2to3D: A User-Friendly Parametric CAD Program for Laser Cutting

Leo McElroy and Ben Brown
Fall 2017 CS701 with Professor Grant
Middlebury College

Abstract

We created a user-friendly Computer-Aided Design (CAD) program for laser cutting which integrates the desired features of vector drawing software and 3D parametric CAD programs. We implemented basic drawing, edit, and transformation tools. The program is capable of outputting SVGs - the Scalable Vector Graphics image format. Geometric constraints are solved using Cassowary.js [1] - an implementation of the Simplex Algorithm. Linear constraints (coincidence, fixing of points, vertical, and horizontal) are efficient and robust. Nonlinear constraints (distance, angles between lines, parallel, and perpendicular) are not naturally accommodate by the Simplex Algorithm and required clever workarounds to function, consequently they are less robust. We were successful in creating an accessible and functional parametric design program which simplifies the laser cutting workflow, however the project requires improvement before we foresee adaptation by laser fabrication hobbyists. Future development of the project would include building out drawing features, adding circle and arc shape primitives, supporting raster image import, utilizing a geometric constraint solver that accommodates nonlinear equations (such as gradient descent), and improving state handling of the program to support undo/redo tools.

Motivation

- Existing CAD programs are not designed for laser cutting.

- Laser cutting requires 2D schematics so non-parametric drawing programs are appealing to use but laser cutters produce 3D objects so parametric design capabilities are highly desirable. Parametric design is based on constraints and dimensions, which facilitates creating real objects. This is the way an engineer would design.

- Commercial software is expensive.

- Using existing CAD software to laser cut is like using a fighter jet to drive a nail. It's over-engineered, arguably not the right tool for the job, and has a high probability of crashing and burning.

- An ideal CAD program for laser cutting would incorporate features of both traditional drawing programs and parametric design programs.

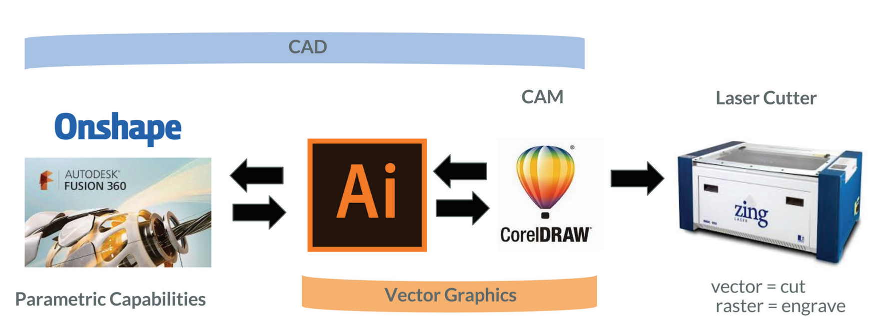


Figure 1: The current digital fabrication workflow for laser cutting involves switching back and forth between drawing programs and parametric modeling programs. But switching is only motivated by a small number of features of each tool.

Goal

Our goal was to create an open-source single-page web-based program specifically designed for laser cutting that combines the best parts of 2D drawing programs and 3D CAD programs, while leaving the unnecessary parts behind. Specifically, our program would output SVGs so drawings could be easily sent to a laser cutter, and would incorporate a geometric constraint solver so we could model parametrically.

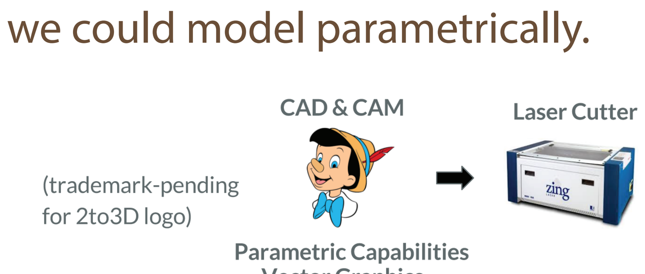


Figure 2: The workflow we want to create for laser cutting.

Terminology

Computer-Aided Design (CAD): A program used to create digital designs. The programs we refer to can be broken up into programs used for creating visual designs (Adobe Illustrator and CorelDRAW) and programs for designing 3D objects intended for fabrication (Fusion360, Solidworks, Onshape) these programs are generally parametric.
Computer-Aided Manufacturing (CAM): A program which inputs a design and outputs machine paths for a digital fabricator.
Direct Modeling: A design paradigm where the user interacts directly with the geometry. Typically involves dragging, rotating, or scaling.
Parametric Modeling: A design paradigm that utilizes a geometric constraint solver that allows the user to specify constraints and dimensions on geometry. The design is than automatically adjusted to satisfy these constraints when the user modifies geometry directly.
Vector Graphics: A format where images are represented mathematically. Vector graphics are scale-agnostic which is important for generating cutting paths in a digital fabricator.
Raster: A format where images are represented by pixels. Rasters are typically interpreted as engravings when laser cutting.

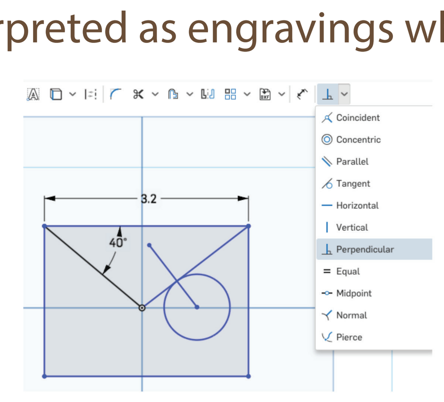


Figure 3: Screenshot of parametric CAD program (Onshape) which shows common geometric constraints.

Methods

Our project was implemented in ES6 JavaScript using node.js and React. The shape primitives are lines, beziers, and freehand curves. Drawings are composed of shape primitives and other shapes made from primitives and constraints. For example, the rectangle tool creates four lines with coincident constraints on their endpoints, the top and bottom lines have horizontal constraints, and the left and right lines have vertical constraints. Implementing a geometric constraint solver in addition to creating a drawing program over the course of one semester was infeasible so we used Cassowary.js to solve the systems of equations created by geometric constraints. Cassowary.js implements the Simplex Algorithm which solves systems of linear equations. Some geometric constraints were not naturally represented with linear equations. We reformed and approximated such constraints using one or more linear equations or inequalities.

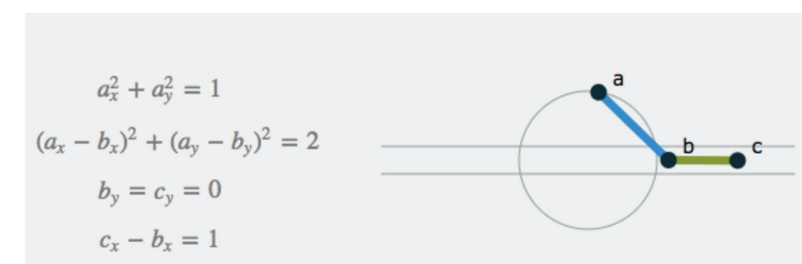


Figure 4: Parametric constraints can be represented as equations and inequalities on the values they constrain. In this figure adapted from Ref. [2] we constrain point *a* to always lie on the circle of radius 1, we fix the line with endpoints *a* and *b* to have length √2 using the distance equation, and we set the line with endpoints *b* and *c* to lie on the *x*-axis and have length 1.

Results

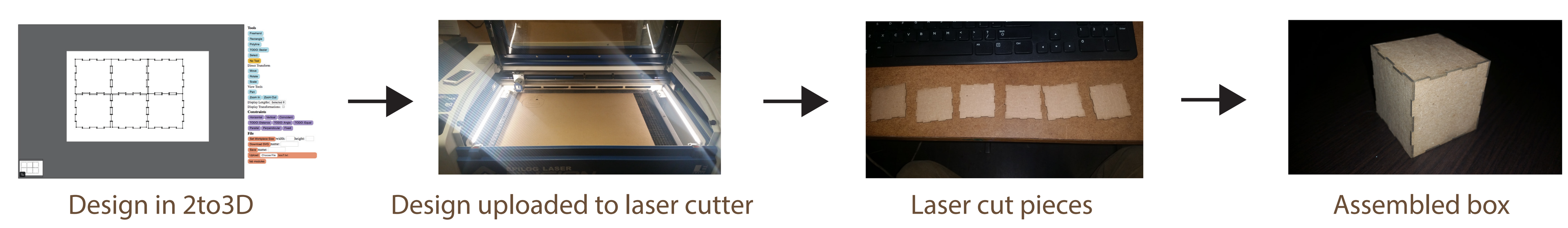
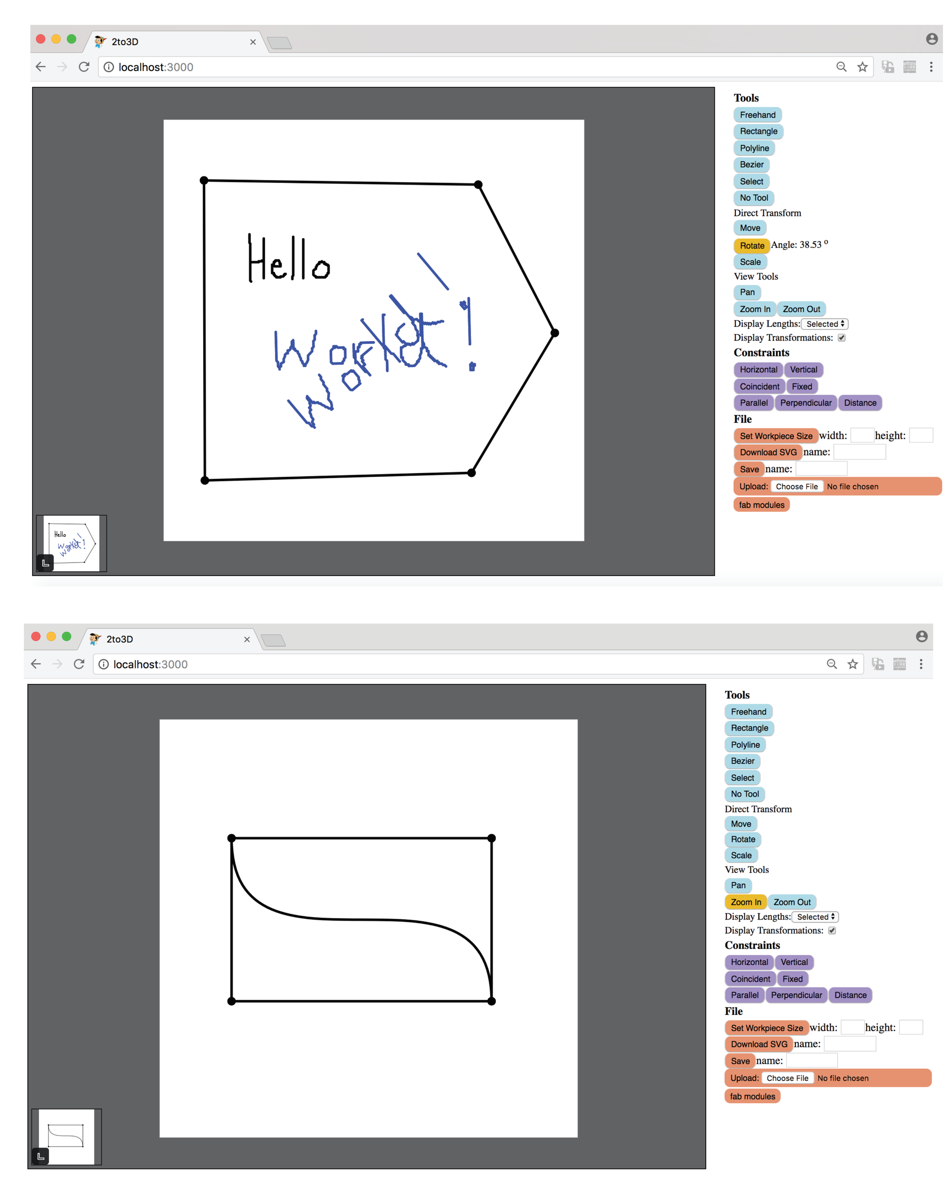


Figure 5: Full digital fabrication workflow using 2to3D. The laser cutter we had access to used CorelDRAW for CAM so the drawing was created in 2to3D and imported to CorelDRAW just for printing.



Capabilities

Drawing
- Freehand
- Rectangle
- Polyline
- Bezier

Direct Transformations
- Move
- Rotate
- Scale

Constraints
- Horizontal (line)
- Vertical (line)
- Coincident (points)
- Fixed (points)
- Parallel (lines)
- Perpendicular (lines)
- Length (line)

Navigation
- Pan
- Zoom in / out

Other
- Click and drag points
- Download SVG
- Save drawing
- Upload drawing
- Copy / Paste
- Delete
- Hotkeys
- Workarea size adjustable

Display Options
- Lengths
- Transformation values

Figure 6: Screenshots of program. Top shot depicts rotation transformation. Bottom shot depicts beziers, rectangles, and the coincident constraint.

Conclusions and Future Work

We accomplished our main goal of creating a parametric web-based vector drawing program, but as with all software engineering, the work is never done. Future work would include the following goals:

- Build out shape primitives to include circles and arcs, as well as associated constraints (such as tangent).

- Improve the program's state-handling to enhance user experience. This could involve introducing atomic operations which would allow tracking of a file's timeline.

- Create a JavaScript library that can efficiently solve systems of nonlinear equations using gradient descent.

- Integrate 2to3D with existing open-source web-based CAM software such as MIT's Center for Bits and Atoms' "fab modules."

References

[1] Badros, G. J., Borning, A., and Stuckey, P. J. 2001. The Cassowary linear arithmetic constraint solving algorithm. ACM Transactions on Computer-Human Interaction 8, 4 (Dec.), 267–306.

[2] Keeter, Matt. "Constraint Solver." Matt Keeter Blog. Web. https://www.mattkeeter.com/projects/constraints/

Acknowledgments

We would like to thank Professor Grant for his guidance throughout the semester and the folks at Maker Works for first teaching Leo to laser cut (among so many other teachings).