3D-Printable Electronics - Integration of SMD Placement and Wiring into the Slicing Process for FDM Fabrication

Florens Wasserfall, Daniel Ahlers, Norman Hendrich, Jianwei Zhang
Department of Informatics, Group TAMS, University of Hamburg, Germany
{*wasserfall, 2ahlers, hendrich, zhang*}@*informatik.uni-hamburg.de*

Abstract

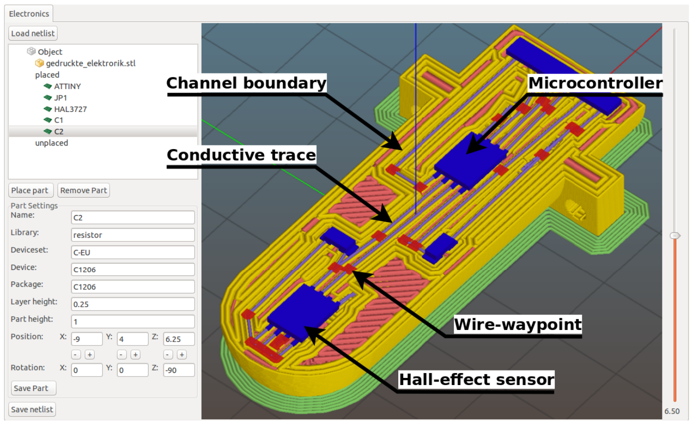
Several approaches to the integration of wires and electronic components into almost every existing additive fabrication process have been successfully demonstrated by a number of research groups in the last years. While the pure mechanical process of generating conductive wires inside of a printed object has proved to be feasible, the design, integration, routing and generation of toolpaths is still a laborious manual task.
In this paper, we present a novel approach to place and wire SMDs in a three-dimensional object, based on schematics generated by conventional PCB design tools such as CadSoft EAGLE. Routing wires in an object for FDM manufacturing requires certain knowledge about the printer's properties to meet the extruder characteristics, avoid non-fillable regions and electric shorts. Correspondingly for the slicing of conductive wires, the software must respect appropriate channel widths, avoid interrupted traces and ensure proper endpoints serving as contact pads for the SMDs. To fulfill those requirements, we implemented the design and routing software as a native extension of an existing slicing software. The user works in a three-dimensional representation of the final extruder toolpath, augmented by the routing information. The actual computing step is executed at the layer level by manipulating the polygons which represent the two-dimensional object topology and toolpath for each single layer, allowing the routing algorithm to avoid the generation of non-printable traces. We successfully designed and printed some test objects including a force-sensor prototype, demonstrating a significant improvement in the usability and efficiency over manual solutions.

1. Introduction

The integration of electric connections and electronic components into additively manufactured parts is considered a key feature for the application in various fields such as wearable devices, (soft)-robotics or prototyping applications. The desire for printing conductive materials has been addressed with an increasing number of different approaches over the past decade. Direct writing of conductive, polymer based silver inks has been successfully demonstrated for SLA [1, 2], and FDM [3, 4, 5] printed objects and free-form surfaces [6]. The ink is dispensed through a nozzle directly to the object's surface from a pneumatic pump or a syringe. Embedding of wires or fibres is a second common technique where a copper wire is either injected into the printed plastic surface by a heated nozzle [7] by ultrasonic welding [8, 9] or by encapsulation into the fluid plastic immediately after extrusion [10]. Other approaches explore the use of inkjettable inks [11, 12, 13] or carbon filled filament [14]. A recent comprehensive review is given by [15].

Most of the work mentioned above is still limited to simple proof of concept applications where the circuit design and toolpath generation can be done manually or with simple scripts. The integration of circuits with higher complexity clearly requires appropriate design tools. A simple solution for this problem is the manual design of channels and cavities with common CAD-tools e.g. Solidworks or OpenSCAD. The object is then printed on a normal printer and the conductive traces are applied in a post processing step [3, 1] or exported as a multi-material model which can be printed directly by a dual-extruder printer [5]. Bayless et al. used a MatLab script to generate G-code from parametric curves and rectilinear patters [7]. It is not stated how the curves and patterns are generated. Macdonald et al. introduced the concept of deforming flat circuit designs created with common PCB design tools [16]. They demonstrate applications where a flat circuit is wrapped around a cylindrical object and an “unfolded” cube which represents the six sides of a gaming dice. In 2015, Swensen et al. presented the concept of computing a system of tubes with varying diameters from a 2-dimensional board layout created in EAGLE [17]. Through-hole components are inserted through connected holes and the tubes are flooded with injected low-melting metal alloy. The flow is controlled by varying friction determined by the tube diameter and length to reach every pin simultaneously. The commercial solution *Project Wire* [18] was announced by Autodesk and Voxel8 and is currently available in a public alpha version. A set of predefined electronic components can be integrated into STL-files and connected via traces in a web-based drag-and-drop application. The result is exported as two separate meshes, one representing the electric wires, the other representing the difference of the original object and the wire traces.

This paper introduces the concept of a design software, aiming to close the gap between available printing processes and common CAD tools for object and circuit design. It is implemented as an extension of the open source slicing software Slic3r [19]. The general idea is to load an existing object mesh and a schematic created by a common PCB design tool and place components at different layers of the sliced object, similar to the layout step for a PCB board as pictured in Fig. 1. The user can vertically “browse” through the object and place components at arbitrary positions inside the object or at its surface. Unrouted connections are visualized as rubberbands. High level routing is achieved by setting waypoints, which serve as interpolation points for the low level routing to generate extrusion paths, approximating the desired route as close as possible while optimizing physical printability.



Part Settings
Name: C2
Library: resistor
Deviceset: C-EU
Device: C1206
Package: C1206
Layer height: 0.25
Part height: 1
Position: X: -9 Y: 4 Z: 6.25
Rotation: X: 0 Y: 0 Z: -90

Figure 1 – Electronic components embedded into a custom 3D-object for FDM manufacturing, showing the wiring layer of a humanoid robot (shoulder) ball-joint with integrated position sensing. Visible are the 3-axis hall-effect sensor at the center of the ball joint (blue, lower left), the microcontroller (blue, near the center), and some passive SMD components. Our slicing tool shows the user-defined wiring waypoints (red) and automatically calculates the wiring extrusion (light blue), channel boundaries (yellow) and infill (purple). Also shown is the printbed adhesion layer (green) required for FDM manufacturing with ABS material.

2. Design process

Manufacturing an integrated 3-dimensional circuit generally requires a model of the object, a circuit design and a transformation into an executable toolpath (slicing). The integration of model and circuit could potentially be implemented into one of the corresponding tools: CAD software, PCB design tool or slicer, or as an additional standalone application. This section motivates and explains our design decisions for the integration of all three aspects.

3D-Electronics design approaches

Creating 3D-electronic designs manually with a CAD software is only feasible for simple test objects since it is laborious, very time consuming and has no reference to any electronic component data.

A more promising approach is the creation of PCB layouts with a PCB designer and either deforming or combining them with a CAD software or to use multilayer PCB circuits with a subsequent registration step that maps each layer of the PCB to a corresponding slice. Both methods can benefit from the well-engineered PCB design software and use features like the existing component databases, electronic rule checking and simulations. However, this approach does not produce real 3D electronics because the final circuit is still limited to a combination or deformation of 2D PCB designs.

Standalone 3D electronic design solutions can simply design real 3D electronics without limitations for component positions and wire paths. They are very flexible due to their independence from surrounding tools like the slicer or the CAD tool. A problem that all these approaches have in common is the missing reference to the actual structure of the print. Without this reference, the designer cannot react to structural differences that 3D printed objects have and the slicer is not able to handle wires and components differently. If slicing information is available, it can be used to ensure a predefined wire thickness, to prevent shorts, to create proper SMD pads, and to ensure the connection on intersections of wires without overfilling them. *Project wire* is currently the only implementation of this approach. However, it has no schematic design feature; the designer has to create the circuit connections while he places the components inside the object. This can lead to missing connections or wrongly connected components.

Implementation considerations

Since all the existing 3D electronic design approaches have some advantages, the new process tries to combine the advantages of the existing approaches. A PCB design software is used for the schematic design, enabling us to use the existing functions and the component database. However, the PCB design software is incapable of designing actual 3D electronic circuits. Therefore, placing of components and wire routing is implemented as an extension to an existing slicing software. During the slicing step, all relevant information is available: the CAD- and PCB-Data and the actual structure of the resulting print.

The proposed process (Fig. 2) starts with the specification of the desired device. Then the CAD design and the schematic design of the circuit can be done simultaneously. The 3D-model and the schematic are imported into the slicing tool. The slicer repeatedly runs through the slicing cycle every time a layer changes. The changes in the layers are triggered by adding components or wires. The cycle starts with the slicing of the whole model and generating the structures for each slice, beginning with the bottom slice. The slicer first generates the perimeter of the object. Then the cavities for the components are generated if they intersect the slice. Wires and wire channels are generated if they are in the slice. Lastly the remaining space is filled with infill structure. When the design is finished the G-code can be exported. The modified slicer is designed to handle different schematics file formats. Due to its simple XML file format, EAGLE was used as a PCB design tool. To use another PCB design software, a simple converter has to be written that converts the schematic file format to the internally used format. Additional information regarding the formats can be found in previous work [20]. To save the design, only the additional information of the placed components and the wires is stored in a new file. The information which is extracted form the schematic file is just referenced in the saved file. This file structure allows editing the schematic with the PCB design tool in the whole process.

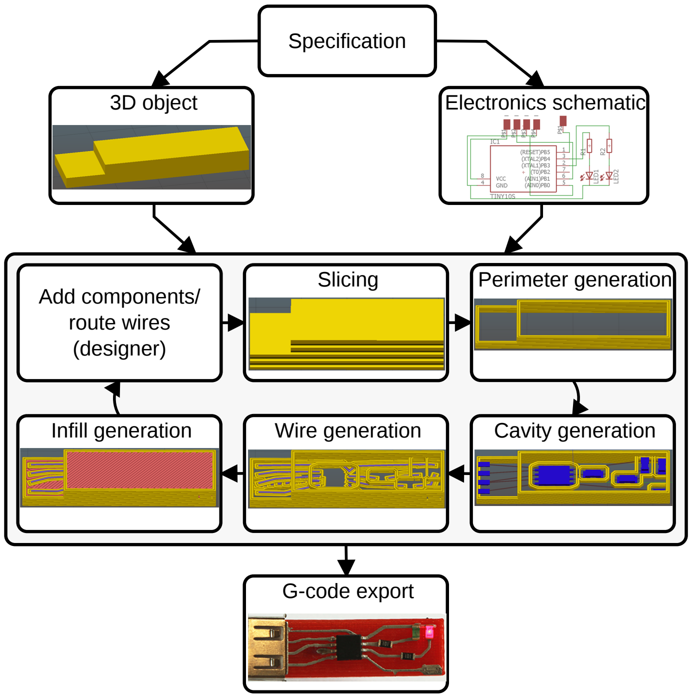


Figure 2 – Overview of the iterative design cycle for 3D-printable electronics

3. Embedding of SMD-components

A 3D-model of each component is required to visualize and place the components in the object. Since EAGLE does not provide those 3D models, they have to be generated during the import step. To generate the 3D models, the outline of each component and the footprint is expanded to a 3D model. The height of the footprint relies on the layer thickness from the sliced model to ensure a proper connection to the ink trace. The height of the component is set to a predefined value that can be set individually for each package. A database of standard package dimensions would simplify this step for the user and could also help render complex components. To place a component, the designer has to select a layer of the object and place the part on this layer. The component can then be rotated around all three axes.

A cavity for each component is generated by first computing a polygon of the component's shape, offsetting this outline by a user defined amount to provide some tolerance and finally subtracting the polygon from the layer, which is also represented by a number of nested polygons [21, 22]. This step is repeated for every layer intersecting the current component (Fig. 3.1).

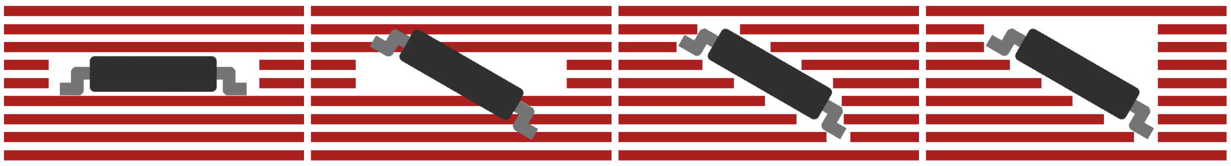


Figure 3 – Subtraction of SMD components from the printed object to create cavities. From left to right: **1**: outline subtraction, **2**: part rotated but subtracted outline generated from flat footprint, **3**: naive polygon subtraction with rotation, **4**: accumulating polygon subtraction with rotation.

The naive outline subtraction cannot handle rotation around the x or y axis (Fig. 3.2). To handle rotated components, the polygon of the rotated component has to be computed and subtracted from the object (Fig. 3.3). For components which are not located at the surface of the object, the cavity must be extended along the z axis to allow placing of the component without extruder collisions. This can be achieved by subtracting the union of all polygons from intersections with layers below the current layer (Fig. 3.4). With this extension it is possible to insert the component into the object directly before the cavity is covered.

4. Wire generation

All wires connecting the individual components are organized as nets by the PCB design tool. Each net contains all pins with equivalent electrical potential and represents the physical connections between this specific subset of pins. In our tool, the "rat's nest" of placed but currently unwired components is visualized by a set of rubberbands. Each net is represented by a set of points and a set of connections between these points, forming an acyclic undirected graph. Rubberbands, points and parts are represented in the GUI as active elements to provide user interactions. The resulting wires are visualized as extrusion objects embedded into the toolpath representation but highlighted by a different color (compare Fig. 4).

Routing concepts

The routing of wires through the 3-dimensional object is divided into two steps: the high-level generation of "waypoints" to define the desired routing paths and the low-level routing of a single line between each tuple of waypoints. This concept allows for a certain level of decoupling between the classic routing task of finding a collision free partitioning of the available space given a number of constraints, and the process related task of finding a set of suitable extrusion paths given a specific printer's parameters. The waypoints are currently manually defined by the user by manipulating the rubberbands in the 3D-visualization. This can easily be extended to import a set of waypoints from an external 3D-electronics design tool or a routing algorithm.

Low level routing

The low level routing step computes a set of extrusion skeletons for each layer from the 3-dimensional set of waypoints. This process is similar to the slicing step which divides an object into a stack of 2.5-dimensional layers by computing the intersection of a set of planes with the tesselated mesh representation. For a given layer L and each wire segment (waypoint tuple) $[a, b]$, the inter-

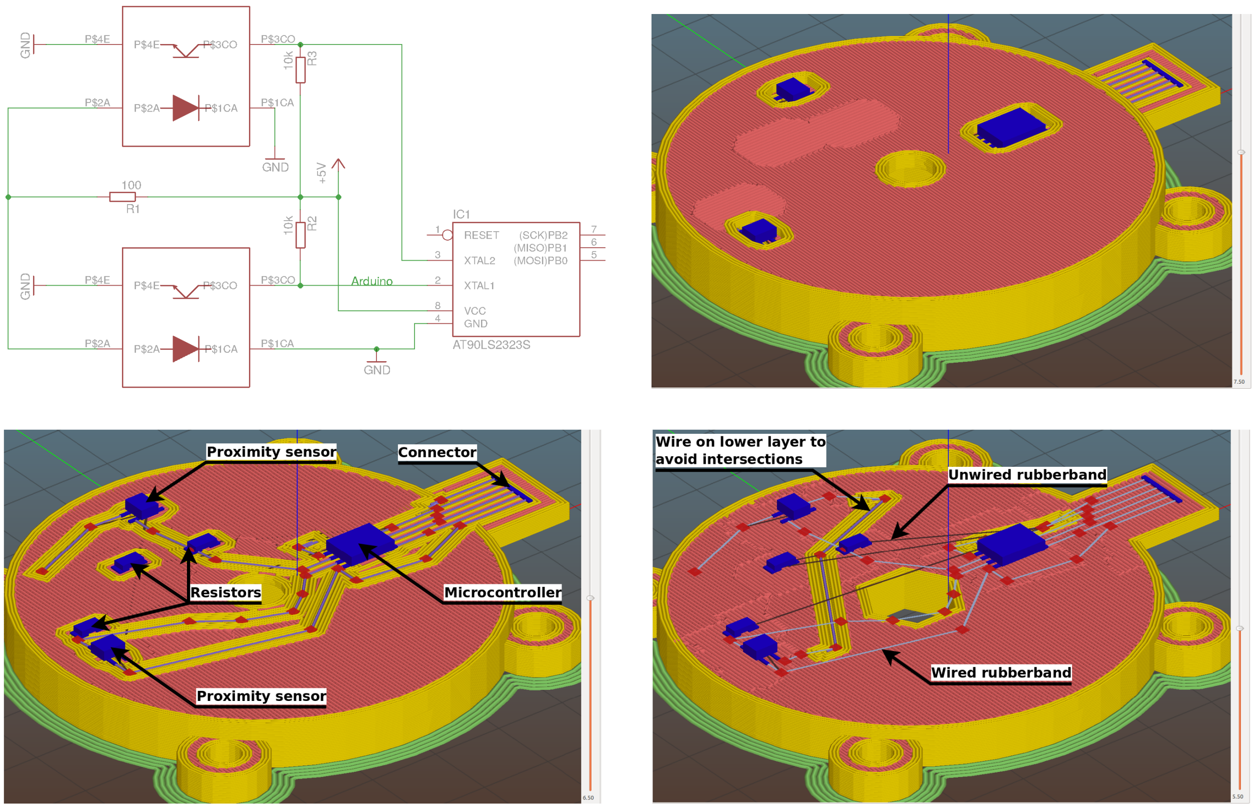


Figure 4 – Prototype of a 2DOF force sensor. **top-left**: EAGLE schematic of the circuit consisting of a microcontroller and two optical proximity sensors. **top-right**: Upper slicing layer, the top of the two optical sensors and the microcontroller and their cavities are visible. **bottom-left**: Middle slicing layer containing most of the components and wires. Waypoints (red) and conductive wires (purple) are visible. **bottom-right**: Lower slicing layer, unwired (black) and wired (gray) rubberbands are also displayed on this layer. Two wires have been moved to this layer to avoid intersections.

section with the lower and upper boundary L_b, L_t is computed by

$$a'_x = \begin{cases} a_x + (b_x - a_x) \cdot \frac{L_b - a_z}{b_z - a_z} & \text{if } a_z < L_b \\ a_x + (b_x - a_x) \cdot \frac{L_t - a_z}{b_z - a_z} & \text{if } a_z > L_t \end{cases}$$

and correspondingly for a'_y and point b, resulting in a 2-dimensional set of lines. This step is trivial for horizontal wire segments, where the intersection equals the planar projection of the original wire. For inclined wires, the start point of the resulting line is then extended by a user defined offset to achieve a stable inter-layer connection to the adjacent wire section if a is outside the boundaries of L.

Extrusion paths

The result of the low level routing step is stored as an unordered vector of lines (as depicted in Fig. 5). Each extrusion path that can be printed as a continuous line is assembled by:

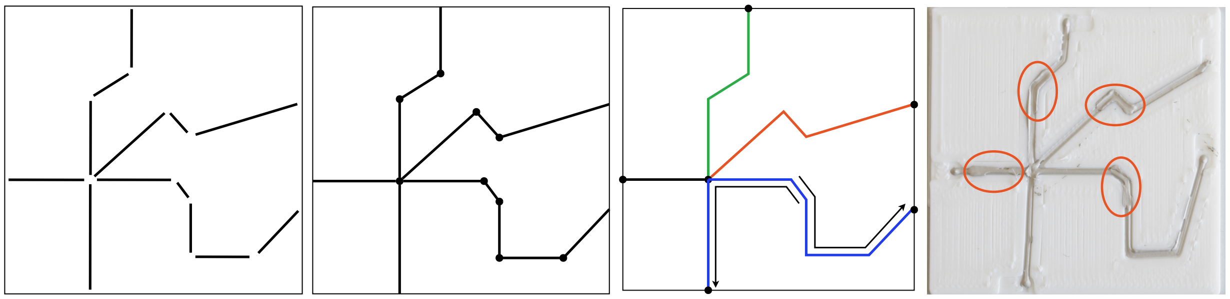


Figure 5 – Generation of extrusion paths for a single layer. From left to right: **1**: Wire segments from slicing. **2**: Assembling a connected graph from unordered segments. **3**: Re-fragmentation at junctions, thin black arrows depict directions of overlapping extrusions. **4**: Printed layer. Overlapping regions to ensure extrusion towards the SMD-pads are marked in red.

1. Finding an endpoint. Iterate the vector until the first line with at least one endpoint which doesn't coincides with any other endpoint. Remove this line from the vector.
2. Traversing adjacent lines until finding a point which coincides with 0 (endpoint) or more than 1 (intersection) other points and removing the lines.
3. Repeat steps 1-2 until the vector is empty.

The resulting polylines are used as a basis for all subsequent steps. Channels for the wires are generated by offsetting the polylines by $\frac{e+\Delta}{2}$ and subtracting the resulting polygon from the current layer. e is the extrusion width of the conductive material extruder, Δ is an offset value provided by the user.

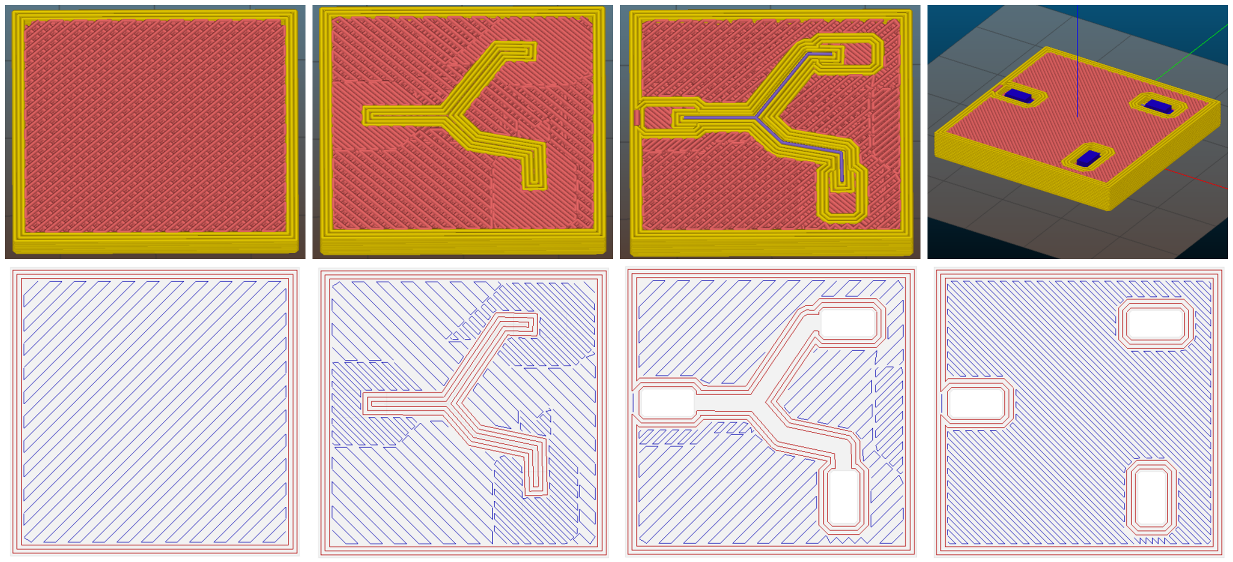


Figure 6 – Different layers of a toolpath with integrated electronic components and wires, generated by the augmented slicing tool. Layers in ascending order from left to right: **1**: Internal layer with sparse infill (red lines, density 50%). **2**: Layer directly below wires and components. The internal perimeters (yellow) serve as a smooth "substrate" for the wires. The infill density is increased at regions where components will be placed in the next layers, to provide a solid surface, covering the sparse infill. **3**: Channels and cavities are generated for the extrusion of wires and components. **4**: Top layer with placed components.

The polylines can be directly used as trajectory for the direct write extrusion. However, since the starting point of a an extrusion path tends to be not reliable after a retraction move, each trajectory is split into two segments and the extrusion is executed from the center of the line to the SMD-pad. The segments are extended by a user defined offset `extrusion_overlap` to ensure a proper electrical connection. Possible over- or underextrusion only occurs at non-critical spots in the object. Priming the extruder is most important after a toolchange. Hence the segments are ordered by length and the amount of overlapping is increased only for the first segment of each layer to minimize double extrusions.
As a final step, "substrate" is generated for the next layer by offsetting the polylines by a small value ε and subtracting the resulting polygon from the contour of L_{-1}. This causes the extrusion of a perimeter, where the "gap" is precisely aligned with the wire to avoid disruptions of the wire extrusion due to a crossing, rough surface as described in [8].

Performance considerations

Since the user is working in a representation of the final toolpath, every interaction e.g. moving a part, routing a wire requires re-slicing and generation of the entire toolpath. For complex objects, this can easily take seconds up to minutes, resulting in poor user experience and high latencies. To address this issue, the layer representation is extended by a boolean flag, indicating whether this layer is affected by a certain user operation. The toolpath generation steps are then only executed for those layers marked "dirty".

5. Evaluation / experimental results

All experiments were conducted by a modified Kühling&Kühling RepRap Industrial FDM-printer [23]. The general design of the modifications is an evolution step from the CNC-mill based prototyping machine which was introduced in our previous work [5]. The printer is equipped with an additional syringe extruder for conductive ink, a pivot-mounted vacuum gripper and two cameras

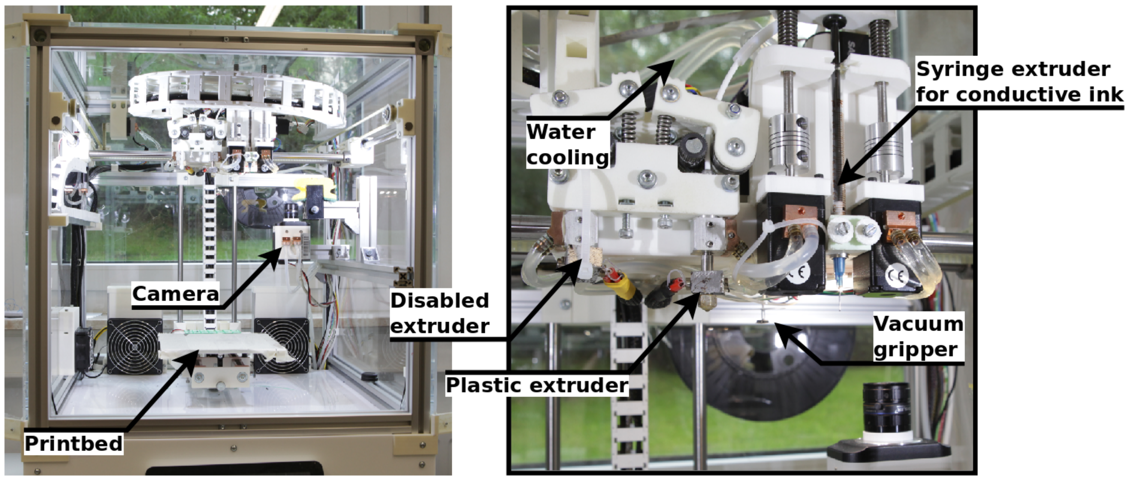


Figure 7 – Modified RepRap Industrial printer, augmented by a conductive ink extruder, vacuum gripper for component placing and cameras to align components.

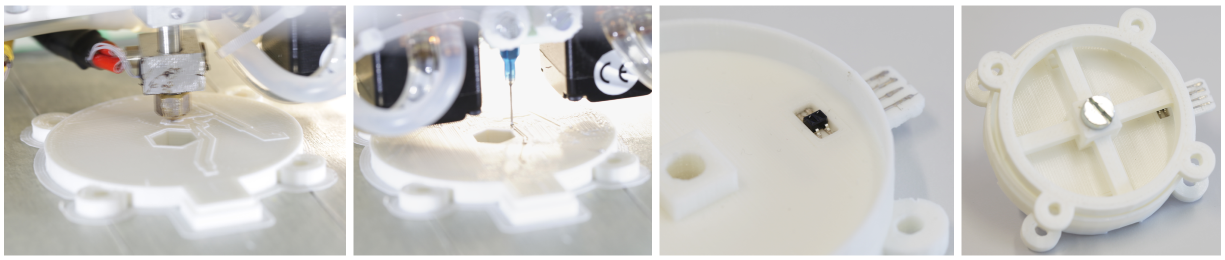


Figure 8 – Fabrication of the 2-DOF force-torque sensor prototype shown in Fig. 4. The device uses infrared proximity sensors to measure the deflection of deformable beams with known stiffness and computes the force vectors on an embedded 8-bit microcontroller.

for robotic placement of SMD-components. The modifications were facilitated by the fact that both hardware design and software of the printer are open source and all parts are specifically designed to be printable on the machine itself.

To test the algorithms with a sufficiently complex design, we fabricated the prototype of a 2-DOF force-torque sensor (see Fig. 4 and 8). The sensor consists of a deformable beam structure with known mechanical properties. The thickness of the beam structure is computed according to the desired measuring range. The beam-deflection induced by an applied force is measured by infrared proximity sensors which are recessed into the surface and the deflection distance is then converted into the applied force by the embedded microcontroller.

The quality and reliability of the printed wires is generally highly dependent on the printers calibration. Even very small z-offsets of the ink extruder cause interrupted extrusions and broken wires due to the polymer's high surface tension. Fig. 9 shows an effect that occurs if the plastic extruder "jumps" crossing the channel. A small amount of plastic is drawn as a thin string from the substrate extruder's hot-end, later causing interruptions at the ink extrusion. Those strings are virtually invisible for the eye. This effect can be mitigated by using plastic with a higher glass transition temperature and lifting the z-axis by several millimeters during the jump.

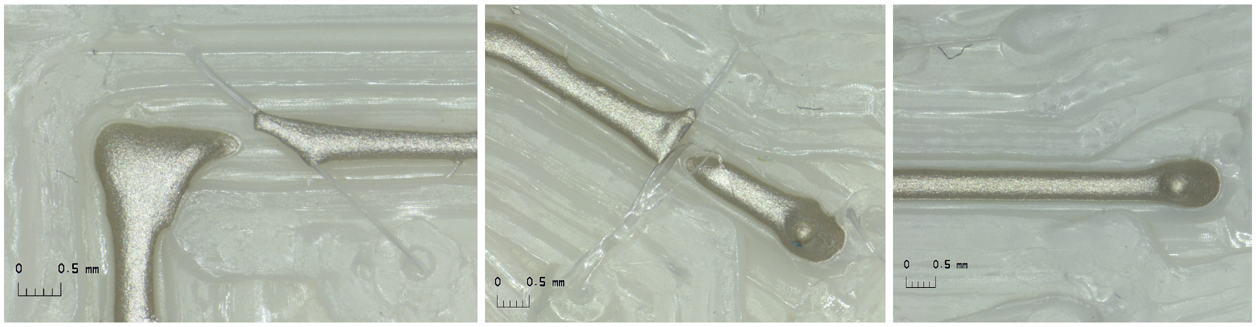


Figure 9 – Interrupted conductive ink extrusion due to thin plastic strings caused by insufficient retraction of the PLA base material (left) and successful extrusion on ABS material (right).

6. Conclusion and future work

This paper presents the concept and first implementation of a software for the integration of electric circuits into 3D-printed objects. So far, we have only tested simple circuits with a few components, but the increase in usability over existing approaches is clear, providing a new tool for researchers and developers. The design is modular and relies on open source projects and widely used common standards and formats.

The pending next development steps will focus on improvements of the user interface and routing algorithms. Our current work is dedicated to the implementation of low level routing approaches for better utilization of the available space inside the objects and alignment to existing structures to improve the quality. Important concepts include "Grid alignment" as a simple limitation of part and waypoint positions to a predefined grid as commonly used in PCB layout tools. The grid resolution is determined by the diameter and spacing of extrusions and the common pitch of the SMD components. An additional approach is to follow the contour of existing perimeters and prior defined wire routes by finding an entry and exit point and offsetting the shortest polygon segment between the two points.

More information, including the source code is available on the projects website: `https://tams.informatik.uni-hamburg.de/research/3d-printing/conductive_printing`.

References

[1] Erick De Nava, Misael Navarrete, Amit Lopes, Mohammed Alawneh, Marlene Contreras, Dan Muse, Silvia Castillo, Eric Macdonald, and Ryan Wicker. Three-Dimensional Off-Axis Component Placement and Routing for Electronics Integration using Solid Freeform Fabrication. *Proceedings of the 19th International Solid Freeform Fabrication Symposium*, pages 362–369, 2008.

[2] Amit Joe Lopes, Eric MacDonald, and Ryan B. Wicker. Integrating stereolithography and direct print technologies for 3D structural electronics fabrication. *Rapid Prototyping Journal*, 18:129–143, 2012. ISSN 1355-2546. doi: 10.1108/13552541211212113.

[3] Daniel Periard, Evan Malone, and Hod Lipson. Printing embedded circuits. *Proceedings of the International Solid Freeform Fabrication Symposium*, pages 503–512, 2007.

[4] Cassie Gutierrez, Rudy Salas, Gustavo Hernandez, Dan Muse, Richard Olivas, Eric MacDonald, Michale D Irwin, Ryan Wicker, K Churck, M Newton, and Brian Zufelt. CubeSat Fabrication through Additive Manufacturing and Micro-Dispensing. *Proceedings from the IMAPS Symposium*, 2011.

[5] Florens Wasserfall. Embedding of SMD populated circuits into FDM printed objects. In *Proceedings of the 26th International Solid Freeform Fabrication Symposium*, pages 180–189, Austin, 2015.

[6] Jacob J. Adams, Eric B. Duoss, Thomas F. Malkowski, Michael J. Motala, Bok Yeop Ahn, Ralph G. Nuzzo, Jennifer T. Bernhard, and Jennifer A. Lewis. Conformal printing of electrically small antennas on three-dimensional surfaces. *Advanced Materials*, 23:1335–1340, 2011. ISSN 09359648. doi: 10.1002/adma.201003734.

[7] Jacob Bayless, Mo Chen, and Bing Dai. Wire embedding 3D printer, 2010.

[8] David Espalin, Eric Muse, Danny W. andMacDonald, and Ryan B. Wicker. 3D printing multifunctionality: Structures with electronics. *International Journal of Advanced Manufacturing Technology*, 72:963–978, 2014.

[9] C.Y Kim, a. Cuaron, M.a. Perez, D. Espalin, E. MacDonald, and R.B. Wicker. Cooperative Fabrication Methodology for Embedding Wireon Curved Surfaces. *Proceedings of the International Solid Freeform Fabrication Symposium*, pages 185–196, 2014.

[10] Matt Saari, M. Galla, Bryan Cox, Edmond Richer, Paul S. Krueger, and Adam L. Cohen. Active Device Fabrication Using Fiber Encapsulation Additive Manufacturing. *Proceedings of the International Solid Freeform Fabrication Symposium*, pages 26–39, 2015.

[11] S. Brett Walker and Jennifer A. Lewis. Reactive silver inks for patterning high-conductivity features at mild temperatures. *Journal of the American Chemical Society*, 134(I):1419–1421, 2012. ISSN 00027863. doi: 10.1021/ja209267c.

[12] 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. *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 363–372, 2013. doi: 10.1145/2493432.2493486.

[13] J. Ledesma-Fernandez, C Tuck, and R Hague. High Viscosity Jetting of Conductive and Dielectric Pastes for Printed Electronics. *Proceedings of the International Solid Freeform Fabrication Symposium*, pages 40–55, 2015.

[14] S. J. Leigh, R. J. Bradley, C. P. Purssell, D. R. Billson, and D. A. Hutchins. A simple, low-cost conductive composite material for 3d printing of electronic sensors. *PLoS ONE*, 7(11), 11 2012. doi: 10.1371/journal.pone.0049365.

[15] K. Blake Perez and Christopher B. Williams. Combining Additive Manufacturing and Direct Write for Integrated Electronics – A Review. *Proceedings of the International Solid Freeform Fabrication Symposium*, pages 962–979, 2013.

[16] Eric Macdonald, Rudy Salas, David Espalin, Mireya Perez, Efrain Aguilera, Dan Muse, and Ryan B. Wicker. 3D printing for the rapid prototyping of structural electronics. *IEEE Access*, pages 234–242, march 2014. doi: 10.1109/ACCESS.2014.2311810.

[17] CadSoft EAGLE website. URL `https://cadsoft.io/`.

[18] Autodesk Project Wire. URL `https://spark.autodesk.com/tags/project-wire/`.

[19] Slic3r project website. URL http://slic3r.org/.

[20] Daniel Ahlers. Development of a software for the design of electronic circuits in 3d-printable objects. Bachelor thesis, University of Hamburg, 2015. URL https://tams-www.informatik.uni-hamburg.de/publications/2015/BSc_Daniel_Ahlers.pdf.

[21] Bala R. Vatti. A generic solution to polygon clipping. *Communications of the ACM*, 35(7): 56–63, 1992. ISSN 0001-0782. doi: 10.1145/129902.129906.

[22] Xiaorui Chen and Sara McMains. Polygon offsetting by computing winding numbers. *Proceedings of the ASME international design engineering technical conferences and computers and information in engineering conference*, 2:565–575, 2005.

[23] Kühling & Kühling RepRap Industrial Printer. URL https://github.com/kuehlingkuehling/RepRap-Industrial.