek to grow the vocabulary of supported tangible input and output modalities, and we propose using pipes and cavities, allowing designers to integrate secondary materials post-print, for this purpose.

This paper introduces a novel tool that provides path planning and other high-level functionality to support the design of interior pipes and cavities within 3D models. These pipes can be filled, post-print, with a variety of media to enable input, display, and tactile feedback. This *subtractive* approach is complementary to *additive* approaches that inject custom print materials, such as conductors, during the actual printing process [25]. While our approach requires some manual assembly after a print completes, it gives makers the opportunity to cheaply and rapidly prototype a range of interactive objects on 3D printers, including popular single-material fused-deposition modeling machines (colored objects in Figure 1). Our software contributions can also provide useful modeling capabilities for future multi-material printers, e.g., printers that that are able to deposit conductive materials directly, without manual assembly.

We describe a new design space of pipes for the purpose of interaction design, where variables include openings, path constraints, topologies, and inserted media. By exploring this design space, new opportunities for adding interactivity to 3D printed objects can be realized. For example, injecting conductive paint can turn pipes into conductors, which can in turn integrate electronic components into prints; electroluminescent (EL) wire can be threaded through pipes to create computer-controlled visual output; or air can be pumped through pipes to produce haptic effects. Existing sensing and actuation approaches, such as FlyEye [33], swept-frequency capacitive sensing [21], and Jamming User Interfaces [4], can be expanded to new contexts using pipes: makers utilizing such I/O strategies can locate input and output at arbitrary points on a printed 3D object's surface. We leverage these existing techniques for sensing and actuation while our work's novelty is in *internal modeling* for 3D printing and the *redirection* of I/O to or through arbitrary locations on a device.

Figure 1 shows a collection of interactive objects fabricated with the help of internal pipes. These objects can sense touch, detect object presence, generate haptic feedback, or show internal illumination. They could also contain other active and passive electronic components. However, manually modeling appropriate pipes for such functionality is not trivial. For one, many 3D modelling tools do not provide extensive tools for modelling the *interior* of 3D models. Furthermore, careful consideration needs to go into the design and routing of pipes when they are used to add interactivity. Pipes need maximum possible bend radius to ease the insertion of media post-print. Independent pipes cannot intersect, as this could lead to electrical shorting or other problems. Pipes should not pass too close to the surface: consumer-grade 3D printers do not guarantee air- or water-tightness of prints, so more than one printed layer may be required to prevent leakage. Pipes for applications like illuminated EL wire signs must allow a single pipe to follow a specific path (e.g., for writing text).

We contribute PipeDream, an interactive design tool for interior pipes. Our system addresses various challenges makers are likely to encounter, such as avoiding intersections and ensuring maximum bend angles. PipeDream offers 3D point-to-point routing of internal pipes based on physical simulation of flexible rods, and a graph algorithm-based path tracing technique for path-constrained pipes. We present a set of example objects, designed with the help of our tool, that demonstrate different interaction modalities. We conclude by discussing limitations and outlining areas for future exploration.

RELATED WORK

Our contributions relate to other efforts aiming to add interactivity to digitally fabricated objects and to techniques for routing and model modification.

Fabrication of Interactive Devices

Previous work has begun to investigate how to fabricate interactive objects using particular sensing and display technologies. Printed Optics [30] uses light pipes to create integrated displays and senses input optically through components added at print time, while Slyper et al., [26] fabricate flexible robot armatures capable of sensing interaction via changes in air pressure. Our work is inspired by these projects, and contributes a general design tool for modelling objects that can have such capabilities.

Sauron [22] tracks interactions with printed physical mechanisms by placing a camera inside hollow objects, where all components must be visible to it. Our technique instead can work with arbitrary, solid models and only carves pipes for interaction where needed. Midas [23] generates fabrication files for custom capacitive touch sensors. Their technique is limited to planar and unfoldable shapes, as it routes connections on the surface of objects. PipeDream can also be used for touch sensing, but routes conductors *through* objects.

Fabrication with Printed Electronics

Researchers are investigating how to print conductors and place electronic components as part of additive manufacturing. Sells, et al., [25] use a heated syringe to deposit metal in printed 2D circuitboard channels on objects' surfaces. Other experiments done by the RepRap project[1] use plastic filament blended with conductive materials, however the resistance is currently too high to be useful in circuitry. Sarik, et al., [20] use aerosol jetting of conductive material onto printed objects. Park, et al., [17] and Majidi, et al., [10] inject liquid metals, and Hudson [7] provides techniques for embedding conductive thread and other electronics within felted objects. Our approach does not deposit these materials automatically, but creates structures inside a model that make manual post-print injection of conductive ink possible.

Interaction for Fabrication vs. Fabrication for Interaction

HCI researchers are also developing novel interaction techniques for providing input to digital fabrication machines [13, 29, 31], new applications to extend machine capabilities [12, 7], and new hybrid hand tools that combine manual processes with digital fabrication [19, 36]. These projects are complementary as they target different aspects of the design process, though it is conceivable that our pipe design tools be added to AR modeling interfaces like MixFab [29].

Routing and Internal Modeling

Part of our contribution comprises algorithms and techniques for routing tubes through 3D objects. Circuitboard routing algorithms, such as Lee's maze router [8], assume 2D surfaces. Multi-layer boards consist of stacked 2D layers connected by vias. We compute true 3D point-to-point routing though the interior of solid models with flexible paths. This approach has been used in one-off designs [16] but we are not aware of design tools for this purpose.

Some prior work has also focused on modifying interior parts of 3D objects. In Make it Stand [18], material is removed from the interior of objects with the goal of changing their center of gravity. Infrastructs [32] encodes information using interior cavities that can be sensed using terahertz imaging. Bickel, et al., [1] enforced deformation behavior with multi-material blending inside objects. Modeling internal chambers is one aspect of our work, however our focus is interaction rather than balance, identification, or deformation.

[1] `http://reprap.org`

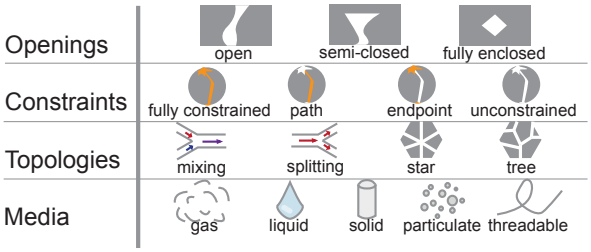

Figure 2. The design space of pipes. Pipe *openings*, *path constraints*, *topologies*, and inserted *media* give rise to unique input/output possibilities.

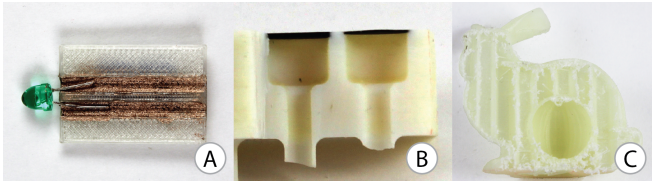

Figure 3. Pipe openings: a) shows a pair of *open* pipes filled with copper paint to power an LED. b) shows two *semi-closed* pipes capped by rubber membranes. In c), a rabbit has a *fully enclosed* chamber to modify its weight.

THE DESIGN SPACE OF PIPES

Our review of prior art indicates the potential benefits of using interior pipes for adding interactivity to 3D printed objects, although no general-purpose tools currently exist for this purpose. To guide the design of such a tool, we now provide a design space of the types of pipes that a maker may want to utilize. We have identified four relevant dimensions of pipe design: *openings*, *path constraints*, *topologies*, and inserted *media*. We describe the space in Figure 2.

Openings

Pipes inside solid objects can be either *open* to the outside on both ends, *semi-closed*, or *fully enclosed* (Figure 2). For interactive devices, openings can be either user-facing, where a user would, for example, touch a model; or system-facing, where sensors and actuators may be connected.

Open pipes may be used to create electronic circuitry. For example, an open pipe filled with conductive paint and connected to a battery can power an LED (Figure 3a). *Semi-closed* pipes are capped on one end and can be used to create tactile output. By fabricating a cap from a malleable material and attaching a pump to the open end, the flexible cap can be raised and depressed (Figure 3b). *Fully enclosed* pipes (cavities) are entirely internal. They may be used as weight modifiers [18] and for object identification [32] (Figure 3c). If printed in parts and assembled post-print, enclosed pipes could hold water or particles that would otherwise fall out.

Path Constraints

Depending on the purpose of the pipe, its geometry may be constrained to specific locations within the 3D model. An *endpoint-constrained* pipe requires specific start and end points, but the particular route of the internal path is irrelevant. This is the case for circuit schematics, where the electronic components have fixed locations, but many routing solutions are valid. In contrast, for a *path-constrained* pipe the shape of the path is important. For example, threading electroluminescent (EL) wire through a clear pipe enables makers to emulate neon art (Figure 1, right). A *fully constrained* pipe requires specific terminals as well as a specific path. If a maker wishes the EL wire to exit the rear of the sign to hide the electronics, she can use a fully-constrained path (Figure 1, right). Pipes may also be fully *unconstrained*, however in this paper we focus on contexts where designers deliberately place constraints on pipe endpoints or path.

Pipe Topologies

Pipe network topologies enable *splitting* or *mixing* of media (Figure 2). *Star* and *tree* topologies extend splitting and mixing primitives, encompassing multiple-in and multiple-out configurations; we use star topologies to create sensing locations for our touch-sensitive toys, with one terminal used for swept-frequency capacitive sensing (Figure 1, center).

Media in Pipes

After printing, different media can be inserted into pipes to change affordances and capabilities. We consider *gas*, *liquids*, *solids*, *particulates*, and *threadables*. Compressible fluids (*"gas"*) and incompressible fluids (*"liquids"*) inside semi-closed pipes can create haptic feedback; pressure can also be sensed to act as an input [27]. Gases can be used as carriers for scents or fog. Conductive liquids like copper paint, when dried, turn pipes into conductive paths and can also fix electronic components in place. Filling pipes with *solids* at print time enables designers to exploit the difference in material properties between pipe and contents. For example, in Printed Optics [30], different refractive indices between solid cladding material and transparent material inside pipes allow light transport along pipes for input and output. *Particulates*, either printed in-place or inserted, can be of varying densities and can contain particles of various and variable size. A single large particle can be used for display. Small particles agitated by air can generate sound. *Threadable* media, such as regular wire, electroluminescent wire, or commercial fiberoptic cables, can be threaded through pipes post-printing. Threading such media enables addition of conduction, illumination, or optical sensing using materials that can easily be procured, but cannot themselves be printed, e.g., on consumer printers with a single material.

In the next section we detail how different pipes within this design space space can be created with the help of our new design tool, PipeDream.

PipeDream - A TOOL FOR PIPE DESIGN

To allow makers to design novel objects with pipe-powered interfaces, we created a tool, PipeDream, to add pipe geometries to arbitrary 3D models. Pipe design is non-trivial. Independent pipes should not intersect inside an object lest their contents be unintentionally mixed. To permit easy insertion of media after printing, pipes should have smooth bends. Pipes, in general, need to avoid objects' surfaces to prevent

fluid leakage when printed by hobbyist-grade machines. Finally, for designing path-constrained pipes that follow a user-specified path, we must ensure certain characteristics of that path (e.g., that it is connected).

PipeDream allows users to design pipes in two ways. In one, they select exterior anchor points on their objects; the tool then creates complete point-to-point routings using A* path search and physical rod simulation (Figure 4). In the other, users import vector art describing desired interior paths (Figure 8). PipeDream , using edge graph manipulation and Euler tour generation, creates a single path that follows the input shape. We thicken these routes to create pipes. The resultant pipes are subtracted from the user's mesh via voxelization and remeshing. Our tool is implemented in C++ as an extension to Meshmixer, a consumer 3D mesh editing tool [24].

Exterior Terminals

To create endpoint-constrained pipes, in which the location of exterior connection points matters, a brush tool is used to specify a pair of points on the mesh's surface (Figure 4a). The system automatically routes the pipe, and displays a preview. Using sliders, the user can independently adjust the starting and ending radii of the pipe. This is useful for, e.g, our breathing bunny (Figure 11b), whose pipes must fit a piece of hardware on one end, but on the other end should have as large an interactive surface as possible.

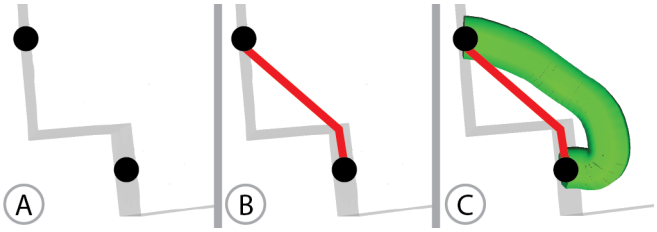

Figure 4. In a), an example mesh with exterior connection points selected. The smoothed A* routing (b) is drawn in red. In green is our physically-based rod after simulation (c).

Routing Algorithm and Physical Simulation

Given two points, we first find a path between them using the A* search algorithm on a voxel representation of the object [6]. A* finds a least-cost path through a graph of nodes given a cost function. Our graph comprises the grid of voxels as nodes, with edges between adjacent nodes; our path cost is based on the Euclidean distance to the selected end point. In this formulation, the routed path is naturally constrained to stay within the object boundaries (Figure 4b).

Our A* pathfinding produces the *shortest* path, but in many cases we require a smoother path to aid insertion of the desired medium, such as a threadable wire. In addition, the distance from the surface is not taken into account in the A* search and pipes may thus be routed along surface contours, which may cause leakage or structural deficiencies.

We solve this problem via physical simulation. We create a straight virtual wire (a 3D polyline or *discrete rod*), and then set its initial position as our A* path. Running the physical simulation allows the wire to attempt to return to a straight configuration, while satisfying the user-selected endpoint position constraints. In addition to user constraints, we require that the rod be perpendicular to the surface at its ends to prevent the terminals' shape from deforming (Figure 4c).

The wire is modelled as a *discrete rod*: a 3D poly-line with a bending constraint between adjacent segments. Our simulation is based on an implementation of Position-Based Dynamics (PBD) [14], with all constraints modelled as penalty forces. To keep the rod inside the mesh, we compute a discretized distance field, then constrain the rod to stay within our offset shell using a simple penalty force. This approach can successfully route pipes through complex geometries (Figure 5).

PBD simulations rarely converge to a steady state, so we simply halt after a fixed number of timesteps (250). Our voxel grid currently uses a resolution of 128x128x128 voxels, which enables interactive operations, but this parameter can be adjusted for different model requirements. The simulation takes about a second for our examples.

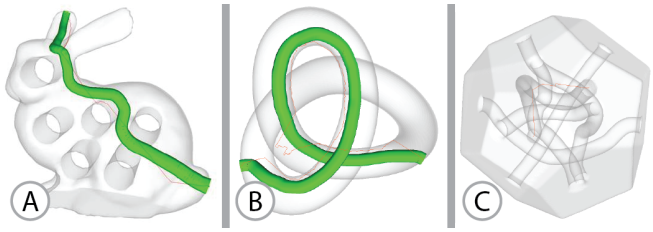

Figure 5. A set of example routings produced by our tool. Note that there is a tradeoff between bending and length of pipes: to reduce bending, pipes may be routed through longer paths around holes in genus > 0 meshes (a). Pipes successfully route through long, curving paths in knots (b), and multiple pipes placed in a single mesh in a tangled configuration successfully avoid each other (c).

Multiple Pipes

Pipes are routed and cut out of the mesh serially. This prevents pipe intersections: once a pipe has been cut out of the mesh, it cannot be part of valid routes in any future pipe routing operation (Figure 5c). After solving for a path, we add it as a constraint to the system, generating a penalty force that pushes additional paths away from it. However, we do not currently track all pipes for global optimization: we greedily select the best routing per pipe.

Pipes with Multiple Endpoints

To allow designers to create pipes with multiple endpoints (e.g., the star topologies as in Figure 2), we use one surface position per pipe and a central fixed endpoint that the user can interactively position (Figure 6). The user can connect several pipes, hitting a key to add each in sequence and positioning them with the mouse. The pipes route as normal.

Interior Paths

For path-constrained pipes, the route of the pipe must conform to a desired shape. A maker can import a vector graphics file (such as SVG) describing the path she wants for her pipes: all lines in this input file are interpreted as desired pipe components for the final artifact. Using sliders, she can change the pipes' radius (Figure 7). A preview is provided to show

the radius and location of the cut, and the maker can align her model as appropriate.

In some cases, such as for the neon route in Figure 8 and resultant sign in Figure 9, a single path must pass through a set of disconnected path segments (i.e., all letters in the word "UIST"). PipeDream leverages graph theory algorithms to generate this single path, as described below. Without endpoint constraints, the path's exit(s) from the model are defined by the SVG: the maker can see where the path exits the model and translate the cut according to her desired outcome.

Routing

Our technique for routing this path is inspired by work in generating continuous line illustrations from images [2, 34]. Our routing problem is a version of the Chinese Postman Problem [11], in which we wish to traverse all edges of the graph described by the user's input, much as a postman needs to walk along every road at least once to deliver mail. In our relaxation, we allow the creation of new edges (i.e., the postman may jump from building to building in addition to using existing roads).

Setup: We first add a temporary edge that connects the user's desired start and end points, and create a full Eulerian graph. This edge is removed at the end to yield a semi-Eulerian graph that can be threaded after printing.

Connect disconnected components: To connect disconnected subgraphs, we find minimum Euclidean distance vertex pairs spanning two subgraphs, and greedily add edges to such pairs until all subgraphs are joined into a single graph.

Eulerization: In order to make our graph Eulerian, we consider all odd-degree vertices in the connected graph and make a clique of potential edges based on distance. We greedily add edges between odd nodes until no odd nodes remain. Finally, we remove our temporary edge, changing the graph from Eulerian to semi-Eulerian. This is a connected, semi-Eulerian graph that contains all edges in the input. A lower total weight matching is possible by connecting components and ensuring node evenness together in a global process, as well as by using minimum-weight matching rather than greedy selection, however this optimization is not crucial for our purposes.

Euler Tour: To find an Euler tour, we use Fleury's algorithm [3] with a modification to preserve continuity: at a given node, from multiple candidate edges, we prefer the edge(s) with the smallest angular deviation from the incoming edge (i.e., we prefer to pass straight through a node, if possible). This minimizes turns in the final artifact, which eases support material removal and assembly. Our EL wire in Figure 9 follows an Euler tour through the input nodes.

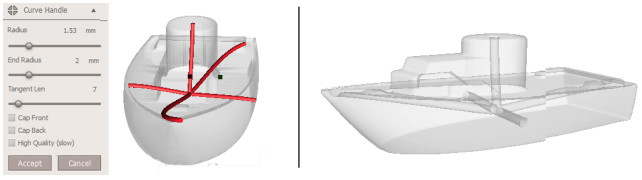

Figure 6. Left: Users can interactively position connection points for pipe topologies; they can also adjust pipe diameter and rod parameters. Right: Resulting model with cut pipes.

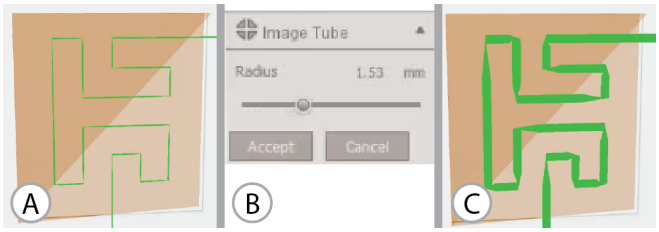

Figure 7. Users can interactively change the radius of their pipes using a slider (b). PipeDream's live preview shows the real size and location of the cut (a, c) before it is committed to the mesh.

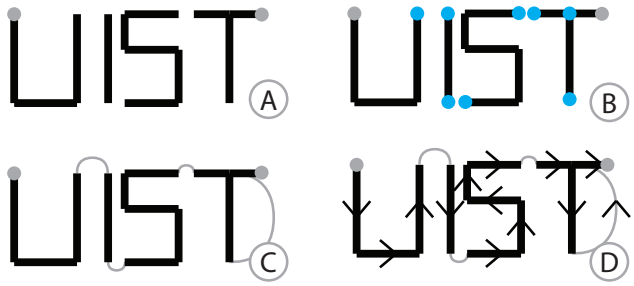

Figure 8. An input vector graphics file with the start and end points highlighted in gray (a) and the points which cannot be tubed as drawn highlighted in blue (b). The connected graph created by our software (c) and the resulting Euler tour (d) permit insertion of media post-print.

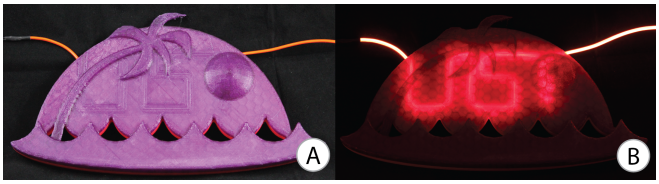

Figure 9. Our neon sign built using the process above. (a) shows the unlit sign, and (b) shows it lit.

Edges that Intersect in the Plane

We do not currently redirect edges that intersect in the plane. Because we have a 3-dimensional canvas in which to work, it would be possible to push overlapping paths into the third dimension. However, we leave this to future work. For our specific applications so far, it has been unnecessary.

Separating at the Pipe Plane

We allow users the choice to separate their meshes at the pipe plane, which may be necessary for support material removal (discussed later) or insertion of media: e.g., EL wire can be bent much more sharply by hand in the plane than it can by pushing from one end into a pipe. To do this, we simply perform a plane cut after the mesh modification operation described below.

Mesh Modification

After the maker hits "Accept" on her preview, we must cut pipes out of the input mesh. At this stage, we verify whether the new pipe will intersect the surface or an existing pipe, and if so display a warning message suggesting that she should

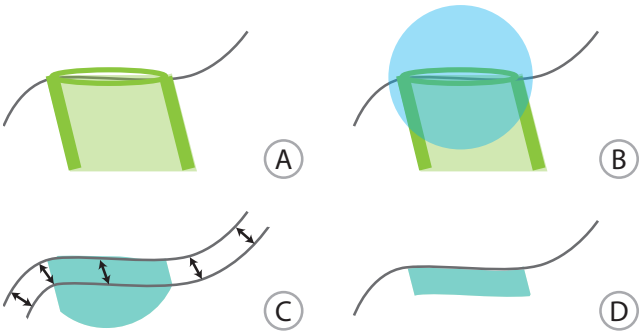

Figure 10. The process to "cap" the end of a pipe. In (a), a pipe end is placed on the surface of the mesh. In (b), we create a sphere whose radius is slightly larger than the radius of the pipe, centred at the pipe's endpoint. In (c), we consider the intersection of the sphere and the pipe, and create an offset surface of the mesh. Finally, in (d) we intersect with the offset surface. This is kept as a cap for the pipe in mesh subtraction.

consider a smaller radius. Our basic strategy for the cut is to create a polygonal tube by sweeping a profile polyline along the path, and subtract the tube from the mesh. We only use circular profiles in our examples, but clearly any other profile could be used. We allow different start and end radii, selected by the maker in the previous step, which are linearly interpolated along the path.

To avoid the complexities of direct mesh booleans and give us additional flexibility, we construct a discretized volumetric (i.e., voxel-based) representation of the original mesh and the tube mesh. Boolean subtraction is trivial in such representations. We specifically use narrow-band level sets [15], which represent the surface as the zero iso-contour of an approximate distance field. We then use marching cubes [9] to produce the final mesh for printing. This strategy does incur some resampling artifacts, however so does the 3D printing; compared to the print process, the time required to voxelize at high resolution is inconsequential. Since we have a distance field, we can easily introduce useful effects, such as the ability to add a thin membrane "cap" over the end of a tube. This is accomplished by applying a standard "thicken" operation to the level set version of the initial mesh, intersecting with a sphere placed at the tube endpoint, and then performing a boolean union with the initial shape-minus-tube result (Figure 10). A straightforward extension would be to add or subtract additional elements at the tube endpoints.

FABRICATION TECHNIQUES

Several pragmatic concerns arise when fabricating models with pipes and cavities. Pipes frequently lead to overhangs (where an underlying supporting layer has a smaller footprint than the subsequent layer). To allow overhangs, 3D printers may lay down support material that must be later removed from the model. The removal process can be time-intensive and challenging. The physical fabrication process used by different machines gives rise to differing strategies for avoiding support material or easing its removal. Techniques differ for fused deposition modeling (FDM) machines, such as the Makerbot[2], polyjet machines such as Stratasys Connex[3], powder-based machines (e.g., ZCorp[4]), and those that use stereolithography (e.g., Form 1[5]).

Support Types

The physical form of support material varies between printer types. On single-material hobbyist FDM machines, the support is created from the same type of material as the model, laid down in a different pattern to ease detachment. Multi-material FDM machines typically feature dissolvable support material: by submersing a part in lye, one can eventually remove support material while leaving the model material behind. Polyjet machines also use a secondary type of material for support: it can be melted out in some cases (e.g., ProJet wax material) or blasted out with high-pressure water. This blasting process requires fairly open access to the support material: the robot head and maze in Figure 1 were cut into two pieces along the plane of the pipes to give better access to the support. For powder-based and stereolithography machines, the material (powder or photo-cure liquid) that is not used for the model remains in the bed and forms the support structure — this is uniquely well-suited to removal, as it can simply be poured out after a print is completed. We hope to experiment with these types of printers more in the future.

Model Orientation and Pipe Geometries

The various types of printers permit different sorts of support-free prints. FDM allows approximately a $45°$ overhang before support is required. For polyjet machines, overhangs can be just $14°$. Reorienting a part within the bed can lead to more or less support required. MeshMixer already has a built-in tool that reorients models to keep as much surface area as possible within the $45°$ constraint imposed by FDM printers, and alternative pipe cross-section geometries (e.g., teardrop) could alleviate the need for support material to bridge the tops of pipes. In the future, pipe routes could be optimized to minimize support material required.

Model Assembly

For especially complex geometries, models can be "cut up" into multiple pieces that can be assembled, avoiding support deposition or easing material removal; this can also ease insertion of the desired medium. This is somewhat undesirable as it means more time spent in assembly post-print, and additionally a cut pipe may no longer be fluid-tight. Our plane cut option offers a simple way to slice a model; more sophisticated techniques may be necessary for internal paths that are not constrained to a plane.

TOTALLY TUBULAR EXAMPLE OBJECTS

To evaluate and highlight our tool's capabilities, we fabricated a set of six prototype objects designed using PipeDream. To show that our approach is printer-independent, we fabricated some on a Makerbot Replicator

[2] `http://makerbot.com`
[3] `http://stratasys.com`
[4] `http://3dsystems.com`
[5] `http://formlabs.com`

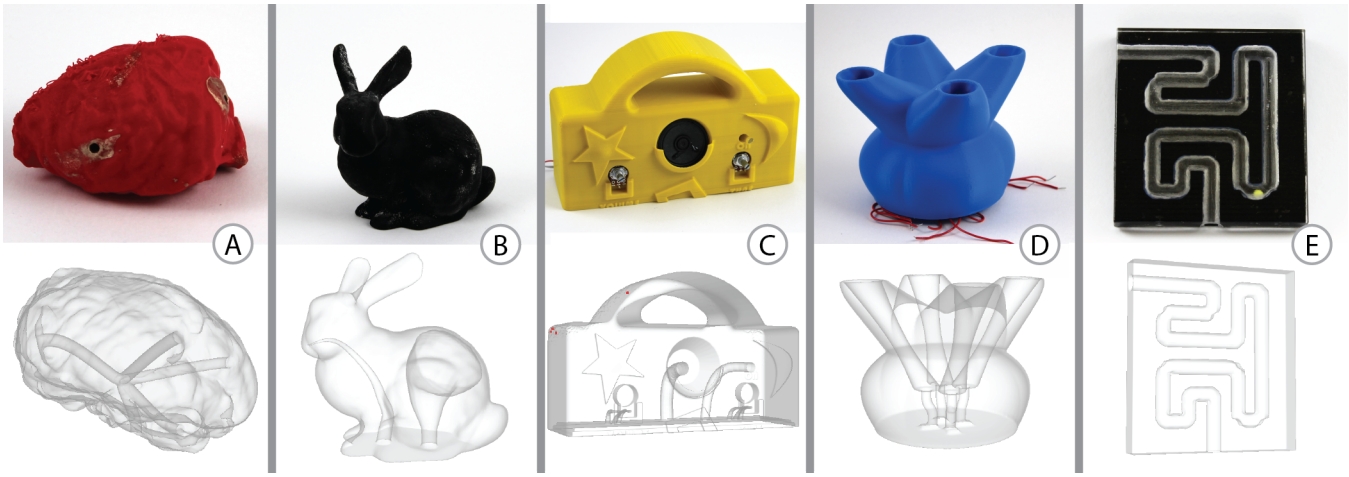

Figure 11. A series of example objects created using our system. (a) is a touch-sensitive brain toy. (b) is a rabbit which "breathes" using wind pipes we built. (c) shows a portable radio. (d) shows a presence-aware pen holder. (e) is a maze game.

2 and the rest on a Stratasys Connex 260. Each prototype object explores the expansion of an existing sensing or actuation technique to the surface of 3D printed objects.

Touch-sensitive Toys (open, endpoint, star, liquid)

To expand Touché [21] to 3D printed forms, we created a set of touch-sensitive toys and a companion app that identifies the objects as well as touched parts of the objects (inspired by the boat application in Acoustic Barcodes [5]). The objects — a brain and boat, in our example — can be set on a base, upon which the computer displays "brain" or "boat". Touching the brain model's front protrusion yields the announcement "olfactory bulb". The distinct touch points on each object are connected by an interior star topology, and touch sensing is performed via a single conductive connection to the base, using swept-frequency capacitive sensing (see Figure 11a). Since each toy and each gesture has a distinct capacitive signature, we use a simple classifier trained to detect both toy and gesture based on profile.

The model we used for the boat was downloaded from Thingiverse, while the brain model came from researchers at one of our instutions. We then used PipeDream to create the desired internal pipes. All components were fabricated on a Makerbot, support-free. After printing, we injected copper paint (CuPro-Cote) into the interior pipes using a craft syringe. This paint requires approximately 2 days to fully dry in this configuration (3mm diameter tubes, maximum tube length 10cm). The resistance of the paint with this cross-section is approximately $2\Omega/inch$. Our smart base is powered by an Arduino Uno running open-source SFCS code[6].

Breathing Bunny (semi-closed, endpoint, gas)

With inspiration from PneUIs [35] for using gases to both provide sensation to the user through openings and deform a model internally, we created a rabbit with a pair of tubes that can simulate breathing (see Figure 11b). When the rabbit inhales through its nose, its abdomen rises, and as it exhales its abdomen falls. For this, we used a combination air/vacuum pump (SparkFun ROB-10398): one terminal creates a vacuum while the other creates positive pressure. The rabbit model, the well-known Stanford Bunny[7], was downloaded from Thingiverse. We then used PipeDream to add two pipes: one open pipe exiting at its mouth, and one semi-closed pipe capped at its abdomen. We connected one pipe to each of our pump's terminals, and using a programmable power supply we mimic a rabbit's breathing pattern. This example was printed on the Connex using flexible material (100% TangoBlackPlus); support was flushed post-print.

Custom Radio (open, endpoint, threadable)

A custom radio (Figure 11c) shows how pipes can be used to integrate electronic components into the user-facing surface of an object. The radio includes a power and volume dial, and LED which goes on when the radio is on, and a tuner dial. An Arduino Uno microcontroller, Si4703 FM tuner, and 6V (4xAA) power supply are embedded in the base of the radio. The sound is played through a single .5 Watt speaker.

The radio shell was modelled in a commercial CAD tool. We included slots on the 3D model matching the shape of the electronic components that were to be embedded. The model was then imported into PipeDream, where we created a network of 8 individual open pipes (3 per potentiometer, 1 for the LED, 1 for the speaker) connecting the components to the radio's base. We fabricated the radio body on our Makerbot, support-free but cut in two pieces to allow the Arduino and battery pack to be inserted. We then threaded the pipes with wires to connect the components to the microcontroller contained in the base.

Presence-aware Pen Holder (open, endpoint, threadable)

Our presence-aware pen holder distinguishes which tool or tools a user has picked up (see Figure 11d). We use a modification of the FlyEye technique described by Wimmer [33] in which an IR camera senses a user's grasp on an object, getting data from light transmitted on fiber optic cables.

[6] `http://www.instructables.com/id/Touche-for-Arduino-Advanced-touch-sensing`

[7] `http://en.wikipedia.org/wiki/Stanford_bunny`

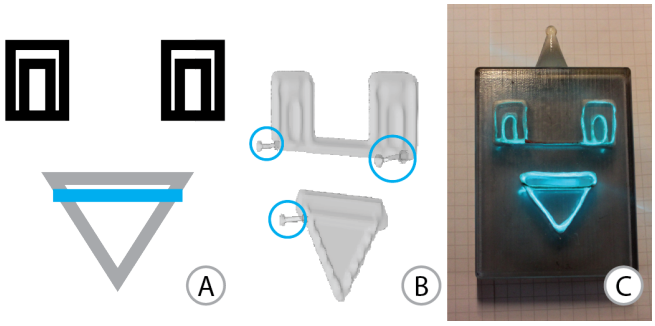

Figure 12. A two-state animated neon sign designed using our tool. (a) shows the input SVG files, with the eyes (non-animated) in black and the two states of the mouth in blue and gray. (b) shows the tubes we subtracted from the mesh, rear exit tubes circled in blue. In (c), we show the fabricated sign.

The penholder body was created in a commercial CAD tool. We used PipeDream to add four internal open pipes connecting each pen chamber to the base of the pen holder. This prototype was built on our Makerbot support-free. Each pipe was then threaded with Sparkfun clear core light pipe (PRT-10697), similar to fiber optic cable. We use a single cable per pipe; our 6mm diameter light pipe, in comparison to the fine fiber optics used in the original work, can emit and receive through the same cable because the cross-sectional area is large enough to accommodate both functions. At the base of each pipe is a QRE1113 line sensor breakout board, with integrated IR emitter and receiver. When a pen is in its appointed place, the emitted infrared light is reflected off its base and travels back to the receiver, where it appears bright.

Maze (fully enclosed, path, tree, particulate)

We created a maze game with a marble that can be navigated through a maze-like path, with exits on either side of the model (Figure 11e).

The maze is based on an SVG file we manually created and processed with our internal path routing tool. We printed the maze on the Connex using a transparent material (100% VeroClear) for the top and a black material (100% TangoBlackPlus) for the bottom. The final artifact was fabricated in two halves fastened together via glue, to allow removal of support material with a high-pressure water tool. We plugged the ends of the pipes post-print with additional material to ensure the ball does not escape.

Animated Neon Sign (open, fully constrained, threadable)

Neon art is traditionally made from hand-formed glass tubes containing neon gas. The tubes light up when a current is passed through them. For this type of art, the path of the tubes is of crucial importance, as it determines how the sign will look. We designed a custom neon robot head which can be animated to "talk". The external geometry of the model was imported into PipeDream, which created the associated pipes using its internal path routing tool. We added additional endpoint-constrained tubes as exits out the rear of the model to allow us to hide the controls (see Figure 12b). We then printed the model on the Connex using transparent material. The model was printed in two halves, cutting through the plane of the internal path, to facilitate assembly.

The pipes are threaded with electroluminescent wire which is lit in sequence, using a sequencer, to create the animation.

DISCUSSION

The promise of 3D printing is two-fold: it enables the manufacturing of complex objects at low volumes, and it allows for the manufacturing of geometries that are impossible to create using traditional techniques like injection molding. Printing pipes to prototype interactive objects exploits both advantages. Prototypes by definition are low-volume parts, and pipes require complex interior geometries that are very difficult or impossible to produce using other processes. Our work has focused on routing pipes that are manually filled after a print has completed. However, the fundamental operations of 3D path routing and topology specification that our tool offers will remain relevant as 3D printing technology advances. Multi-material printers are already available; as they advance and support the embedding of additional materials (e.g., conductors), PipeDream can enable designers to efficiently specify where different functional materials should be placed at design time.

Limitations and Future Work

While pipes provide a powerful mechanism for directing interaction to arbitrary locations on an object's surface, they do have shortcomings which warrant discussion. In particular, they cannot replace the active elements required for a functioning interactive device: they simply redirect sensing to different locations on an object's surface. All pipes need to, at some point, connect to a sensor, microcontroller, or pump.

Some media are difficult to insert in pipes, or may not be compatible with all types of pipes. Threadable materials in particular can provide challenges, as they can get stuck while being fed into pipes. Our system mitigates this by minimizing the bend energy of the interior pipes, but for complex pipe geometries threading could be a challenge.

Fluids also raise unique challenges. For example, the conductive paint we used in our designs can take multiple days to dry in long pipes. Liquids kept in the pipes can also cause discoloration due to seepage. Furthermore, not all printers guarantee watertightness of their prints, so leakage can occur.

As discussed, removal of support material can also be a challenge. Particularly on multi-jetting printers, printing support-free is not possible for most geometries, and detaching printed support is not always trivial. In some cases we printed our models in several pieces, to avoid this problem.

There are also areas for improvement and future work with our software tool. The design of PipeDream was based on our proposed design space of pipes. However, the design tool is not yet fully capable of realizing all possibilities in this design space. Specifically, we did not explicitly explore the mixing or splitting of pipes, and our path-constrained cuts are strictly 2D for now. A 2.5D SVG import using, e.g., line colors as height would be straightforward, and our described algorithm

would continue to work properly. An extension to PipeDream could include detection of where the 2D path intersects itself; intersecting pipe components could be "pushed into 3D" using the discrete rods from our endpoint-constrained routing algorithm. Furthermore, since pipes are committed to the mesh one at a time, our tool does not currently optimize networks of multiple pipes. Instead, we utilize a greedy algorithm and route a single pipe at a time. This suggests the need for a better general 3D routing algorithm with multiple traces, which could take into account global bending energy, global pipe length, or other task-specific requirements. For integration with common electronic components, it would be possible to build a library of "footprints" which could be used for recessing them into printed objects' surfaces; such footprints could also be used for automatically generating connecting pipes based on an electrical connection graph, as in circuitboard routing tools like Eagle[8]. Future work should also include assisting makers in the fabrication process itself by automatically cutting objects to reduce support material removal issues.

The basic approach of using a discrete rod for physical simulation does have various limitations, in particular it cannot recover if the rod becomes tangled or unstable. However we have observed these problems only when attempting to break the system; they did not occur in any of the cases shown in our figures. Another issue is parameters, as the penalty forces used in PBD do not correspond to physical units. We avoided serious issues by uniformly scaling the initial mesh to a unit box, and tuning the parameters on several test cases.

We are also eager to test our tool with users and get feedback from makers who use it in the wild. Based on our own experiences, we believe our system could be readily utilized by makers and researchers, and serve as a useful tool for rapidly prototyping interactive 3D objects. We plan to release PipeDream as a part of Meshmixer later this year.

CONCLUSION

Pipe mediation opens new interactive device prototyping opportunities both for makers with consumer-grade 3D printers and research and industrial labs with higher-end machines. While today's 3D printers are not yet able to fabricate active components in-place, we suggest that 3D printed interactive devices can be created with redirection of active input and output via pipes. This amplifies and extends the utility of existing sensing and actuation approaches. In addition to describing the design space of pipes, we also presented PipeDream, a design tool for the creation of a series of tubes [28] inside arbitrary 3D models, and discussed many techniques that combine nicely with pipes. We fabricated several example objects using our tool to explore new points in the design space. We look forward to extending our software contribution into future multi-material 3D modelling tools.

ACKNOWLEDGEMENTS

The authors would like to thank Karl D. D. Willis for his incredibly broad knowledge of paper references and Tim Campbell for his assistance creating Instructables. This work is part of a line of research on digital fabrication that is supported in part by the NSF GRFP under grant DGE 1106400 and a Sloan Fellowship.

[8] `http://www.cadsoftusa.com`

REFERENCES

1. Bickel, B., Bächer, M., Otaduy, M. A., Lee, H. R., Pfister, H., Gross, M., and Matusik, W. Design and fabrication of materials with desired deformation behavior. In *SIGGRAPH '10*, 63:1–63:10.
2. Bosch, R., and Herman, A. Continuous line drawings via the traveling salesman problem. *Oper. Res. Lett. 32*, 4 (July 2004), 302–303.
3. Fleury, M. Deux problèmes de géométrie de situation. *Journal de mathématiques élémentaires 2*, 2 (1883), 257–261.
4. Follmer, S., Leithinger, D., Olwal, A., Cheng, N., and Ishii, H. Jamming user interfaces: Programmable particle stiffness and sensing for malleable and shape-changing devices. In *Proc. UIST '12*, 519–528.
5. Harrison, C., Xiao, R., and Hudson, S. Acoustic barcodes: Passive, durable and inexpensive notched identification tags. In *Proc. UIST '12*, 563–568.
6. Hart, P., Nilsson, N., and Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics 4*, 3 (July 1968), 100–107.
7. Hudson, S. E. Printing teddy bears: A technique for 3d printing of soft interactive objects. In *Proc. CHI '14*, ACM (2014), 459–468.
8. Lee, C. An algorithm for path connections and its applications. *IRE Transactions on Electronic Computers 10*, 2 (1961), 346–365.
9. Lorensen, W. E., and Cline, H. E. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics 21*, 4 (July 1987), 163–169.
10. Majidi, C., Kramer, R., and Wood, R. J. A non-differential elastomer curvature sensor for softer-than-skin electronics. *Smart Materials and Structures 20*, 10 (2011), 105017.
11. Mei-Ko, K. Graphic programming using odd or even points. *Chinese Math. 1* (1962), 273–277.
12. Mueller, S., Kruck, B., and Baudisch, P. Laserorigami: Laser-cutting 3d objects. In *Proc. CHI '13*, 2585–2592.
13. Mueller, S., Lopes, P., and Baudisch, P. Interactive construction: Interactive fabrication of functional mechanical devices. In *Proc. UIST '12*, 599–606.
14. Müller, M., Heidelberger, B., Hennix, M., and Ratcliff, J. Position based dynamics. *J. Vis. Comun. Image Represent. 18*, 2 (Apr. 2007), 109–118.
15. Museth, K., Breen, D. E., Whitaker, R. T., Mauch, S., and Johnson, D. Algorithms for interactive editing of level set models. *Comp. Graph. Forum 24*, 4 (2005), 821–841.

16. Navarrete, M., Lopes, A., Acuna, J., Estrada, R., MacDonald, E., Palmer, J., and Wicker, R. Integrated layered manufacturing of a novel wireless motion sensor system with GPS. *Technical Report, University of Texas at El Paso* (2007).

17. Park, Y.-L., Chen, B.-R., and Wood, R. J. Design and fabrication of soft artificial skin using embedded microchannels and liquid conductors. *IEEE Sensors Journal 12*, 8 (August 2012), 2711–2718.

18. Prévost, R., Whiting, E., Lefebvre, S., and Sorkine-Hornung, O. Make it stand: Balancing shapes for 3D fabrication. *ACM Trans. Graph. 32*, 4 (July 2013), 81:1–81:10.

19. Rivers, A., Moyer, I. E., and Durand, F. Position-correcting tools for 2D digital fabrication. *ACM Trans. Graph. 31*, 4 (2012), 88.

20. Sarik, J., Butler, A., Villar, N., Scott, J., and Hodges, S. Fabricating electronics with rapid prototyping tools. In *Adj. Proc. TEI 2012*, 339–342.

21. Sato, M., Poupyrev, I., and Harrison, C. Touché: Enhancing touch interaction on humans, screens, liquids, and everyday objects. In *Proc. CHI '12*, 483–492.

22. Savage, V., Chang, C., and Hartmann, B. Sauron: Embedded single-camera sensing of printed physical user interfaces. In *Proc. UIST '13*, 447–456.

23. Savage, V., Zhang, X., and Hartmann, B. Midas: Fabricating custom capacitive touch sensors to prototype interactive objects. In *Proc. UIST '12*, 579–588.

24. Schmidt, R., and Singh, K. Meshmixer: An interface for rapid mesh composition. In *SIGGRAPH '10 Talks*, 6:1.

25. Sells, E. Rapid prototyped electronic circuits. `http://fennetic.net/irc/reprap_circuits.pdf`, 2004.

26. Slyper, R., and Hodgins, J. Prototyping robot appearance, movement, and interactions using flexible 3D printing and air pressure sensors. *IEEE Xplore* (2012).

27. Slyper, R., Poupyrev, I., and Hodgins, J. Sensing through structure: Designing soft silicone sensors. In *Proc. TEI '11*, 213–220.

28. Stevens, T. Full Committee Markup - Communications Reform Bill. `http://www.commerce.senate.gov`, June 2006.

29. Weichel, C., Lau, M., Kim, D., Villar, N., and Gellersen, H. Mixfab: A mixed-reality environment for personal fabrication. In *Proc. CHI '14*, ACM (2014), 3855–3864.

30. Willis, K., Brockmeyer, E., Hudson, S., and Poupyrev, I. Printed optics: 3D printing of embedded optical elements for interactive devices. In *Proc. UIST '12*, 589–598.

31. Willis, K. D., Xu, C., Wu, K.-J., Levin, G., and Gross, M. D. Interactive fabrication: New interfaces for digital fabrication. In *Proc. TEI '11*, 69–72.

32. Willis, K. D. D., and Wilson, A. D. InfraStructs: fabricating information inside physical objects for imaging in the terahertz region. *ACM Trans. Graph. 32*, 4, 138:1–138:10.

33. Wimmer, R. FlyEye: Grasp-sensitive surfaces using optical fiber. In *Proc. TEI '10*, 245–248.

34. Wong, F. J., and Takahashi, S. A graph-based approach to continuous line illustrations with variable levels of detail. *Computer Graphics Forum 30*, 7 (Nov. 2011), 1931–1939.

35. Yao, L., Niiyama, R., Ou, J., Follmer, S., Della Silva, C., and Ishii, H. PneUI: pneumatically actuated soft composite materials for shape changing interfaces. In *Proc. UIST '13*, 13–22.

36. Zoran, A., and Paradiso, J. A. FreeD: a freehand digital sculpting tool. In *Proc. CHI '13*, 2613–2616.