

Software Pipeline for End-to-End Fabrication of Functional Devices

By

Christina C. Liao

S.B., EECS, Massachusetts Institute of Technology (2019),
Computer Science and Engineering

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology

May 2020

© Massachusetts Institute of Technology 2020. All rights reserved.

Author: _____
Department of Electrical Engineering and Computer Science
May 19, 2020

Certified by: _____
Stefanie Mueller, Professor of Computer Science and Engineering
Thesis Supervisor
May 19, 2020

Accepted by: _____
Katrina LaCurts, Chair, Master of Engineering Thesis Committee

Software Pipeline for End-to-End Fabrication of Functional Devices

by

Christina C. Liao

Submitted to the Department of Electrical Engineering and Computer Science
on May 19, 2020 in Partial Fulfillment of the
Requirements for the Degree of Master of Engineering in
Electrical Engineering and Computer Science

ABSTRACT

Fabricating custom electronic devices has gained popularity in recent years thanks to advances in digital fabrication technologies. However, the software tools to aid users with the fabrication process have lagged behind, either being too complex for the novice user or constricting the design space too much for the sake of learnability.

This thesis describes the software pipeline for a novel fabrication method that uses a laser cutter to automatically create fully-functional devices that work immediately after fabrication is complete. The software pipeline consists of a design interface, in which users can define the circuitry and geometry of the device to fabricate, and a post-processing system that transforms the design into machine instructions that the laser cutter uses to fabricate the device. With this pipeline, users can design and fabricate high-fidelity devices without needing to know the specifics behind the fabrication process.

Thesis Supervisor: Stefanie Mueller

Title: Professor of Computer Science and Engineering

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Professor Stefanie Mueller. Her passion for research is so inspiring, and she taught me about more exciting fields of research than I could have imagined. Taking her class in Fall 2018 was what inspired me to continue on from undergrad and write this thesis; this project would not have been possible without her guidance and support.

I would also like to thank my collaborators, Martin Nisser, Aradhana Adhikari, and Yuchen Chai. You have all been amazing to work with, and I could not have asked for a better team. Martin was the driving force behind the project and was an excellent mentor and role model - I am eternally grateful for his help. Ara helped a lot with the folding aspect of the project and has been a great companion for both this project and our TA-ships. I regret that we were unable to work together as team in-person for longer, but things really came together in the end and I am proud of what we were able to accomplish remotely.

Next I want to thank the other members of the Human Computer Interaction Engineering Group for creating a fun, welcoming, and productive environment. Everyone is friendly yet has such a strong dedication to their work, a quality that has also inspired me to reach higher.

Lastly, I want to thank my parents and my sister. Their endless support is what has enabled me to come this far.

Table of Contents

I. Introduction	9
II. Related Work	11
2.1 Automated Circuit Fabrication	11
2.2 Design Tools for Fabrication	13
2.3 Augmenting Fabrication Machines	14
III. Our System: Automated Fabrication Pipeline	16
3.1 Overview	16
3.2 Fabrication Process	17
3.3 System Walkthrough	18
3.3.1 Designing the Device	19
3.3.2 Extras	20
3.3.3 Export & Post-Processing	21
IV. Implementation	23
4.1 User Interface	23
4.1.1 Toolbar	24
4.1.2 Components	24
4.1.3 Geometry & Traces	26
4.1.4 Time Estimate	27
4.1.5 Encoding Information	28
4.2 Post-Processing	28
4.2.1 Generating Additional Trace Paths	29
4.2.2 Pick-and-Place Tray	31
4.2.3 Folding	34
4.2.4 Vias	36
V. Applications	38
5.1 Quadrotor	38
5.2 Reading Light	39
VI. Discussion	41
6.1 Limitations	41
6.1.1 Well-Formed Drawings	41

6.1.2	Fabrication Constraints	42
6.2	Future Work	43
6.2.1	Circuit Validation.....	43
6.2.2	Multi-Layer Structures.....	43
VII.	Conclusion.....	44
	References.....	45

List of Figures

Figure 1: LED circuit turns on after fabrication	10
Figure 2: Hardware add-on.....	16
Figure 3: System Pipeline	17
Figure 4: UI Toolbar	18
Figure 5: Stages of Device Design.....	19
Figure 6: Extra Fabrication Features.....	20
Figure 7: Settings Panel	21
Figure 8: Full View of UI	23
Figure 9: Component Files.....	25
Figure 10: Special Components	26
Figure 11: Pipeline Steps and Color Settings	27
Figure 12: Post-Processed File.....	28
Figure 13: Silver and Solder Paths.....	30
Figure 14: Tray and Pick-and-Place Paths.....	32
Figure 15: Folding Example	35
Figure 16: Via Pipeline	37
Figure 17: Quadrotor Design Progression	38
Figure 18: Quadrotor Flying	39
Figure 19: Reading Light Design Progression.....	39
Figure 20: Reading Light in use.....	40

I. Introduction

Advances in technology, such as 3D printing and laser cutting, have made fabricating custom items such as electronics and small parts much easier, gaining popularity both for research and for casual DIY projects. However, fabricating high-fidelity electronics has not followed the same trend and is still a very time-consuming process [1]. Current options include either hand-assembling a board with soldering or sending a printed circuit board (PCB) design to a company to be printed. Neither option is very appealing: hand assembly is error-prone and still takes significant time even for expert users, and it can take several weeks to get a print back from a PCB manufacturing company. As a result, manufacture of these electronics is still mostly confined to university laboratories or factories rather than becoming as widely available as 3D printers.

In tandem with this trend, the software tools to aid users with the fabrication process have also lagged behind. Many programs commonly used with 3D printing and laser cutting still require calibration of various settings that quickly overwhelm the novice user [5]. Previous attempts at making electronics fabrication more accessible have revealed a tradeoff between ease of understanding by novices versus the complexity of circuits that can be fabricated [4]. Research that has aimed to simplify the process to make it easier to understand by novice users have as a byproduct also restricted the design space of what can reasonably be fabricated using their method. Meanwhile, other methods that support automated fabrication of complex circuits in turn require expert knowledge of the fabrication process far beyond the reach of novice users.

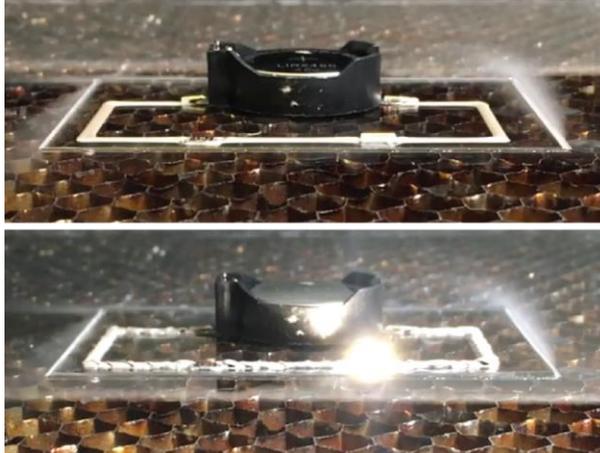


Figure 1. Our system can fabricate fully-functional devices that work immediately after fabrication is complete. Here we see that the LED in this circuit lights up once the last electrical connection is soldered.

To address some of these issues, we introduce a complete, novel fabrication pipeline where users can, after designing their circuit, press a button to have the laser cutter assemble a complete device that works as soon as it is finished as shown in Figure 1. We make high-fidelity prototyping easier for users who may not have as much technical knowledge and eliminate the need for self-assembly. Our method works without the need for any additional hardware outside of our specially-designed add-on, keeping it accessible to anyone who has access to a laser cutter.

The work described in this thesis is centered around creating the software pipeline through which users will be able to design and instantly fabricate devices using our novel method. This pipeline consists of a user interface, with which users freely design the circuitry and geometry of their devices, and a post-processing workflow that automatically transforms the user's design into machine instructions that the laser cutter uses to fabricate the device. Together, these tools enable users to fabricate fully-functioning devices without extensive knowledge about the inner workings of the fabrication pipeline.

II. Related Work

This work is related to previous literature in HCI, namely regarding circuit fabrication, design tools, and augmenting existing fabrication machines.

2.1 Automated Circuit Fabrication

A wide assortment of methods have been formulated in HCI literature to enable customized sensor and electronic circuit fabrication. Early research in this area focused on creating novel ways of fabricating circuit traces on a variety of different substrates to support circuits, starting with manually drawing in circuit traces using silver ink [12]. Instant Inkjet Circuits [7] demonstrated that inkjet printers can be used to print circuits by modifying the printer to deposit conductive silver ink, a technique that has since been applied to other mediums such as for fabricating touch displays as in PrintScreen [15]. Midas [21] used a vinyl cutter to cut out customized electric tape circuits and attached them to existing devices to augment them with various touch capabilities. Printem [18] used a laser printer to print a circuit negative onto a special film, from which a layer can then be peeled off to reveal a completed circuit board.

Latter research focused on developing automated circuit fabrication techniques, with the end goal of speeding up and streamlining the fabrication process. CircuitStack [28] combined printed circuits with PCBs to enable faster prototyping of circuits via swapping out the printed circuit layers but does not consider the geometry of the finished part, being closer to the breadboarding stage than the finished product. Valentine et al. [26] combined direct silver ink writing with automated pick-and-place to create circuits on elastic substrates to produce soft electronic boards. LASEC produces single-layer circuits

by using a laser cutter to selectively remove sections of a conductive layer from a 2-layer material [4]. Lambrichts et. al [9] used a laser cutter to cure solder paste as part of a fabrication process for producing flexible PCBs, a technique that is similar to our trace-making process but again lacks pick-and-place functionality. All of the aforementioned methods share the problem of being not fully automated; some form of manual assembly is still required at some point in the fabrication process in order to make the circuit fully functional [10]. In contrast, our method is capable of making fully functional electronic circuits without human intervention, making it among the first of its kind.

Fabricating 3D electronic devices also poses its own set of challenges, warranting a different set of approaches but still sharing the same issues with requiring manual assembly. Foldio [16] uses a combination of inkjet printers and screen printing to print circuit traces on a 2D sheet using conductive ink that can then be cut and folded to create 3D interactive objects. Peng et. al [17] wound coils around a 3D object during the printing process to create motors and solenoids. FiberWire [23] uses a combination of 3D printed carbon fiber, laser-etching, and silver deposition to create conductive traces that are embedded directly within a 3D printed object. Voxel8 [27] embeds traces directly into 3D-printed objects during the printing process using silver ink. SurfCuit [25] routes traces onto a 3D model's surface to better enable surface-mounted circuitry, but comes with the drawback of still requiring manually attaching copper tape and soldering. Foldtronics [30] uses foldable honeycombs to embed a wide range of electronics into low-fidelity 3D prototypes but required equally many manual fabrication processes. Our system supports fabrication of 3D electronic devices by enabling users to include folds in

their designs, leveraging the same technique to fold acrylic using a defocused laser as first described in LaserOrigami [13].

2.2 Design Tools for Fabrication

Customized fabrication techniques often have their own specialized design tools in order to guide users through the additional steps and processing required to fabricate a circuit with the method. The LASEC [4] interface enables users to place components, define the stretch of varying regions, then uses a routing algorithm to automatically determine the optimal path for circuit traces to take through cut patterns. SurfCuit [25] includes an interface that generates the channels for copper tape directly on the 3D print model based on user-defined component placements and connections. ./trilaterate [22] features a full pipeline for manufacturing custom 3D-printed objects that are sensitive to hover, touch, and force inputs; electrodes and connections are generated automatically by the program within a user-supplied 3D model and exports a fabrication-ready file. Common to all of these interfaces is how the user is given the freedom to custom design their circuit while the program handles satisfying the constraints of the particular fabrication method. We designed the interface for our system with a similar approach of enabling interactive component placement and connections for the circuit design while abstracting the fabrication implementation details to the post-processing step.

A secondary goal for these design tools is often learnability. Previous work in designing interfaces that are more readily adoptable by novice users tend to focus exclusively on circuits. Fritzing [8] is a widely-used open source tool for laying out circuit schematics on a simulated breadboard. Lo et. al [11] created a circuit design tool geared towards new learners of electronics, complete with circuit validation checking. These interfaces are

generally used only for design purposes and do not contribute to the fabrication of the actual circuit. The few usability-focused interfaces that are also part of larger fabrication pipelines mostly focus on making paper circuits. PEP [14] featured an integrated pipeline that guided users through the process of designing 3D folded paper-based electronics. PaperPulse [19] presented a simplified circuit fabrication process that enabled designers without a technical background to use inkjet printers to produce multi-layered electronic circuit designs on paper. These fabrication techniques are readily adoptable by novice users, but paper circuits are inherently less robust compared to the acrylic-backed devices our method can produce. Our interface is among the first that supports fabricating fully-functional devices with both circuitry and geometry. By being at a later stage of the prototyping process, our system sacrifices some usability for increased functionality, though keeping the design process simple and understandable by users with only a basic knowledge of the fabrication process was still a top priority in the interface design.

2.3 Augmenting Fabrication Machines

Augmenting commercial fabrication machines has become an increasingly popular approach for making advanced fabrication techniques more accessible, especially with 3D printers. Katakura et al. [6] fit a fused deposition modeling (FDM) printer with an attachment that uses its own CNC to assemble parts, break support and actuate simple mechanisms on the buildplate. Revomaker [3] fit a FDM printer with a revolving platform that re-oriented a cuboidal box with pre-installed electrical components to integrate electronics into 3D-printed objects. XPrint [29] augmented a 3-axis CNC machine with an assortment of swappable magnetically attached components to enable additional functionalities, such as syringes for dispensing additional materials and a

magnetic stirrer among others. On the other hand, augmentation is far less common with laser cutters. PacCAM [20] and LaserCooking [2] augmented a laser cutter with a camera to detect material types, but note that this addition only adds sensor and not mechanical functionality.

In general, augmentation carries the benefit of adding additional functionality not present on the original fabrication machine without requiring separate hardware, benefits we were looking to utilize when designing the our process to be carried out strictly with the laser cutter itself and an add-on. To the author's knowledge, our process is the first to augment a laser cutter to provide additional fabrication capabilities, including end-to-end circuit fabrication. XPrint [29], while close to our method in principle, is notably missing a pick-and-place mechanism and as such has no way of leveraging existing components to complete circuits.

III. Our System: Automated Fabrication Pipeline

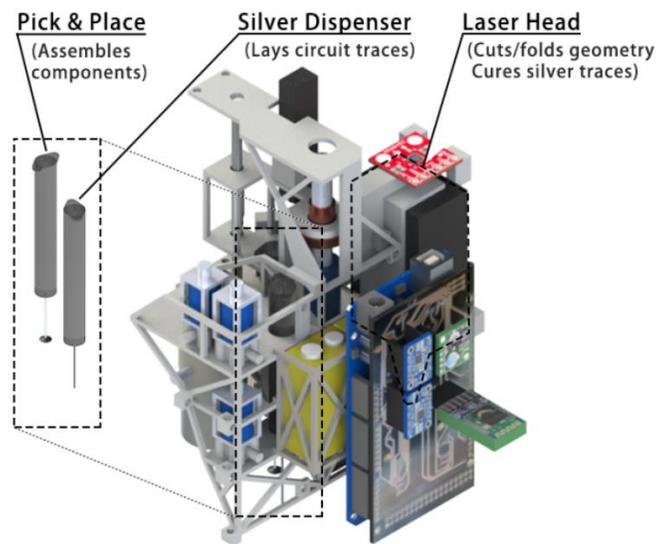


Figure 2. Render of the hardware add-on. The add-on adds silver dispensing and pick-and-place capabilities to the laser cutter, which, combined with the laser cutter’s innate ability to cut and fold acrylic, enables it to fabricate fully-functioning devices.

3.1 Overview

Our method is a new fabrication pipeline that allows novice users to create complete, functional devices with a laser cutter that work immediately after fabrication is complete without requiring any additional assembly by the user. Devices fabricated with this new method are comprised of electronic components connected with soldered silver paste traces mounted on an acrylic base substrate.

Some of this functionality, namely the component placing and trace-making, is not natively supported by the laser cutter. In order to keep our method accessible to a wide audience, we worked to make the method be usable without requiring any modifications to the laser cutter hardware or software. To achieve this, we designed an apparatus that is

externally mounted to the laser cutter head and contains a syringe for silver dispensing, a pick-and-place mechanism, and an Arduino microcontroller (Figure 2).

3.2 Fabrication Process

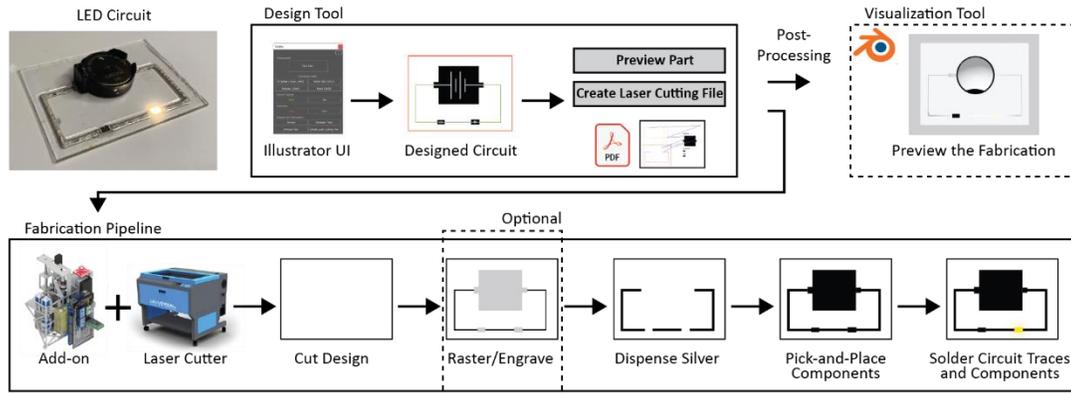


Figure 3. System Pipeline. Design Tool: Users place components and draw geometry and circuit traces, **Post-Processing:** On export, the design file is translated into machine instructions. **Fabrication Pipeline:** The user sends the file to the laser cutter, which cuts the geometry, dispenses silver for circuit traces, pick-and-places components, and then cures the silver to fabricate the device.

Our system automatically fabricates devices using a five-step process. First, the laser is used to cut out the outer shape of the device. Next, a silver solution is dispensed according to the layout of the traces before a pick-and-place mechanism is used to place components from the side into their correct location and orientation in the circuit. Lastly, the laser is used in a defocused state to heat the dispensed silver and “solder” them to create electrical connections, after which the device is functional. We also support fabrication of 3D electronic devices by enabling users to include folds in their designs, leveraging the same technique to fold acrylic using a defocused laser as first described in LaserOrigami [13].

3.3 System Walkthrough

To help users fabricate devices with our method, we provide a 2D design tool (Figure 4), developed as a plugin to Adobe Illustrator, that helps users with defining the circuits and geometry of their device. The interface is intended to abstract away as many of the implementation details of the fabrication process as possible from the user, allowing them to focus purely on the design. As such, the order that the design is constructed in the design tool is different from the order used in the actual fabrication. This interface, along with the post-processing pipeline for translating designs into a laser cutter ready file, is the main focus of this thesis. Now we will walk through how a user designs and fabricates a device using our pipeline.

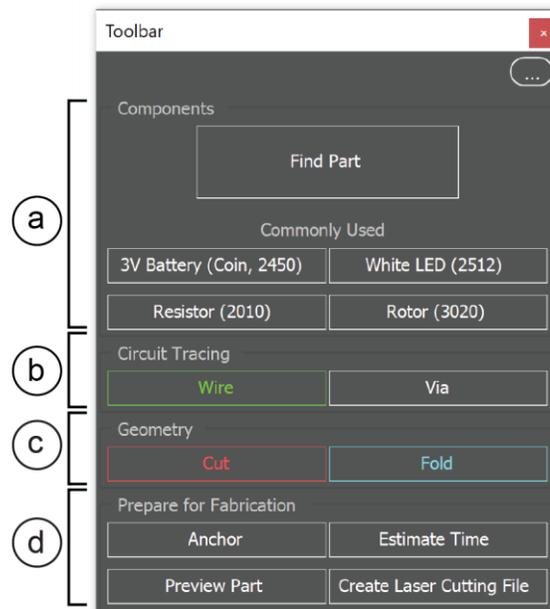


Figure 4. UI Toolbar: a) component placement b) circuit tools c) geometry tools d) export tools.

3.3.1 Designing the Device

Users start by creating the 2D design. First, users place the electronic components of the circuit into the drawing space (Figure 5a) by clicking the ‘Find Part’ button and locating the desired file within the component library. Certain commonly used components are accessible directly from the interface and load the respective component without the need to progress through several menus. Once loaded into the drawing space, components can be freely translated and rotated. Both the component footprints and the electrical symbol of the components are shown so that the user knows both the component’s physical shape and role in the circuit; such practice is common among other circuit design tools [8].

Next, users draw in the circuit traces that connect the components together using the ‘Wire’ tool (Figure 5b). Components have anchor points at the terminals to help guide users with connecting the components in the correct places. Last, the user selects the ‘Cut’ tool to specify where the device should be cut out from the surrounding acrylic as well as any holes if needed (Figure 5c). At this stage, the design is ready for the post-processing step.

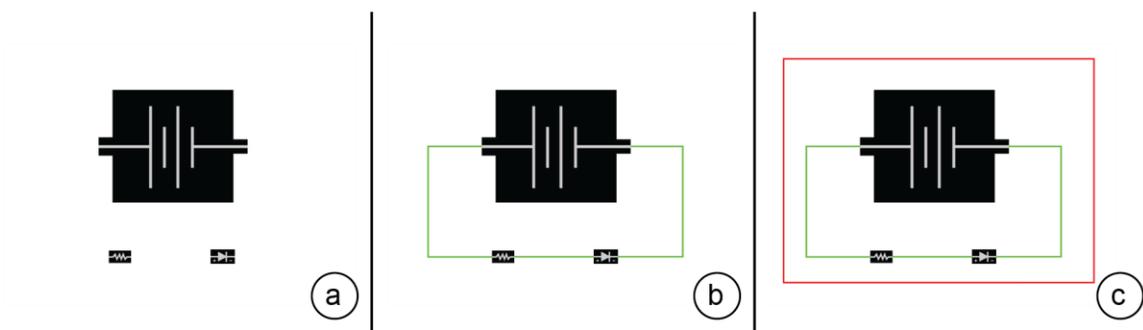


Figure 5. The three main stages of drawing a device: a) placing components b) drawing traces c) defining geometry.

3.3.2 Extras

Our interface supports a few extra features that, although not required, enable a wider range of designs to be fabricated. The Fold and Anchor tools are used to specify folds, which enable the creation of 3D devices (Figure 6a): Users draw in lines where the device should be folded down across with the Fold tool, then specify the face which other areas are folded with respect to by placing the Anchor component within the desired face. The Via (vertical interconnect access) component is used to mark intersections of two intersecting wires where the wires should remain electronically separate, allowing support for higher complexity circuits. Though the single-sided circuits made by our system differ from the double-sided PCB context in which vias are normally found in, we replicate the functionality of keeping intersecting traces electronically separate by crossing a 0Ω resistor with a 0Ω jumper as shown in Figure 6b.

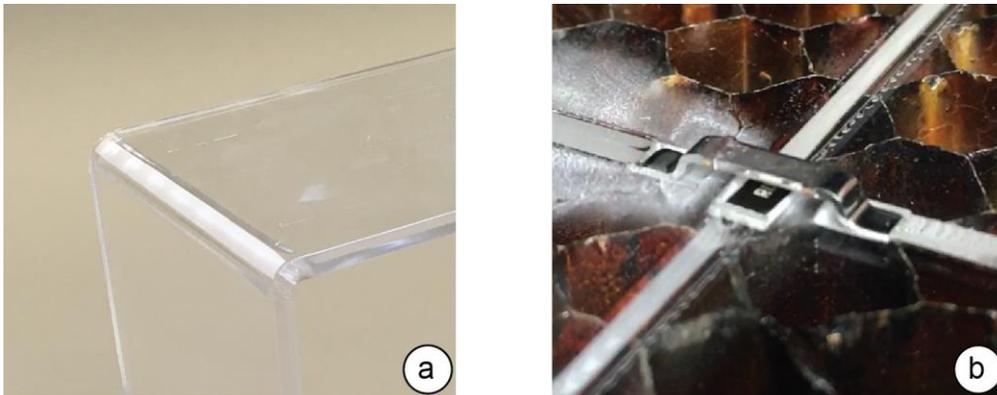


Figure 6. Extra Fabrication Features. a) Folding: enables creation of 3D device designs. b) Via: enables support for higher complexity circuits. In our system, vias consist of a 0Ω resistor crossed with a 0Ω jumper.

There is a settings menu (Figure 7) accessible from the top right button in the interface gives users some extra control in the fabrication process, namely the choice to enable channel engraving and component rastering. If enabled, two extra steps are added to the fabrication pipeline that raster component footprints and engrave channels for the silver

paste to be dispensed into, reducing flow and enabling traces to be closer together. In exchange, the fabrication process takes longer, which may not be necessary for designs with widely-spaced traces. Lastly, before finalizing their design, users can view an estimate of the time it will take to fabricate the component by clicking the ‘Estimate Time’ button.

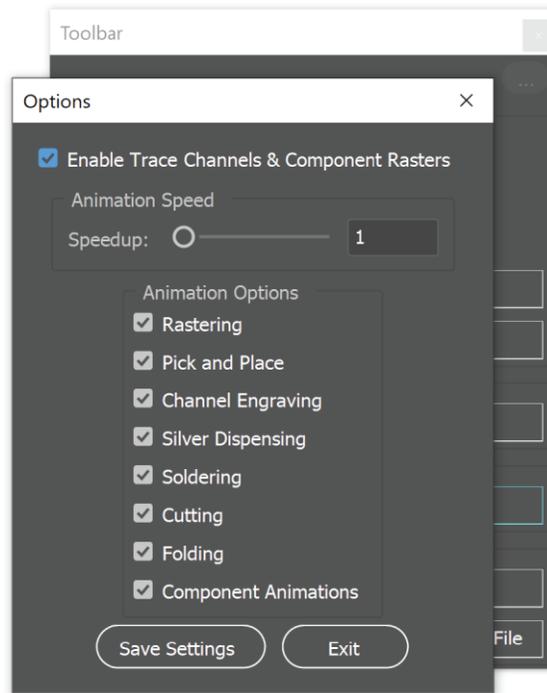


Figure 7. The settings panel. Checking ‘Enable Trace Channels & Component Rasters’ adds extra steps to the fabrication process in exchange for enabling traces to be closer together. Most of the other settings are for the 3D visualization tool, which is the subject of a different thesis.

3.3.3 Export & Post-Processing

On export, the design is translated into machine-readable fabrication instructions via a processing step that parses the design file and manipulates the content for fabrication. The post-processing step can be started automatically via the ‘Prepare Laser Cutting File’ button and requires no further input from the user. Once completed, the resultant file can

be sent directly to the laser cutter, where the device gets autonomously fabricated. Once the last connection is soldered, the device is functional.

IV. Implementation

In this section, we will discuss the implementation details of the software component of the pipeline. This component is broken into two main parts: 1. The UI through which users design their devices and 2. The post-processing pipeline, which takes the device schematic designed by the user and performs the necessary augmentations needed for the laser cutter to fabricate the device.

4.1 User Interface

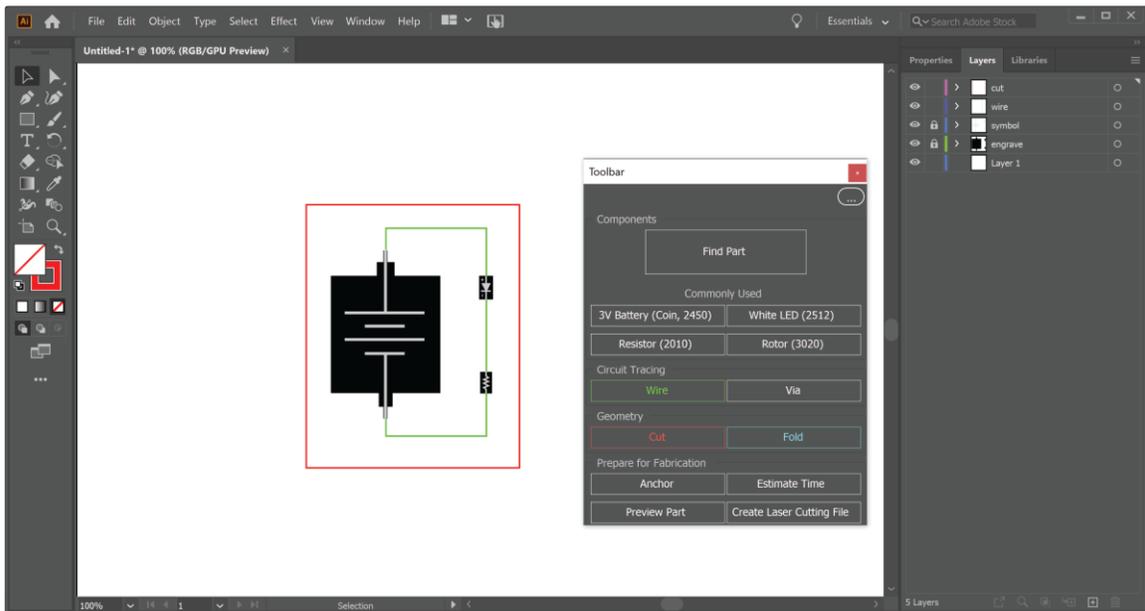


Figure 8. Full view of the user interface.

Our user interface is built on top of Adobe Illustrator. We chose to extend Illustrator over creating a custom interface from scratch in order to leverage the drawing and layer functionalities already implemented in Illustrator. Illustrator is also a widely used drawing tool, which means that many users would already be familiar and would thus avoid needing to learn an entirely new interface. In exchange, we lose some flexibility to

implement certain features more specific to circuit design such as circuit validation. More on this limitation will be covered in the Discussion section.

4.1.1 Toolbar

All functions in the interface are accessible through a supplemental interface that can be opened and executed directly through Illustrator (Figure 8). This interface is created via a script written in Illustrator's native Javascript environment. Operations available through the interface are not executed by the interface script itself; rather, the interface script enables direct execution of other independent Illustrator scripts that carry out the desired action when the appropriate buttons are clicked. The settings window, obtained by clicking the top right button, enables users to directly alter certain internal settings that affect other parts of the fabrication pipeline. The settings are saved to the user's computer locally in a .json file; Each time this settings window is opened, it looks for the .json file to restore any previously saved settings, which enables the user's settings to persist across multiple runs of the toolbar script. This .json file is also read by scripts in other parts of the pipeline to alter certain settings accordingly.

4.1.2 Components

Our library of components are represented as .ai files that contain two layers, engrave and symbol. The engrave layer contains the footprint of the component that is rastered to indicate where the component is placed in the fabricated device. Footprint shapes are grouped together in a group named with the name of the component as an indirect way of storing the component name in the svg for future steps. The symbol layer contains metadata about the component that is useful for the user to see during the circuit design process but does not end up in the final exported file; such metadata includes electrical

symbols, any overhangs on the part that are not in the footprint but would affect the component size, a custom pick point for the pick and place if needed, and the orientation of the part for modeling purposes. Figure 9 shows some examples of component files compared with their real-life counterparts, illustrating how the footprints match the real-life size of their counterparts. These components can be readily placed into the main design file through the use of the Find Part button on our interface, which loads the component data into the clipboard.



Figure 9. Some examples of our component files compared with their real-life counterparts. a) battery b) resistor. The black engrave layer represents the footprint of the component, while the symbol layer contains the component's electric symbol and other metadata. Component footprints match the real-life size of their counterparts.

The Anchor and Via components are special components that are placed in the same way as normal components but serve unique functions and are not retained in the final post-processed file. The Anchor component is meant purely for indicating the reference face in a component with fold lines. Its location is used by certain scripts in the post-processing step to properly prepare the design to be folded and then subsequently removed. The Via component is used to mark wire intersections where the intersecting wires should remain

electrically separate. Its design is meant to emulate wires crossing to the other side as in a PCB editor to more intuitively indicate to users its purpose. During the post-processing step this temporary symbol is removed and replaced with the actual footprint of the via. We chose not to have users place the footprint directly as with the other components because of its unintuitive appearance; given that vias in our system consist of a resistor crossed with a jumper, the user would have to take care not to connect the jumper with the resistor, whereas in all other circumstances the user should be joining component footprints together with wires. In addition, components are generally placed first in our design process, but users would only know where to place vias after finalizing the other component placements and drawing in the traces.

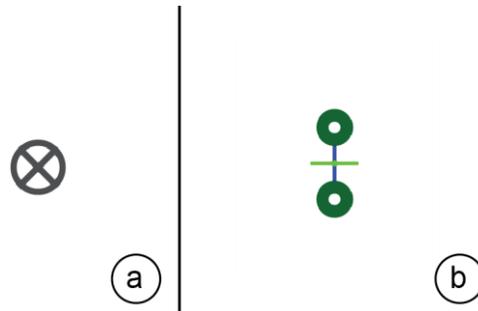


Figure 10. Special Components, a) Anchor b) Via. These components are handled separately in the post-processing pipeline and do not end up in the final file.

4.1.3 Geometry & Traces

Red and green lines represent cut and wire lines, respectively. If folding is desired, cyan lines can be drawn where the main acrylic piece should be folded downwards. The colors for these lines in the interface are chosen to match the colors that these lines should be in order to be performed in the appropriate order by the laser cutter (see Figure 11). When the respective button is clicked, the interface automatically switches the user's active

color and layer accordingly. All cut, wire, and fold lines are expected to be drawn using either the pen or line tools in Illustrator.

Step	Color	Description	Mandatory?	Offset (X/Y mm)
#1a	Red	Cut outline	Yes	(0,0)
#1b	Black	Raster footprints	No	(0,0)
#1c	Green	Engrave channels	No	(0.4,0.7)
#2	Yellow	Dispense silver	Yes	(-1.8,-68)
#3	Blue	Pick-and-Place	Yes	(1.5,3)
#4	Magenta	Solder silver	Yes	(1.5,3)
#5a	Cyan	Folding	No	(0,0)
#5b	Orange	Cut 2nd outline	No	(0,0)

Figure 11. Laser cutter colors and ordering mapped to steps in the fabrication pipeline. Lines in the file must be one of these 8 colors to be executed by the laser cutter.

4.1.4 Time Estimate

The time estimate is obtained by summing up the lengths of lines in each separate layer of the design's svg file, which correspond to the different stages of the fabrication process, and then multiplying the total distance in each layer by a constant factor corresponding to the speed of the laser cutter when completing that step. These constant factors were obtained via timed experimentation with the physical setup. The svg file is the same as that of the post-processed file in order to generate all of the additional lines for summing. The time estimate calculation is implemented in python. Once again, since the interface cannot directly communicate with the python script, a workaround was necessary in which the Illustrator script, after invoking the python script, waits for the python script to write the resulting calculation to a temporary file on disk before reading that file and displaying the value to the user on the interface.

4.1.5 Encoding Information

When the user is finished with their design, their design is saved as a svg file. Each layer in the design is converted into a grouping that contains all shapes and lines in that layer as separate svg elements and encodes the layer name into the id attribute of the group tag. To help the post-processing scripts more easily discern which lines of the svg correspond to which parts of the design, the toolbar scripts ensure that users draw the corresponding parts onto the correct layers.

4.2 Post-Processing

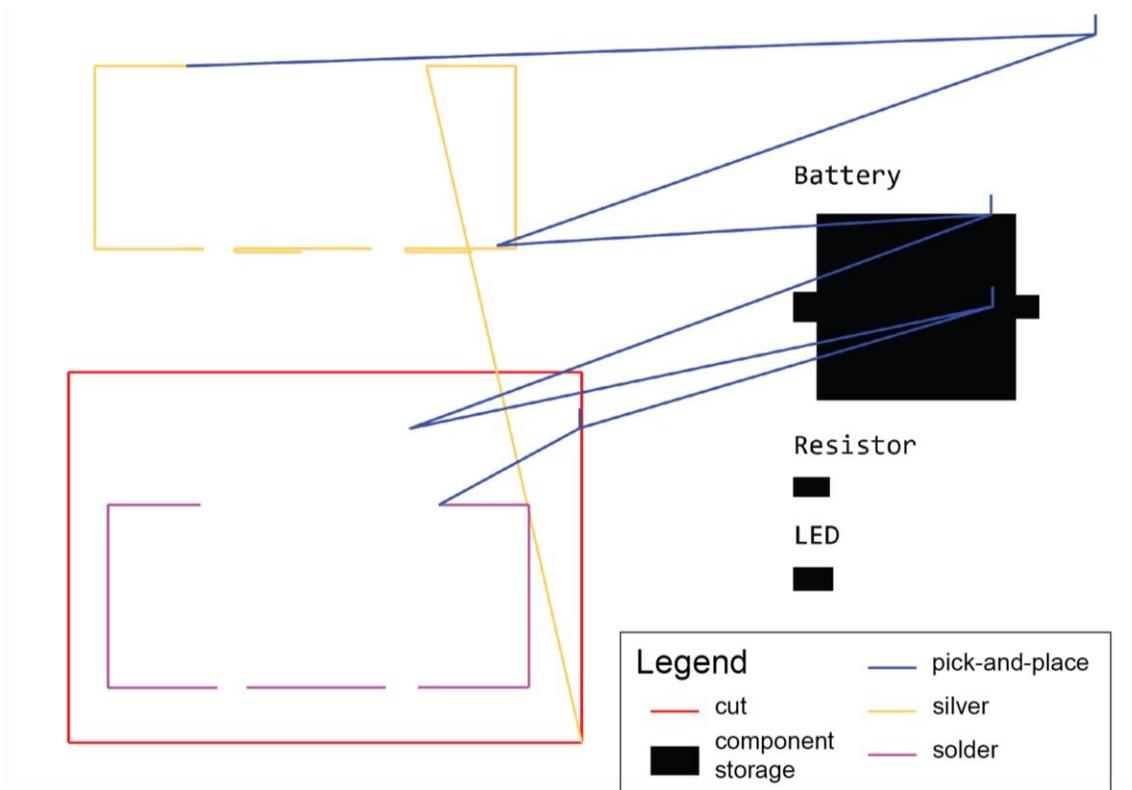


Figure 12. The post-processed file. All paths are .00001pt wide in the actual file but are thickened here for clarity.

The post-processing pipeline takes in the design svg and generates a document with the machine instructions for fabricating the device (Figure 12). All post-processing is done

through a series of scripts written in python, which manipulate the svg file directly and produce a new svg file as output.

The processing is split across several different python scripts that are each responsible for a different step of the process in order to more easily locate any errors in the pipeline; each subsequent script in the pipeline takes in the output svg of the previous step as input. Since it is not possible to call python scripts directly from the Illustrator script Javascript runtime, a workaround was enacted where the needed terminal commands are first written to batch executable file (.bat in Windows, .command in Mac) and executed via running the executable file. The following sections go into more detail about what goes into preparing the laser cutter ready file.

4.2.1 Generating Additional Trace Paths

Fabricating traces is a two-step process involving extruding silver paste along the paths of the traces and soldering them using a defocused laser as the heat source. The user file only specifies the location of these traces with the wire lines, from which additional paths for the silver dispensing and soldering steps must be generated. These paths are required to be .00001pt wide in order to be recognized as a vector by the laser cutter. Lines drawn directly by the user that will end up in the post-processed file are altered in this step to have the appropriate thickness, which is done through direct manipulation of the encoded svg styles; any styles that are a color corresponding to one of the fabrication steps are directly altered to the needed thickness. In addition, styles are inserted for any new colors generated by this step that were not present in the original svg, namely the styles for the silver and solder lines. Figure 13 shows a close-up view of what the silver and solder paths look like in the post-processed file.

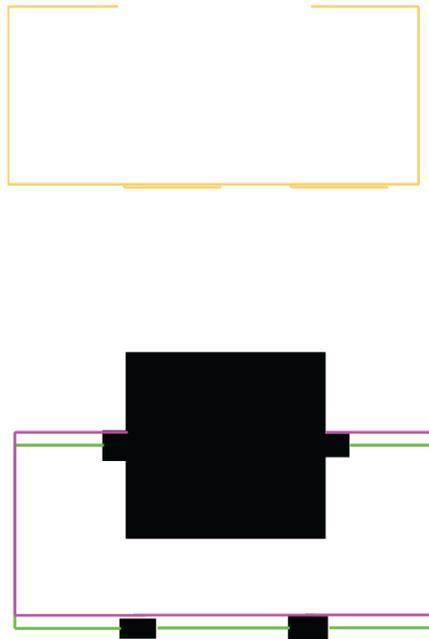


Figure 13. Close-up of silver (yellow) and solder (magenta) paths (component footprints left in for reference). Paths are extended and offset from the original user-defined trace path (green), silver significantly more so than solder. Note the deozing pattern at the end of each silver segment apart from the last, which transitions directly to the pick-and-place lines.

Silver and solder path generation starts with obtaining the raw user-defined wire lines from the svg file. Both types of lines require additional encodings to be inserted at the beginning and end of each continuous stretch of wire. To make inserting these encodings during the output svg writing process easier, the individual lines are grouped into continuous stretches, henceforth referred to as ‘chains’. The same set of chains are used as the base of both the silver and solder lines, with which layer-specific transformations such as offsetting and encodings are applied individually before getting written back. Both types of lines are also extended slightly on both ends of each chain to help better connect the traces with the components.

Silver paths are executed by a syringe mounted externally on the laser head, requiring that the path be offset from the actual trace 1.8mm to the left and 68mm up to account for the location of the silver dispensing syringe on the apparatus. To align with our goal of not requiring any modifications to the laser cutter, the silver dispensing path encodes when to start and stop dispensing with an extra line 3mm in length at the beginning and end of every chain, which creates a motion signal strong and unique enough for an IMU to detect and signal the microcontroller mounted to the apparatus to operate the syringe. Each of these patterns is also followed by a short pause to account for the slight delay for the silver inside the syringe to come out. Each chain is also ended with an additional de-ozing pattern to help the system brush off any excess silver: this pattern consists of a short 0.5mm line perpendicular to the last line of the chain followed by a 15mm line in the opposite direction. When a chain contains at least two lines, the orientation of the last two lines in the chain are used to determine which direction the perpendicular line is written in to minimize the chance of intersecting existing silver lines with the pattern; specifically, the perpendicular line should always be oriented the same direction as the second last line to direct the pattern away from the corner.

Solder paths are offset upwards by 2mm to account for the defocused laser used to solder traces, but are otherwise identical to the base chains.

4.2.2 Pick-and-Place Tray

To support the pick-and-place functionality, the components in circuit are duplicated and shifted to the far right, creating a tray with indentations indicating where the components are meant to be placed in by the user. The indentations are grouped by component types,

complete with a rastered component name, to enable users to more easily identify which engraved footprints belong to which components.

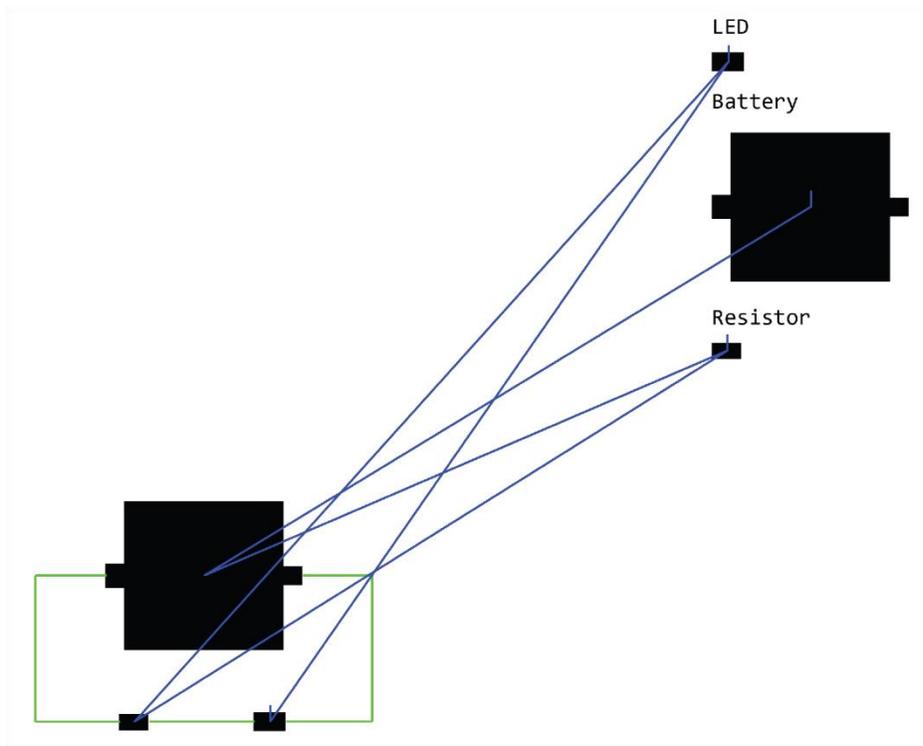


Figure 14. Close-up of tray and pick-and-place paths (normal path offset has been removed here for clarity). Tray features component footprints and labels. Paths direct the laser cutter to and from components in the tray and their proper locations in the circuit.

The script first goes through the lines of the svg to collect the shapes that make up each component's footprints and symbol. The lines corresponding to the footprint are saved in order to be written back later to form the tray, while the symbol lines are discarded after the appropriate information is extracted. For each type of line, the line is parsed in order to determine the shape represented by the line's contribution to the component's bounding box, which is important for ensuring that component footprints are not placed too close to each other in the tray. For the symbol lines, additional metadata such as a customized pick location for the part is also parsed and stored in this stage to be used later.

With all of the components, their bounding boxes, and footprint lines collected, the script next determines the final position of each component in the tray. Prior to this step components are sorted by type in order for them to be grouped together. To ensure that the tray does not overlap any existing elements in the input .svg file, the tray is placed off to the extreme right, outside the original bounds of the canvas; the dimensions of the output .svg are extended accordingly to account for the size of the tray, but this extension amount is not determined until after all component placements are finalized. Starting from the top right corner of the original canvas, components are placed one-by-one at the current reference point. After each placement, the reference point is shifted downwards accordingly by the bounding height of the placed component. If this new point is determined to be outside the vertical range of the canvas, the component is moved to a new column, shifted based on a running tally of the width of components placed in the previous column. Labels are placed into the tray in much the same way as regular components but are only engraved and will not have pick-and-place paths generated for them.

Lastly, the script writes in the paths to direct the pick-and-place to the proper locations from the tray to the component's correct placement in the circuit. Paths consist of both those that guide the laser head from the tray to the circuit and those that guide the laser head after a placement to the next component. By default, components are picked up by their center of mass, determined as the midpoint of the bounding box of their footprint. Custom pick points are encoded directly in the component file in the case for components where the center of mass differs from this midpoint. The order that the components are placed is determined by sweeping the circuit from left to right, then from top to bottom if

there is a tie, to minimize any collisions that may take place. As such, the order that components are engraved into the tray may not necessarily be the order in which they are placed. Figure 14 shows an example of a finished tray and pick-and-place paths, which illustrate the left to right placement order of components.

The paths start and end with 65 0.1mm lines in order to hold the laser head in place and allow time for the pick-and-place to perform the actual picking up and placing of components. These series of very short lines are a workaround to the lack of a native hold function in the laser cutter software; the duration of the hold scales proportionally with the number of line copies, which in this case produces a 12 second pause. The pick end of the path also has a special pattern consisting of a single 3mm line to signal to the Arduino to begin the pick process. To differentiate this pattern from those of the previous silver layer, a special signal was written at the end of the silver layer to indicate that the Arduino should switch to operating the pick-and-place syringe.

4.2.3 Folding

Users are able to, in addition to cutting out their devices, also specify lines at which their devices can be folded to produce a 3D result. Physically, this is done by elevating the acrylic, cutting out the part to be folded, then going over the fold line using several passes of a defocused laser, after which gravity will cause the part to bend downwards at up to a 90-degree angle. Due to the nature of our folding process, folds can only be performed on areas of the acrylic that are, at the time of folding, still parallel with the bed of the laser cutter. As such, the folds have to be executed in an order from the outermost going inwards relative to the plane of reference, the face of the device from which all other folds are made. Similarly, cut lines must be cut out in a certain order to enable folding,

with at least some part of the plane of reference remaining attached to the main acrylic sheet until all folds are completed.

To accommodate this additional feature, the post-processing script infers the fold and cut order directly from the plane of reference marked by the user with the anchor symbol in the interface (Figure 15a). A Breadth First Search (BFS) is used to obtain all of the polygons in the device from the raw svg lines, in which the polygon with an edge closest to the anchor point that still encloses it is determined to be the reference face. From there, all edges that make up the reference face, excluding fold lines, are designated to be anchor lines, lines that are cut last to enable folding. A short length of buffer is also required to be cut into the anchor lines at the beginning and end of each fold line to make room for the actual fold; these sections are calculated by turning the segments of the reference face polygon into a double linked list and splicing the anchor lines 3.5mm away from the fold lines. In the final post-processed file, the fold lines are extended by 5.3mm on each side to ensure that the entire length of the fold is heated. Figure 15b shows the result after processing, including the anchor lines and extended fold lines.

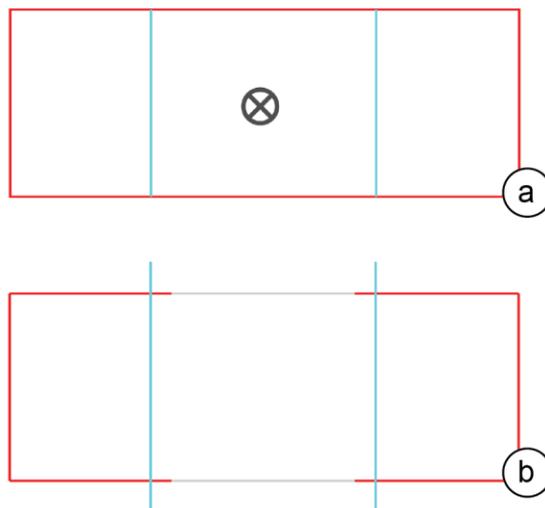


Figure 15. Example fold design. a) the design file: the anchor component specifies the reference face of the device from which other faces will be folded down with respect to. b) post-processed file, with extended fold lines and secondary cut areas (normally orange but recolored for clarity). The anchor component is removed, having fulfilled its purpose. Note that the primary cut areas extend slightly into the reference face to allow room for folding.

In the event that traces intersect fold lines in the design, an additional processing step is required to ensure that the trace is properly conductive over the bend; namely, sections of the trace that are within an immediate vicinity of the fold line must be left unsoldered so that those areas are not double-soldered when the folding is conducted. In terms of the post-processed file this means that there should be breaks within the solder lines in the vicinity of intersections with fold lines. The implementation for this step is very similar to the one used to splice wire lines during the via footprint replacement process, only that the spliced-out length corresponds to a predetermined buffer for folding.

4.2.4 Vias

A separate python script that replaces Via components in the file with the rastered footprint is run only when the input design file contains one or more Via components. The process begins by running an algorithm to find all intersections in the input .svg file. Intersections between two wire lines that are within a 5pt radius of a vias component are flagged to be replaced with a via footprint (Figure 16b). The replacement is a 2-step process in which the footprint shapes are placed and the wire segments spliced accordingly to make room for the footprint (Figure 16c). The footprint itself consists of three rectangles of a predetermined size, of which a copy is inserted directly into the resulting svg file shifted by the appropriate distance such that the footprint is centered over the intersection point. The wire segments involved in the intersection are spliced such that the sections directly underneath the footprint are removed. The resulting pieces

are written back in the same relative order as the original segment in order to preserve the order of the wire lines in later processing steps. From here on out, the via is treated in much the same way as the other components, but with one notable exception: in the pick-and-place tray, it is necessary for the resistor and jumper that make up the via to be placed separately, with the resistor placed before the jumper; as such, the footprints for each respective piece are separate in the final file.

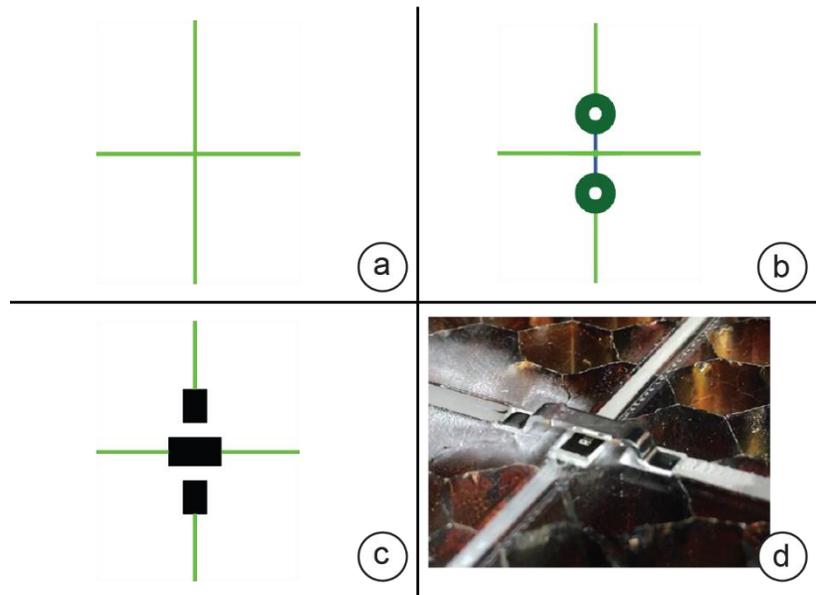


Figure 16. Via pipeline. To include a via in the device, the user a) starts with intersecting wires and b) places the via symbol over the intersection, which then c) gets replaced with the footprint during post-processing. d) shows the finished result.

V. Applications

In this section we demonstrate some examples of devices that can be fabricated end-to-end using our system.

5.1 Quadrotor

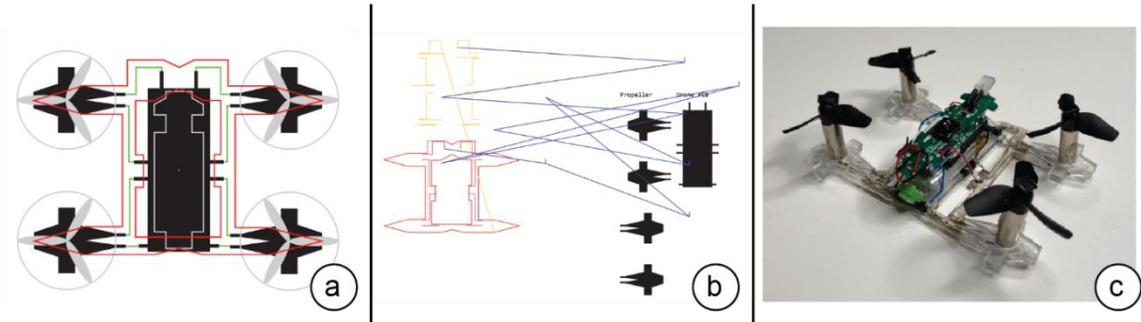


Figure 17. Design progression for the quadrotor. a) design b) post-processed file c) the fabricated result.

Our system was able to make a fully-functional quadrotor that was able to fly directly off the laser cutter bed. The components in this quadrotor were obtained by disassembling an existing quadrotor and reconfiguring them slightly so that they can be properly placed and connected using our system, but by showing that automatic fabrication is successful we demonstrate that quadrotors can be made from scratch given the appropriate pre-assembled components.

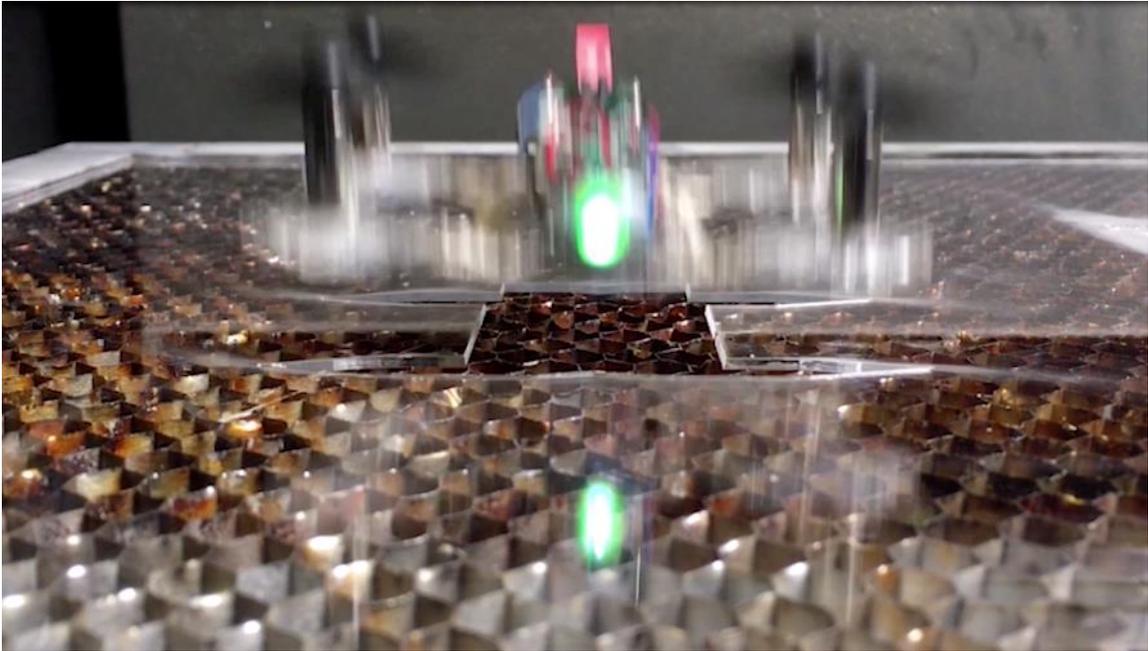


Figure 18. Quadrotor flying directly off the fabrication platform.

5.2 Reading Light

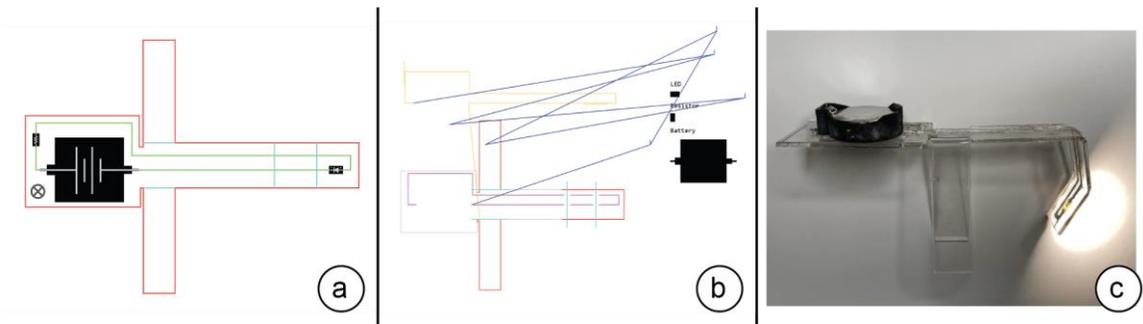


Figure 19. Design progression for the reading light. a) design b) post-processed file (note: gray lines represent anchors and are normally colored orange but have been recolored for clarity) c) the fabricated result.

A simple LED circuit that demonstrates our system's ability to automatically make folded designs, including those where traces go over fold lines. The part can be placed on top of a laptop screen to light up the keyboard when working in a dark environment. Note that the fold angles usually end up being less than 90 degrees in practice.

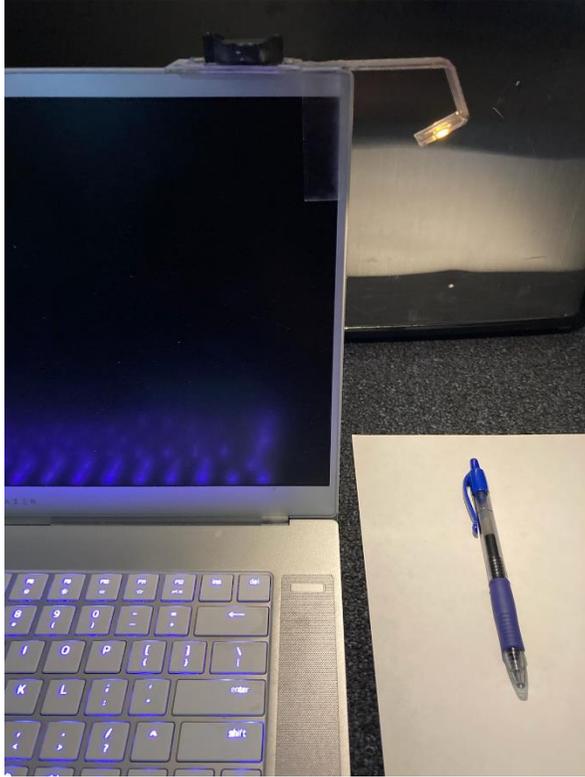


Figure 20. Reading Light in use to light up a laptop keyboard.

VI. Discussion

Our system was successful at making the electronic device fabrication process more streamlined by automating several steps that would otherwise each take significant time and practice to do manually. Folding a sheet of acrylic manually, for instance, takes both additional equipment and practice to get right; the user would need some way of heating the acrylic along the bend area uniformly and must also be careful not to execute the bend too quickly in order to avoid uneven folding. Soldering traces also requires users to have a full set of soldering equipment and takes time even for experienced users.

More importantly, the process is accessible to any user who has access to a laser cutter. The add-on is the only piece of additional hardware needed, which has a total part cost of \$150 and is made entirely up of easily acquirable commercial off-the-shelf items mounted on a 3D printed scaffold. The interface is built on top of an already widely used drawing tool, saving some users the overhead of learning an entirely new system. Lastly, the entire fabrication process is automated after the design phase, eliminating the need for extensive knowledge about its inner workings.

6.1 Limitations

Although our system is powerful and can fabricate a wide variety of designs built with our interface, the system is subject to the following limitations:

6.1.1 Well-Formed Drawings

Since users can freely design their devices in Illustrator, the scope of possible designs is much larger than the scope that we can reasonably account for in our post-processing

pipeline. Our tools, including the post-processing step and the time estimate feature, rely on the design being reasonably drawn, meaning that:

- Component rotations and wire lines are limited to 90-degree orientations
- For a device that has folding, the anchor component must be present and placed within a face of the device that can reasonably serve as the reference face
- Fold lines are drawn only in places where the acrylic can reasonably be folded down with the given cut lines

Failure to adhere to these guidelines currently results in the post-processing pipeline failing and being unable to generate a valid laser-cutter-ready file. As a research-oriented project, it was outside our scope to provide extensive error handling, but for future work it is definitely possible to extend the system to give users more detailed feedback should their designs not meet the requirements.

6.1.2 Fabrication Constraints

There are inherent hardware limitations on what types of designs can successfully be fabricated using our method. The following invalid designs currently pass the post-processing step but cause problems during the fabrication process:

- Overlapping components
- Components placed across fold lines
- Wire lines drawn over components

There are currently no safeguards in place in the software pipeline to preemptively stop the user from fabricating invalid designs that pass the post-processing step. As such, though it was one of our objectives to abstract the details of the fabrication process away from the design phase, the user still needs to take care when designing their devices to

avoid the above failure conditions. Most of these conditions, however, are issues that arise in circuit design in general and are not specific to our fabrication method.

6.2 Future Work

6.2.1 Circuit Validation

Our interface contains no innate checking on user designed circuits, meaning that it is up to the user to verify whether their circuit actually works as intended. As multiple programs made in the past have already featured circuit validation, it was not a high priority to implement circuit validation in the the design tool, especially given that Illustrator offers no innate support for drawing validation. However, as it has already been shown previously [8, 11] that circuit validation features help with learnability in novice users, it could become a feature we implement in the future as well.

6.2.2 Multi-Layer Structures

Currently electrical components are the only objects that are pick-and-placed by the fabrication pipeline, but in theory a much wider variety of objects, including pieces of acrylic, can also be placed with the pick-and-place provided they weigh less than 65g. One possible line of future work thus is to extend the pick-and-place functionality such that devices can be assembled from multiple pieces of acrylic stacked together, akin to the method first described in LaserStacker [24]. The pick-and-place script in the post-processing pipeline would need to be modified to allow for pick locations to be defined anywhere in the design, not just being limited to the tray, but doing so would enable our system to fabricate a larger assortment of 3D geometries including multi-layered devices.

VII. Conclusion

We presented a complete, novel fabrication pipeline that creates fully-functional electric devices that require no additional manual assembly. We showed that our interface enables users to design and fabricate their own complete devices without requiring the user to have extensive knowledge about the inner workings of our fabrication process. The pipeline, while not unlimited in terms of design capability, is accessible to the average laser cutter user and saves valuable time by automating key parts of the fabrication process. The work covered in this thesis represents a step forward in creating automated fabrication systems that are both accessible and capable of producing high-fidelity devices.

References

1. Patrick Baudisch, and Stefanie Mueller. Personal Fabrication. In *Foundations and Trends® in Human–Computer Interaction* 10, 3–4, 165-293, 2017.
2. Kentaro Fukuchi, Kazuhiro Jo, Akifumi Tomiyama, and Shunsuke Takao. Laser cooking: a novel culinary technique for dry heating using a laser cutter and vision technology. In *Proceedings of the ACM multimedia 2012 workshop on Multimedia for cooking and eating activities (CEA '12)*, 55–58.
3. Wei Gao, Yunbo Zhang, Diogo C. Nazzetta, Karthik Ramani, and Raymond J. Cipra. 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.
4. Daniel Groeger and Jürgen Steimle. LASEC: Instant Fabrication of Stretchable Circuits Using a Laser Cutter. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*, Paper 699.
5. Nathaniel Hudson, Celena Alcock, and Parmit K. Chilana. Understanding Newcomers to 3D Printing: Motivations, Workflows, and Barriers of Casual Makers. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*, 384-396.
6. Shohei Katakura, Yuto Kuroki, and Keita Watanabe. A 3D Printer Head as a Robotic Manipulator. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*, 535-548.
7. Yoshihiro Kawahara, Steve Hodges, Benjamin S. Cook, Cheng Zhang, and Gregory D. Abowd. Instant inkjet circuits: lab-based inkjet printing to support rapid prototyping of UbiComp devices. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing (UbiComp '13)*, 363-372.
8. André Knörig, Reto Wettach, and Jonathan Cohen. Fritzing: A Tool for Advancing Electronic Prototyping for Designers. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction (TEI '09)*, 351-358.
9. Mannu Lambrichts, Jose Maria Tijerina, Tom De Weyer, and Raf Ramakers. DIY Fabrication of High Performance Multi-Layered Flexible PCBs. In *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '20)*, 565–571.
10. Hod Lipson and Melba Kurman. Factory @ Home: The Emerging Economy of Personal Fabrication. *Report Commissioned by the US Office of Science and Technology Policy*, 2010.
11. Joanne Lo, Cesar Torres, Isabel Yang, Jasper O’Leary, Danny Kaufman, Wilmot Li, Mira Dontcheva, and Eric Paulos. Aesthetic Electronics: Designing, Sketching, and Fabricating Circuits Through Digital Exploration. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*, 665–676.

12. David A. Mellis, Sam Jacoby, Leah Buechley, Hannah Perner-Wilson, and Jie Qi. Microcontrollers as Material: Crafting Circuits with Paper, Conductive Ink, Electronic Components, and an "Untoolkit". In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction* (TEI '13), 83-90.
13. Stefanie Mueller, Bastian Kruck, and Patrick Baudisch. LaserOrigami: laser-cutting 3D objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '13), 2585-2592.
14. Hyunjoo Oh, Tung D Ta, Ryo Suzuki, Mark D Gross, Yoshihiro Kawahara, and Lining Yao. 2018. PEP (3D Printed Electronic Papercrafts): An integrated approach for 3D sculpting paper-based electronic devices. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (CHI '18), Paper 441.
15. Simon Olberding, Michael Wessely, and Jürgen Steimle. PrintScreen: fabricating highly customizable thin-film touch-displays. In *Proceedings of the 27th annual ACM symposium on User interface software and technology* (UIST '14), 281-290.
16. Simon Olberding, Sergio Soto Ortega, Klaus Hildebrandt, and Jürgen Steimle. Foldio: Digital Fabrication of Interactive and Shape-Changing Objects With Foldable Printed Electronics. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (UIST '15), 223-232.
17. Huaishu Peng, François Guimbretière, James McCann, and Scott Hudson. A 3D Printer for Interactive Electromagnetic Devices. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (UIST '16), 553-562.
18. Varun Perumal C and Daniel Wigdor. Printem: Instant Printed Circuit Boards with Standard Office Printers & Inks. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (UIST '15), 243-251.
19. Raf Ramakers, Kashiyap Todi, and Kris Luyten. Paperpulse: An integrated approach for embedding electronics in paper designs. In *Proceedings of the 2015 CHI Conference on Human Factors in Computing Systems* (CHI '15), 2457-2466
20. Daniel Saakes, Thomas Cambazard, Jun Mitani, and Takeo Igarashi. PacCAM: material capture and interactive 2D packing for efficient material usage on CNC cutting machines. In *Proceedings of the 26th annual ACM Symposium on User Interface Software and Technology* (UIST '13), 441-446.
21. Valkyrie Savage, Xiaohan Zhang, and Björn Hartmann. Midas: Fabricating Custom Capacitive Touch Sensors to Prototype Interactive Objects. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (UIST '12), 579-588.
22. Martin Schmitz, Martin Stitz, Florian Müller, Markus Funk, and Max Mühlhäuser. /trilaterate: A Fabrication Pipeline to Design and 3D Print Hover-, Touch-, and Force-Sensitive Objects. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (CHI '19), Paper 454.
23. Saiganesh Swaminathan, Kadri Bugra Ozutemiz, Carmel Majidi, and Scott E. Hudson. FiberWire: Embedding Electronic Function into 3D Printed Mechanically Strong,

- Lightweight Carbon Fiber Composite Objects. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (CHI '19), Paper 567.
24. Udayan Umapathi, Hsiang-Ting Chen, Stefanie Mueller, Ludwig Wall, Anna Seufert, and Patrick Baudisch. LaserStacker: Fabricating 3D Objects by Laser Cutting and Welding. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (UIST '15), 575–582.
 25. Nobuyuki Umetani and Ryan Schmidt. SurfCuit: Surface-Mounted Circuits on 3D Prints. In *IEEE Computer Graphics Applications* 38, 3, 2017, 52-60.
 26. Alexander D. Valentine, Travis A. Busbee, John William Boley, Jordan R. Raney, Alex Chortos, Arda Kotikian, John Daniel Berrigan, Michael F. Durstock, and Jennifer A. Lewis. 2017. Hybrid 3D printing of soft electronics. *Advanced Materials* 29, no. 40: 1703817.
 27. Voxel8. 2019. Retrieved November 21, 2019 from = <https://www.voxel8.com/>
 28. Chiuan Wang, Hsuan-Ming Yeh, Bryan Wang, Te-Yen Wu, Hsin-Ruey Tsai, Rong-Hao Liang, Yi-Ping Hung, and Mike Y. Chen. CircuitStack: Supporting Rapid Prototyping and Evolution of Electronic Circuits. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (UIST '16), 687-695.
 29. Guanyun Wang, Lining Yao, Wen Wang, Jifei Ou, Chin-Yi Cheng, and Hiroshi Ishii. XPrint: A Modularized Liquid Printer for Smart Materials Deposition. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (CHI '16), 5743–5752.
 30. Junichi Yamaoka, Mustafa Doga Dogan, Katarina Bulovic, Kazuya Saito, Yoshihiro Kawahara, Yasuaki Kakehi, and Stefanie Mueller. FoldTronics: Creating 3D Objects with Integrated Electronics Using Foldable Honeycomb Structures. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (CHI '19), Paper 628.