On-The-Fly Print: Incremental Printing While Modeling

Huaishu Peng, Rundong Wu, Steve Marschner, François Guimbretière
Computing and Information Science
Cornell University
{hp356, rw489}@cornell.edu, {srm, francois}@cs.cornell.edu

ABSTRACT

Current interactive fabrication tools offer tangible feedback by allowing users to work directly on the physical model, but they are slow because users need to participate in the physical instantiation of their designs. In contrast, CAD software offers powerful tools for 3D modeling but delays access to the physical workpiece until the end of the design process.

In this paper we propose *On-the-Fly Print*: a 3D modeling approach that allows the user to design 3D models digitally while having a low-fidelity physical wireframe model printed in parallel. Our software starts printing features as soon as they are created and updates the physical model as needed. Users can quickly check the design in a real usage context by removing the partial physical print from the printer and replacing it afterwards to continue printing. Digital content modification can be updated with quick physical correction using a retractable cutting blade. We present the detailed description of *On-the-Fly Print* and showcase several examples designed and printed with our system.

Author Keywords
3D printing; fabrication; computational craft; CAD; rapid prototyping; interactive devices.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): User Interfaces.

INTRODUCTION

Since the notion of interactive fabrication was introduced by Willis et al. [32], several approaches have been proposed for hands-on digital fabrication. For example, Constructable [17] allows the step-by-step fabrication of functional objects using a laser cutter controlled by a laser pointer; D-Coil [19] enables non-experts to design 3D digital models from scratch using a digitally controlled wax extruder; ReForm [31] merges manual shaping with digital milling and extrusion of synthetic clay. On the one hand, these interactive fabrication systems offer immediate, tangible feedback that can benefit

CHI'16, May 07-12, 2016, San Jose, CA, USA

DOI: http://dx.doi.org/10.1145/2858036.2858106

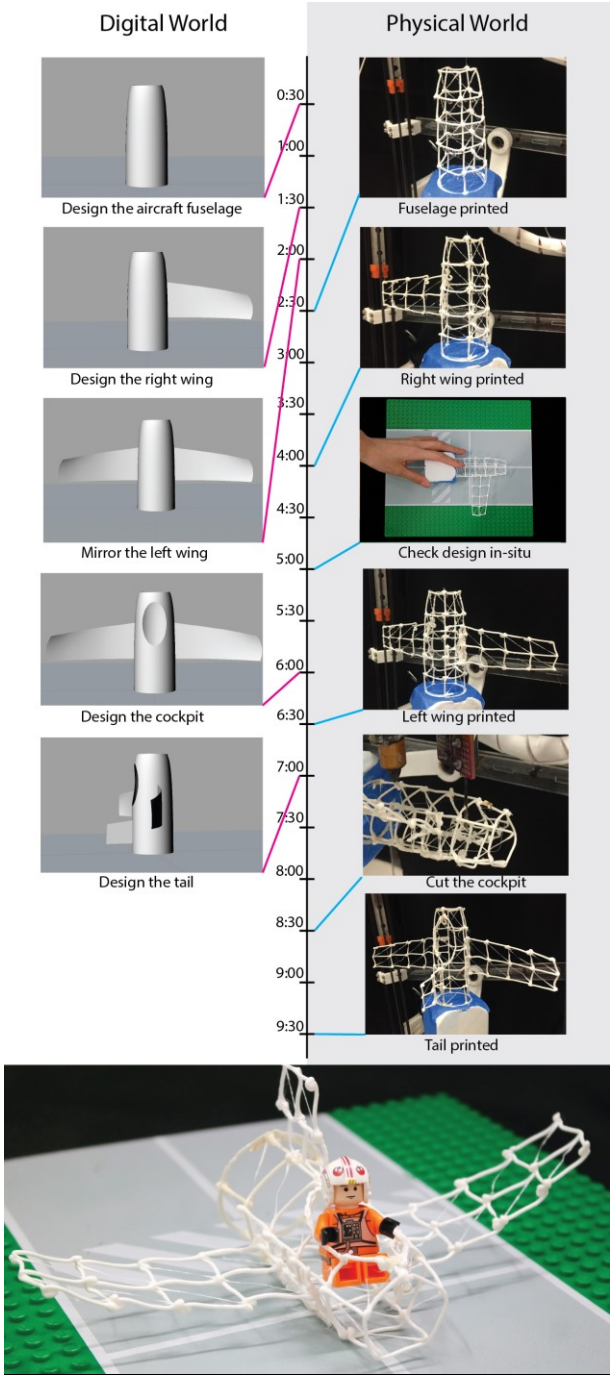

Figure 1: Designing an aircraft model with On-the-Fly Print in 10 minutes.

the design process; on the other hand, they are often slow because designers have to instantiate the physical model themselves. Further, CAD users might find them frustrating to use because they lack support for implicit 3D commands (such as construction solid geometry and complex surface creation), which are difficult to implement in interactive fabrication systems.

At the same time, there has been a push to reduce the lead-time between the creation of a 3D model and its actual instantiation. For example, faBrickation [18] and Platener [2] limit the use of a 3D printer to complex geometry, and WirePrint [16] creates a low-fidelity mesh representation of the model. All these approaches serve the traditional "design-fabricate-refine" workflow, which is common in digital fabrication, and focus on printing completed models.

In this paper, we explore how to bridge these two approaches using *On-the-Fly Print*. In our system, printing takes place incrementally, in parallel with the use of the CAD system (Figure 1). As primitives are added to the digital model, our 3D printer instantiates them using a low-fidelity wireframe representation. During the creation of the digital model, the designer can remove the building platform from the printer and observe the model in the context of its future use. She can then return the model to the printer and further modify the digital model as needed while the printer synchronizes modifications to the physical model.

On-the-Fly Print relies on a faster, incremental implementation of the WirePrint [16] system. To speed up the original system up to four times, we extended the reach of the print head and added water mist cooling to quickly solidify the extruded filament. We further modified an off-the-shelf delta 3D printer by adding a two-degree-of-freedom (2DOF) rotation platform to enable incremental prints from a wide range of angles. Finally, we added a retractable cutting blade to allow for subtractive operations and error correction. The printer is controlled by a customized Rhino plug-in [20], which observes the geometry created by the user and can optimize the printing schedule and resolve possible printing collisions to maximize printability. If unsolvable collision is detected, our system can leave parts unprinted, but it will never prevent the user from creating the desired geometry.

In the following sections, we illustrate how the *On-the-Fly Print* could provide CAD designers with tangible feedback during the creation of a digital model (Figure 1). We present a detailed description of our system. We conclude with limitations of our current system and propose possible future directions.

RELATED WORK

Our work builds upon the notions of interactive fabrication, fast fabrication, and hybrid fabrication systems.

Interactive Fabrication

ModelCraft [26] examined the use of pen annotation to create precise 3D digital models with in-situ measurements, but the modifications stayed virtual until the next 3D print cycle. CopyCAD [7] enables a user to remix designs by interacting with and using real world objects. Inspired by traditional craft activities, Willis et al. [32] introduced the concept of interactive fabrication, mixing the hands-on approach of craft with the advantages of digital construction. This concept has been instantiated by several systems. Constructable [17] supports interactive 2D laser cutting for functional objects. Rivers et al. [22] use a digitally controlled 2D router for better cutting precision. FreeD [33, 34] lets users personalize the rendering of a 3D model using a digitally controlled milling tool, and D-Coil [19] uses digitally controlled wax extrusion and cutting to blend digital and physical creation of a model. ReForm [31] combines direct hand shaping, digital extrusion and milling to provide the maximum flexibility to users. Like Constructable, D-Coil and ReForm, our goal is to offer a readily available physical instantiation of the digital model *during* the design process. However, we specifically target CAD users who might not be interested in directly participating in the physical creation their model and prefer that a fast 3D printer create a low-fidelity prototype of the current digital model for them.

Fast Fabrication

The availability of fast prototypes is another approach to a more interactive fabrication process. To this end, several solutions have been proposed by printing with a low-fidelity model. faBrickation [18] replaces most of the building volume with Lego bricks with only the detailed parts being 3D printed. Platener [2] follows a similar path with laser cut sheets being used to create the bulk of the volume, while 3D printed parts are used for complex shapes. One drawback of both approaches is that they require assembly. WirePrint [16], in contrast, saves significant time in 3D printing by printing only a mesh of the shell of the prototype. *On-the-Fly Print* relies on an optimized version of WirePrint for fast and incremental printing. Thus, it provides users with ongoing access to a physical copy of their design while it is being created in the digital realm.

Finally, Carbon3D [30] provides CLIP printing that is 25 to 100 times faster than traditional printing techniques. This hints at a not-too-distant future when there will be reduced trade-offs between speed and quality of rendering. This system is not incremental, however, which might create a large overhead in material and time.

Hybrid and Augmented Fabrication

High DOF CNC machines have been in use in industrial contexts for a long time (e.g., to create complex carbon fiber shapes or in subtractive fabrication). Mataerial [14] introduced a robot arm to create complex 3D shapes, and Song et al. [27] demonstrated a parallel kinematic machine for conformal printing. Gao et al. [8] showed how electronics can be embedded into a 3D print with a rotational cuboidal platform. MultiFab [25] demonstrated vision based approaches for printing on existing objects with multiple materials. To save 3D material and eliminate reprint, Teibrich et al. [29] illustrated a system to patch 3D printed objects using a 3+2 DOF printer. To enhance 3D printed object, Shi

et al. [24] introduced a labeling toolkit that can add audio with 3D printed features. Encore [5] allowed the user to augment physical object with printed attachment. Like these systems, *On-the-Fly Print* explores how one could extend the pattern of use of 3D printing in design practice by combining additive and subtractive methods with a 5DOF motion platform. Our system is unique in that it focuses specifically on how to print a low fidelity physical model in parallel with the creation of the digital model.

DESIGN GOAL

Our main design goal with *On-the-Fly Print* is to provide CAD users with a tangible preview of their digital model during the digital design process. To illustrate a typical interaction, we consider the case of a user designing an aircraft model compatible with a Lego airport runway set (Figure 1). After measuring a standard Lego airport size, the user starts the design of the aircraft fuselage in Rhino. The system starts printing the fuselage automatically after the geometry is finalized. As this happens, the designer moves on to specify the right wing using a curved design. She then mirrors the first wing to create the left wing. About 2 minutes later, the first wing is printed and she pauses the printer to remove the model and check if the proportion of the wing matches the Lego runway. Satisfied with the proportion of the design, she puts the model back into the printer and focuses on the design of the cockpit while the printer resumes printing. Her goal is to fit a Lego pilot inside the model. She creates cutting geometry to open a hole in the fuselage, and finally adds a tail. Within 3 minutes of her last addition, she has a finished WirePrint model in her hand (Figure 1), which has already been evaluated in-situ at key design steps. As pointed out by Buxton in the context of interface design [3], this ability to quickly check a design against its intended pattern of use significantly enhances the quality of the final design.

The workflow illustrated above is made possible by the design of a fast incremental WirePrint printer, together with a Rhino plugin that generates the proper printing instructions for each new feature. Our plugin also controls the order in which each feature is executed to limit possible conflicts during the construction process. We now present both aspects in more detail.

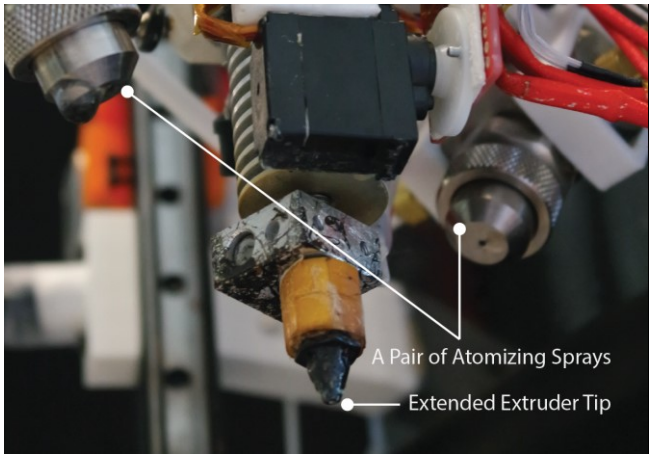

Figure 2: Print head design with extended extruder tip and mist cooling sprays.

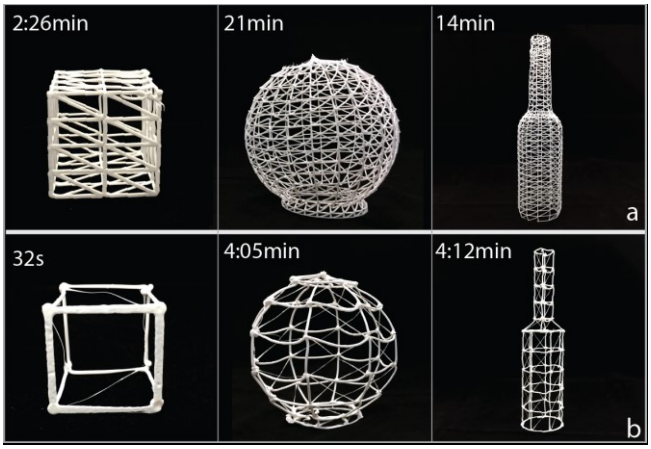

Figure 3: Speed comparison between the original WirePrint (a) and On-the-Fly Print (b).

HARDWARE: FAST AND INCREMENTAL 3D PRINTER

One key prerequisite of *On-the-Fly Print* is the ability to print basic primitives very quickly to catch up with the design of the digital model. In this regard, WirePrint [16] was a natural starting point. We noted that the speed of WirePrint could be increased significantly if one were to reduce the total number of cells to be printed by increasing their size. To do so, we modified the print head with an extended extruder tip to reach deeper into the model without collisions. To maintain the stiffness of the model we also modified the extruder tip to create a thicker (1mm) strand of ABS. One of the drawbacks of creating a thicker filament is longer cooling time. To minimize the need for cooling pauses, we added mist cooling to our system by placing two atomizing nozzles flanking the print head (Figure 2). Together, these modifications allowed us to print a 28 x 28 x 28 mm wireframe cell in 32 seconds as compared to 2:26min reported in the original WirePrint paper (Figure 3). We conducted similar comparisons for two additional models presented in the original WirePrint paper (Figure 3). Our modifications increased speed by a factor of 4.6, 5.1 and 3.3 respectively.

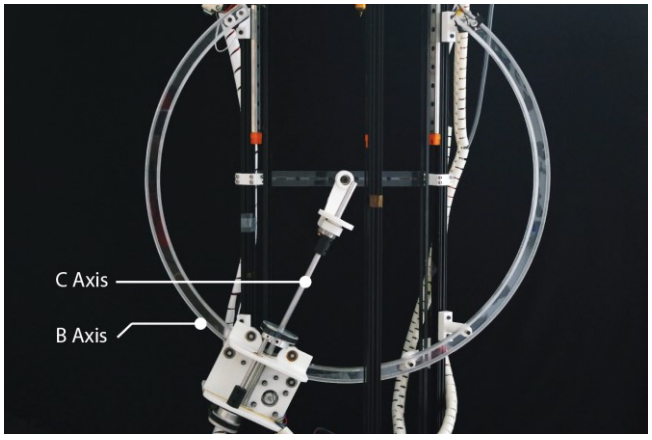

Figure 4: 5DOF add-on design to an off-the-shelf Delta 3D printer.

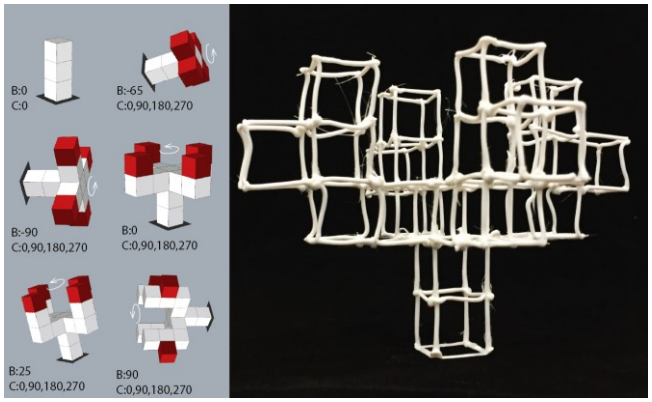

Figure 5: Calibration results. Cubes are printed at the various poses shown at left, yet are properly aligned.

Incremental Printing

Coupling 3D printing and the CAD design process requires the ability to print incrementally to avoid reprinting a model from scratch every time it is modified. Because modifications may affect all parts of the model, not just the easily accessible top, we added two rotational degrees of freedom to an off-the-shelf delta printer, with a B axis (rotating around the Y axis) and a C axis (rotating around the Z axis) (Figure 4). Compared to previous 5 DOF designs [29, 31], our design creates a large open space around the model to ensure easy access for the print head. Further, we used a fully synchronous 6DOF system (5DOF for motion and 1DOF for extrusion) so that instead of printing and cutting in a layer-based manner, we achieve conformal printing and cutting, which offers better physical rendering and quicker cut operations.

Our system is designed with the aim of affordable hardware setup and large open space. Thus, to create robust prints we pay particular attention to the calibration of the rotation platform using a computer vision calibration method [12]. This approach reaches an RMS positioning error of 0.5mm, which is sufficiently small compared to our 1mm diameter nozzle. To demonstrate our calibration results, we show a model with multiple stacked cubes printed from different angles in Figure 5. Notice that all nodes are aligned and well connected. We provide more details in the implementation section.

Cutting Operations

To let the user correct mistakes, adjust geometry and perform subtractive operations, we include a cutting tool in our system.

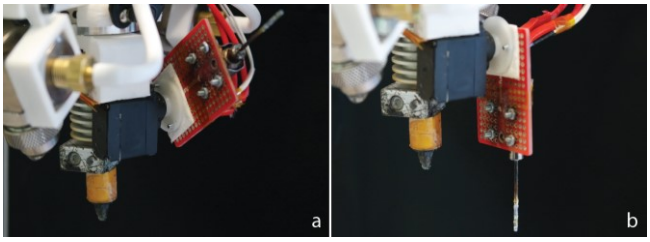

Figure 6: The cutting blade tucks away during the printing process and extends out during cutting.

We first considered using a milling head [29, 31], but this approach is problematic because milling arrangements are heavy and occupy a lot of volume, which could significantly limit the accessibility of the cutting tool. Instead we found that the wireframe shell can be cut using a simple heated blade. Our blade is actuated so it can be tucked away when not in use (Figure 6). When cutting is needed, the blade is moved to the cutting position, which is 15mm lower than the extruder tip to leave as much clearance as possible. This solution is also faster than having to load a new tool.

Moving the Printed Model In and Out of the Printer

As illustrated in the scenario above, to get full advantage of our system, the user should be able to easily remove the model from the printer to compare against real-world constraints and return it later to continue the design. To this end, our system includes a simple removable building platform aligned by a set of magnets.

SOFTWARE

The main goal of the software plugin is to convert the geometry created inside Rhino into a set of printer commands (G-code) suitable for efficient printing. For additive operations, this includes creating a mesh for the shape to be added as well as any connecting structure necessary; for subtractive operations and corrections, this includes converting the input into a cut operation and then creating any repairs necessary. Another important aspect of the plugin is to optimize the print operations to maximize printability without distracting the user. In the following, we first present the basic additive and subtractive approaches, before presenting how they can be modified to take into account print head accessibility.

Additive Geometry

In typical use, the plugin tracks the creation of geometry to decide when a new feature is ready for printing. More specifically, we consider a new feature for printing if it has not been modified for 5s and either lands on the printing platform or is connected with previously printed geometry. For example, in Figure 7, it is only when the sphere is

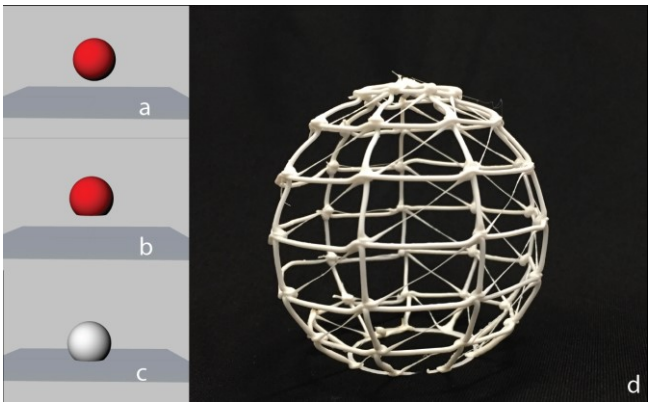

Figure 7: Criteria to commence physical printing. Printing (d) won't start until the digital model intersects with the building plane (a, b, c). Red (a, b) is color coded as unprinted part.

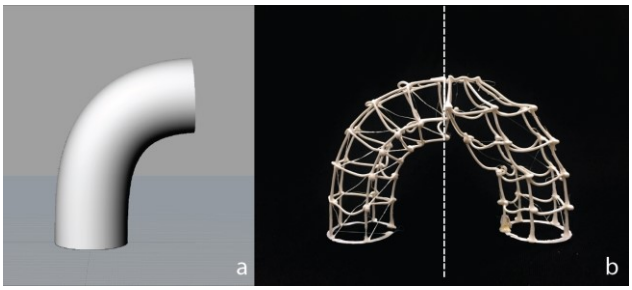

Figure 8: (a): A digital tube model, (b): the model sliced based on UV mapping (left) and traditional parallel slicing (right).

truncated and made to intersect with the building plane that printing starts. Once a new feature is ready to be printed, the plugin uses the UV mapping associated with the feature surface to generate a regular mesh. As shown in Figure 8, using UV coordinates better conveys the structure of the shape when printed in large cells. As UV mapping may lead to non-parallel slicing, our printer rotates the model to extrude upward. The resulting G-code is sent to the printer, which in turn sends an acknowledgement upon completion.

Creating connecting patches

We are using large print cells to speed up the printing process, but this leaves large gaps in the external surface of the object. If new geometry is to be created at the location of a gap, we first need to patch the surface. With speed in mind, we implement this by sub-dividing the patch area and printing the corresponding denser pattern. Because each printed segment rests on an established boundary, this printing is very fast and does not require cooling (Figure 9).

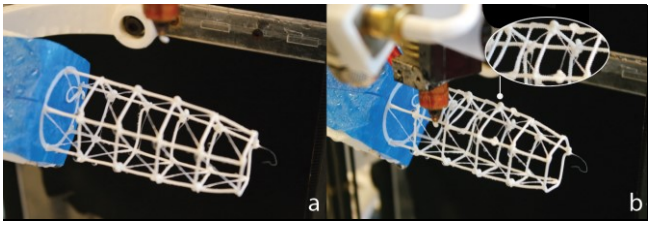

Figure 9: Creating a connecting patch for the wing.

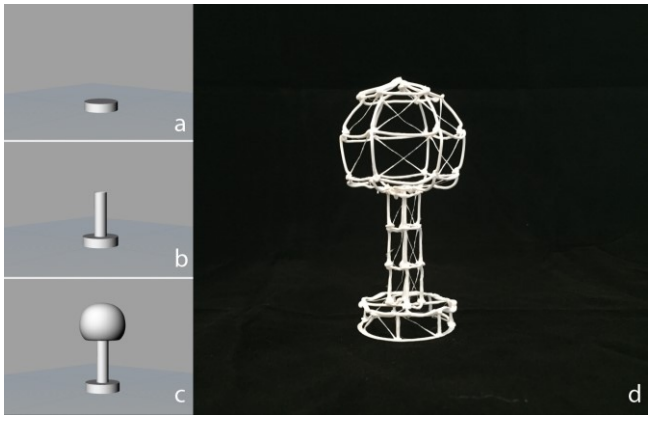

Figure 10: The system accommodates overhang by extending from the contacting curve to the new contour.

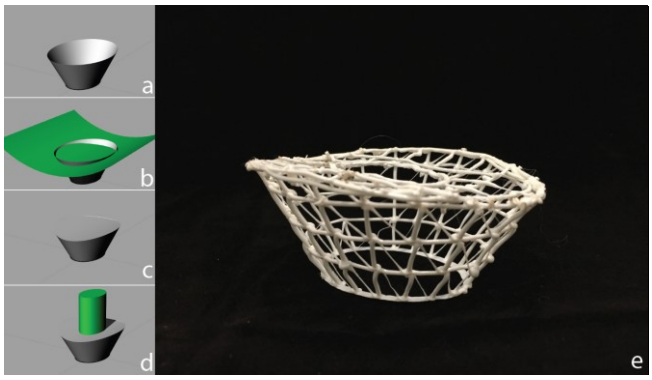

Figure 11: The Bird Nest stadium showing complex cut patching. Green (b, d) is color coded as the cutting geometry.

Another important case to consider is when incremental components create overhangs, such as in the model of the Jucker and Wagenfeld lamp shown in Figure 10. In that case we create a mesh extending from the top surface of the base to the bottom contour of the shade to provide support. A similar approach is used when the design object is bigger than our printing plate. The dinosaur shown in Figure 14a is an example of such a case.

Subtractive Geometry

Our system uses cutting surfaces to carry out cutting operations. Once the user has created the cutting geometry, we compute the intersection curve between the original object and the cutting geometry. This curve is then used to drive the cutting blade. With the help of the rotating platform, the cutting blade is maintained in an orientation normal to the object's surface and the cut is performed with all 5DOF moving synchronously. Once the cut is completed, the print head follows the same contour to "heal" the edge of the cut. This step is necessary because we are using a mesh representation of the model. We then generate the mesh necessary to close any holes as required by the user. In the Bird's Nest stadium example shown in Figure 11, the printer first cuts the tapered geometry (a) and creates a curved patch (b) before creating a new cut in the center (c).

As in the case of additive geometry, when a new cut feature is smaller than the size of the mesh cells, our system can create a supporting patch before performing the cut. We show an example of this feature in Figure 12.

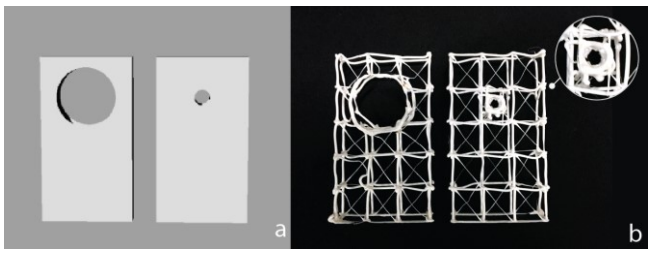

Figure 12: Adding holes to a model. When the hole is much smaller than the cell, we first create a supporting patch.

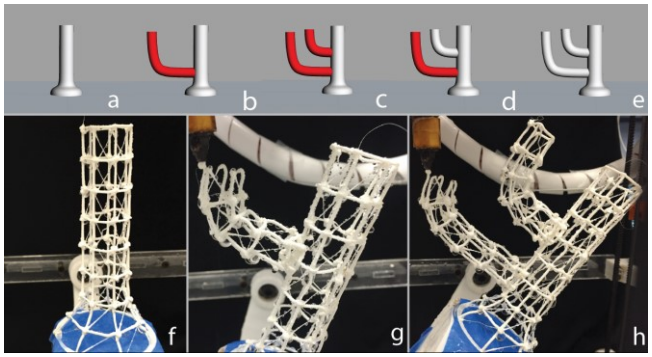

Figure 13: Reordering. The user designs the bottom branch first (b) and the top branch later (c). Noticing that the print of the bottom branch will block the way for printing the top branch, the system reorders the print sequence (g, h).

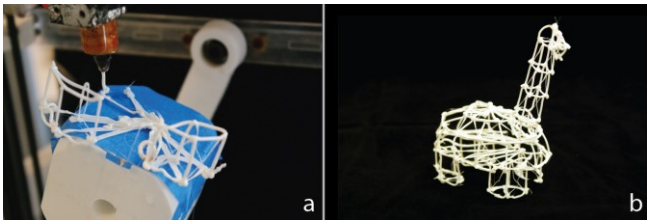

Figure 14: Dinosaur legs are printed at an angle to avoid collision.

Dealing with Collisions

So far, our presentation has ignored the problems caused by collisions. As explained above, our printer was designed from the start to provide maximum tool accessibility, yet it is to be expected that some design sequences will create conflicts between the existing model and the added geometry. We now present how our system uses a staged approach, focusing on early prevention of collisions, to deal with these conflicts.

Out of order printing

Our first step is to explore whether re-ordering the pending primitives will improve printability. This is possible because the printer often lags behind the user's actions. To do so, before printing any new primitive we first observe the queue of pending operations. We compute the possible printing orders and chose the one with the least printing conflicts. For example, in the candelabra shown in Figure 13, the user creates the branches from bottom to top, but we detect that printing the bottom part first will interfere with printing the top branch, so our system prints them from top to bottom instead to limit conflicts. To allow users to easily monitor the progress of the system, features that are delayed in this way can be color coded in red on the screen.

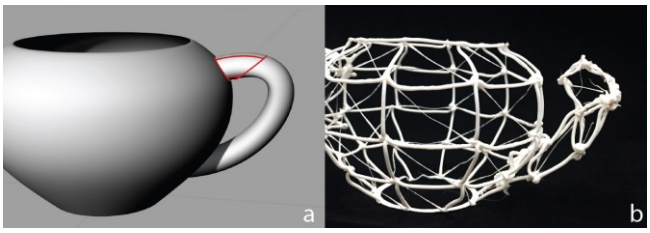

Figure 15: Example of omitted details. Printing the teapot handle would collide with body, so our system omits the part that would cause collisions and prints the rest.

Relaxing printing orientation

Our second approach is to relax the printing orientation using collision detection, similar to [29]. As a default, our system will orient the model so that printing proceeds upward. If this causes a conflict, we will try to repose the object to avoid collision. One example of this approach can be seen in Figure 14, where the system prints the body of the dinosaur sideways to prevent collision between the print head and the previously printed legs. More details about the algorithm are provided in the implementation section.

Omitting geometry

If it is still not possible to print a full feature after sampling different orientations, the system will attempt to print as much of the new feature as possible, leaving the parts that would cause collisions unprinted. This reflects our aim to provide a low-resolution overview of the model as it is designed with minimum distraction to the user's workflow. For example, when creating the handle of the teapot shown in Figure 15, part of the handle is not printed because it is not reachable. Similarly, the system will not attempt to print the walls of the holes shown in Figure 12.

Pause and manual mode

So far, we have described the system working in fully automatic mode. This mode generates a corresponding physical model in parallel to the digital design process. During the design process, the user can also postpone the physical instantiation using a *Pause* command. This allows

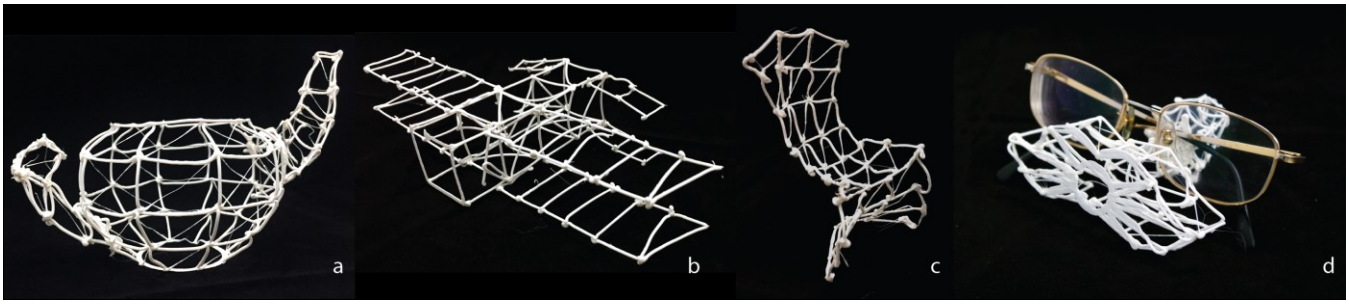

Figure 16: Printing samples. (a): tea pot created printed in 3 steps; (b): Wright brothers' airplane printed in 9 steps; (c): Panton's stacking side chair designed using a loft operation; (d): glasses holder printed in 3 steps.

the user to quickly try several designs in the digital realm without worrying about printability or the physical costs of updates. When printing is resumed, queued primitives may also provide opportunities to our scheduling algorithm to reorder parts for better printability. Finally, if the user would like to have full control of the printing process, our software offers a *Manual* mode. In this mode, the software only prints the primitives that are manually selected by the user.

Further Examples

In Figure 16 we show several examples of models created with our system, including a teapot, a model of the Wright brother's airplane, a chair inspired by Panton's stacking side chair created using a loft operation, and a glasses holder larger than the build platform.

IMPLEMENTATION

Our system has two main components: a custom-built 5DOF printer and a Rhino plugin that processes the geometry and generates the G-code to control the printer. The systems communicate through a job queue, with a notification from the printer to the Rhino plugin to indicate when a job has been completed. Our plug-in was designed to limit user distraction to a minimum. It adds a simple set of controls to the Rhino screen to let the user pause the system, visualize the actual meshing and control the mesh cell size, among other functions. It uses color coding to provide feedback about the state of the print.

Printer Modification

Our Mini Kossel [15] delta printer is driven by a Beagle Bone Black [1] connected to a CRAMPS 2.0 module [6]. It is running LinuxCNC [11] modified to support a 6DOF (5DOF motion and 1DOF extrusion) system.

Extruder

As shown in Figure 2, we modified the standard hot end assembly by adding a 15mm pipe adaptor to extend its reach. The chamber of the extrusion head is kept at 270C to speed up the extrusion of our 1mm filament. We also added two SU11 atomizer nozzles [28], one on each side, to create a mist that rapidly cools the extruded material. In our current prototype, curves in the U direction are printed without cooling to guarantee a firm connection between layers. As a result, the curvature of U might not be preserved as accurately.

Cutter

The cutter (Figure 6) was created by adapting a heating cartridge [9] and a thermistor to fit a cutting blade [13]. The assembly is attached to a small servomotor so that it can be raised when not in use. The cutter's heating cartridge is maintained at 350C so that the cutting tip can easily cut 1mm thick ABS wires.

5 DOF extension

Our prototype uses a circular rail of 260mm radius for the B axis, so as to maximize the accessible print volume. To lower cost, this rail is made of two 6mm-thick acrylic layers glued together to create a butterfly profile. The range of motion is +/– 120°. The C axis can rotate indefinitely. To accommodate the circular rail, we extended the height of the printer by changing the support rails. Our setup is inexpensive and relies only on easily fabricated parts in the spirit of the Mini Kossel design, yet, with the proper calibration, it has enough precision for fast and incremental WirePrint.

Calibration

Our printer was designed for maximum printhead accessibility with affordable hardware setup. As a result, small construction errors adds up during the printing. Our computer vision based calibration system guarantees good accuracy with simple "maker" grade assembly. Two stationary cameras on the side of the machine capture images of a cube placed on the platform for 114 combinations of B and C angles. The OpenCV calibration program [4] recognizes checkerboards attached to five sides of the cube and computes an initial estimate of the intrinsics and extrinsics of the camera-checkerboard system. Given those estimates as input, we used Caliber [12] to recover the exact poses of the platform at the corresponding angles.

With these samples, we can then interpolate the data to predict the pose of the platform given an arbitrary combination of B and C angles. We model the B axis rotation as rigid body motion and fit a circle to the translational part of the pose samples. Then we interpolate the deviation from the best-fit circle to obtain the translation given an arbitrary B angle. We use spherical linear interpolation to obtain the rotational part. The motion of the C axis is well modeled as a rotation, so we only need to solve for the unknown rotation axis using the calibration data.

Key Aspects of Software

Here we present the key features of our Rhino plugin.

UV Mesh and Patch Generation

As a spline-based modelling tool, Rhino offers built-in APIs [21] to generate UV maps. For a given primitive, our plugin first calls the Surface.IsoCurve() function to create a reference V contour. It then samples equally spaced points along this V contour to generate all U contours for printing. The plugin then picks one U contour as the reference to create all V contours for printing.

The connection patches are generated by a similar method. To patch an area, the plugin finds the boundary of the area, sub-samples it to create new V and U lines, and prints accordingly.

Collision Detection and Out of Order Printing

To detect collisions and relax printing orientation, the system positions a built-in extruder model at each of the (U, V) nodes along the primitive mesh and uses the Brep.IsPointInside() function to check for a collision. If a collision is detected at a given node, the system will move the extruder to different orientations until a collision free direction is found (or it decides not to print the edge). Orientations are sampled on a fixed grid over the hemisphere.

A similar method is used to implement out-of-order printing. Given the list of pending primitives, the plugin first

Figure 17: The same vase created in CAD in a multi-step approach (left) versus a single-step approach (right) results different physical appearances. On the left, the top section of the sphere felt into the model during the cutting process.

enumerates all possible printing orders. For each order, it uses collision detection to evaluate the printability (with high printability assigned to the case when the extruder tip doesn't need to be repositioned). The system then picks the highest printability order to print. This computation might take up to a couple of seconds in our non-optimized implementation, but it happens in the background so it has little impact for the user.

DISCUSSION AND FUTURE WORK

The prototype described above illustrates the potential of *On-the-Fly Print* for providing CAD users with a quick tangible feedback on their design. We now consider some aspects of the current prototype that warrant further research efforts.

Printability, Limitation and Future Improvement

A main objective of our system was to limit user involvement during printing. To do so, *On-the-Fly Print* implements a "*best-effort printing*" policy based on limited information about the final shape of the model. We never prevent the user from creating geometry, but we might leave some parts unprinted. Printability or even final results will therefore depend on the steps by which a model is created. As an example, Figure 17 shows the same vase created either in a stepwise modification of a sphere or in a single step via revolution of a profile; each ends up with a different physical rendering.

Similarly, some operations like *Scale* or *Move*, might be costlier than others because they will require the selected features to be reprinted physically. The worst case is a global rescale, which will require restarting from scratch. This costs more time than other operations, but might not be a big issue given the fast printing speed. However, field deployment will be needed to better understand the impact of this and how people approach the design of complex objects with our system.

We should also note that our current printing strategy, which proceeds in a fixed sequence dictated by the UV mapping, makes it difficult to print certain topologies such as a torus (Figure 15). Inner details created after the bulk of the object has been created will also be ignored (or will require a full reprint).

Future Improvement

Currently, our approach stops short of using the cutting tool to selectively remove the part of the model that is in the way of the print head. Instead, new features conflicting with the current model are marked as pending and might never be printed as part of the preview. We envision that making use of the cutting capability would further improve printability. We foresee two general approaches: First, the system could identify which features need to be cut away and returned to the print queue to be reprinted in a different order. Second, the system could identify the minimal flat cut across the model making printing possible and proceed using a standard WirePrint slicing.

Our current implementation does not check for cut-off parts falling into the geometry. For example, in Figure 17 left, the top hemisphere falls into the printed body after the cut operation (but doesn't cause a printing problem). However, several options could solve this as a future improvement. 1) The system could pose the print so that the cut part does not fall inside; 2) The system could pour and shake the cut part out using the rotation platform after cutting (but it might get stuck); 3) The system could pause the print to let the user remove the part if needed.

Finally, our current implementation ignores collisions during cut operations. This has not been a problem in our sample cases, but a complete system would implement this feature using similar approaches as for adding geometry.

Low Resolution Printing

To enable fast preview, we traded speed for accuracy using an extension of the WirePrint method. In the early stages of the design process, the low resolution of the resulting models is not a concern. In fact, low-fidelity models could be a benefit because they clearly communicate to viewers that the design is not finished, but is a sketch to be critiqued [23]. However, low print resolution can become a problem later on when users want to add more details to a design. Like in WirePrint, one solution could be to let users balance the speed/resolution trade-off by giving them the option to incrementally render accurate shells on specific areas of interest. Because our printer is conformal this will not always require a full reprint, as in the case of the original WirePrint implementation. However, it may require a redesign of the cutting mechanism. New printing approaches such as Carbon3D [30] might soon reduce the need for such trade-offs.

High Degree of Freedom Printers

This project and others like the Patching paper [29] are accumulating evidence that adding a rotating platform to the more standard 3D printer might significantly improve the flexibility of these systems when it comes to extending the

way they could be used in practice. Both systems demonstrate that the benefits might be worth the extra cost.

Interactive Versus On-the-Fly Print

One of the most interesting aspects of this work is that it makes it possible to empirically study several important questions including: 1) When will users prefer interactive printing over *On-the-Fly Print*? 2) How might both approaches change the way people create digital models? 3) Will either approach bring a more reflective practice to the design of digital models? Traditionally, reflective practice has focused on lower fidelity techniques such as sketching [3], but we believe that the progress in CAD tools and better tangible feedback might change this. We are planning to conduct empirical studies exploring these questions in more detail.

CONCLUSION

In this paper we present *On-the-Fly Print*, a system bridging the gap between hands-on interactive fabrication and purely digital modeling practice. To do so, *On-the-Fly Print* creates low-fidelity physical sketches in parallel with the creation of the corresponding digital model. Throughout the 3D modeling process, the user can hold the physical print in hand to gather tangible feedback and evaluate its fit with external constraints.

As the user creates new features in our CAD software, our system automatically creates a WirePrint representation and sends it to our fast 5DOF printer. Our system can handle both additive and subtractive operations and is designed to minimize the impact of possible interference between the print head and the existing model. We demonstrated how the system can handle a wide variety of shapes and described the limitation of the current implementation as well as ways to limit their impact.

On-the-Fly Print lets CAD users have continuous access to a low-fidelity representation of their design. We believe that this approach has the potential to improve the overall quality of the design process.

ACKNOWLEDGEMENTS

This work was supported in part by NSF Awards IIS-1422106 and IIS-1011919 and by Autodesk Corporation. We thank Corinna Loeckenhoff and Ge Gao for their assistance and the anonymous reviewers for their valuable comments.

REFERENCES

1. Beagle Bone Black. Retrieved Dec 24, 2015 from http://beagleboard.org/BLACK
2. Dustin Beyer, Serafima Gurevich, Stefanie Mueller, Hsiang-Ting Chen, and Patrick Baudisch. 2015. Platener: Low-Fidelity Fabrication of 3D Objects by Substituting 3D Print with Laser-Cut Plates. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (CHI '15), 1799-1806. http://dx.doi.org/10.1145/2702123.2702225
3. Bill Buxton. 2010. *Sketching user experiences: getting the design right and the right design: getting the design right and the right design*. Morgan Kaufmann.
4. Camera Calibration with OpenCV. Retrieved Dec 24, 2015 from http://docs.opencv.org/doc/tutorials/calib3d/camera_calibration/camera_calibration.html
5. Xiang 'Anthony' Chen, Stelian Coros, Jennifer Mankoff, and Scott E. Hudson. 2015. Encore: 3D Printed Augmentation of Everyday Objects with Printed-Over, Affixed and Interlocked Attachments. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (UIST '15), 73-82. http://dx.doi.org/10.1145/2807442.2807498
6. CRAMPS 2.0. Retrieved Dec 24, 2015 from http://reprap.org/wiki/CRAMPS
7. Sean Follmer, David Carr, Emily Lovell, and Hiroshi Ishii. 2010. CopyCAD: remixing physical objects with copy and paste from the real world. In *Adjunct proceedings of the 23nd annual ACM symposium on User interface software and technology* (UIST '10), 381-382. http://dx.doi.org/10.1145/1866218.1866230
8. Wei Gao, Yunbo Zhang, Diogo C. Nazzetta, Karthik Ramani, and Raymond J. Cipra. 2015. RevoMaker: Enabling Multi-directional and Functionally-embedded 3D printing using a Rotational Cuboidal Platform. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (UIST '15), 437-446. http://dx.doi.org/10.1145/2807442.2807476
9. Heating Cartridge. Retrieved Dec 24, 2015 from www.makergeeks.com/cecahefor3dp.html
10. Joong Han Lee. Haptic Intelligentsia. Retrieved Dec 24, 2015 from http://studio-homunculus.com
11. LinuxCNC. Retrieved Dec 24, 2015 from http://www.linuxcnc.org
12. Albert Liu, Steve Marschner, Noah Snavely. 2015. Caliber: Camera Localization and Calibration Using Rigidity Constraints. *International Journal of Computer Vision*, pp.1-21. http://dx.doi.org/10.1007/s11263-015-0866-1
13. Master Appliance. Retrieved Dec 24, 2015 from https://www.masterappliance.com/content/35407-tip-assortment-kit-5-tips-1-adaptor
14. Mataerial. Retrieved Dec 24, 2015 from http://www.mataerial.com
15. Mini Kossel. Retrieved Dec 24, 2015 from http://reprap.org/wiki/Kossel
16. Stefanie Mueller, Sangha Im, Serafima Gurevich, Alexander Teibrich, Lisa Pfisterer, François Guimbretière, and Patrick Baudisch. 2014. WirePrint: 3D printed previews for fast prototyping. In *Proceedings of the 27th annual ACM symposium on*

User interface software and technology (UIST '14), 273-280. http://dx.doi.org/10.1145/2642918.2647359

17. Stefanie Mueller, Pedro Lopes, and Patrick Baudisch. 2012. Interactive construction: interactive fabrication of functional mechanical devices. In *Proceedings of the 25th annual ACM symposium on User interface software and technology* (UIST '12), NY, USA, 599-606. http://dx.doi.org/10.1145/2380116.2380191
18. Stefanie Mueller, Tobias Mohr, Kerstin Guenther, Johannes Frohnhofen, and Patrick Baudisch. 2014. faBrickation: fast 3D printing of functional objects by integrating construction kit building blocks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '14), 3827-3834. http://dx.doi.org/10.1145/2556288.2557005
19. Huaishu Peng, Amit Zoran, and François V. Guimbretière. 2015. D-Coil: A Hands-on Approach to Digital 3D Models Design. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (CHI '15), 1807-1815. http://dx.doi.org/10.1145/2702123.2702381
20. RhinoCAD. Retrieved Dec 24, 2015 from http://www.rhino3d.com
21. Rhinocommon SDK. Retrived Dec 24, 2015 from http://4.rhino3d.com/5/rhinocommon/
22. Alec Rivers, Ilan E. Moyer, and Frédo Durand. 2012. Position-correcting tools for 2D digital fabrication. *ACM Trans. Graph*. 31, 4, Article 88. http://dx.doi.org/10.1145/2185520.2185584
23. Jutta Schumann, Thomas Strothotte, Stefan Laser, and Andreas Raab. 1996. Assessing the effect of non-photorealistic rendered images in CAD. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '96), 35-41. http://dx.doi.org/10.1145/238386.238398
24. Lei Shi, Idan Zelzer, Catherine Feng, and Shiri Azenkot. 2016. Tickers and Talker: An Accessible Labeling Toolkit for 3D Printed Models. In *Proceedings of the 34rd Annual ACM Conference on Human Factors in Computing Systems* (CHI '16). http://dx.doi.org/10.1145/2858036.2858507
25. Pitchaya Sitthi-Amorn, Javier E. Ramos, Yuwang Wangy, Joyce Kwan, Justin Lan, Wenshou Wang, and Wojciech Matusik. 2015. MultiFab: a machine vision assisted platform for multi-material 3D printing. *ACM Trans. Graph*. 34, 4, Article 129. http://dx.doi.org/10.1145/2766962
26. Hyunyoung Song, François Guimbretière, Chang Hu, and Hod Lipson. 2006. ModelCraft: capturing freehand annotations and edits on physical 3D models. *In Proceedings of the 19th annual ACM symposium on User interface software and technology* (UIST '06), 13-22. http://dx.doi.org/10.1145/1166253.1166258
27. Xuan Song, Yayue Pan, Yong Chen. 2015. Development of a Low-Cost Parallel Kinematic Machine for Multidirectional Additive Manufacturing. *Journal of Manufacturing Science and Engineering*, 137(2), 021005. http://dx.doi.org/10.1115/1.4028897
28. Spraying System. Retrieved Dec 24, 2015 from http://www.spray.com/cat70/cat70pdf/ssco_cat70_f.pdf
29. Alexander Teibrich, Stefanie Mueller, François Guimbretière, Robert Kovacs, Stefan Neubert, and Patrick Baudisch. 2015. Patching Physical Objects. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (UIST '15), 83-91. http://dx.doi.org/10.1145/2807442.2807467
30. John R. Tumbleston, David Shirvanyants, Nikita Ermoshkin, Rima Januszewicz, Ashley R. Johnson, David Kelly, Kai Chen, Robert Pinschmidt, Jason P. Rolland, Alexander Ermoshkin, Edward T. Samulski, Joseph M. DeSimone. 2015. Continuous liquid interface production of 3D objects. *Science*, 347(6228), 1349-1352. http://dx.doi.org/10.1126/science.aaa2397
31. Christian Weichel, John Hardy, Jason Alexander, and Hans Gellersen. 2015. ReForm: Integrating Physical and Digital Design through Bidirectional Fabrication. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (UIST '15), 93-102. http://dx.doi.org/10.1145/2807442.2807451
32. Karl D.D. Willis, Cheng Xu, Kuan-Ju Wu, Golan Levin, and Mark D. Gross. 2010. Interactive fabrication: new interfaces for digital fabrication. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction* (TEI '11), 69-72. http://dx.doi.org/10.1145/1935701.1935716
33. Amit Zoran and Joseph A. Paradiso. 2013. FreeD: a freehand digital sculpting tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '13), 2613-2616. http://dx.doi.org/10.1145/2470654.2481361
34. Amit Zoran, Roy Shilkrot, and Joseph Paradiso. 2013. Human-computer interaction for hybrid carving. In *Proceedings of the 26th annual ACM symposium on User interface software and technology* (UIST '13), 433-440. http://dx.doi.org/10.1145/2501988.2502023