`Trace2SCAD`: Converting Images into Parametric `OpenSCAD` Models

Henry (Hank) Dietz

COIMG-162: 11:40 – Noon, March 5, 2026

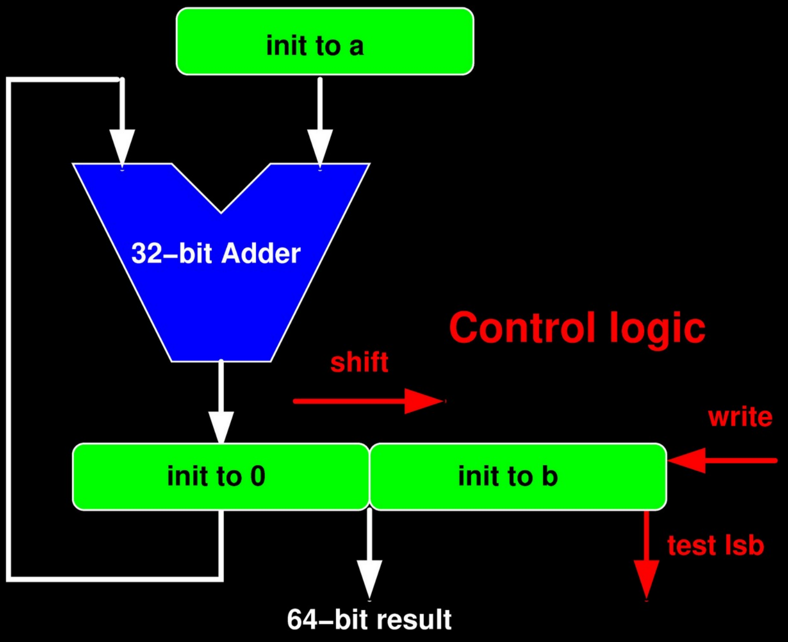
University of Kentucky
Electrical & Computer Engineering

Abstract

Ironically, it is often useful to manipulate 2D graphics for 3D printing. **`Trace2SCAD`** was originally written to convert 2D diagrams into printed structures that could be understood by a blind student feeling the surface. However, by its initial open source release in January 2015, it had already been applied to convert pixel-mapped images into parametric 3D models and 3D-printable image renderings (e.g., lithophanes). The original version was a shell script using a variety of helpers, especially **`potrace`**. The new version is pure C++ code using the **`OpenCV`** library and incorporating many new features, including the ability to render color images as 3D models to be printed for viewing by reflected light (similar to the filament paintings produced by **`HueForge`**). It is also significant that the output generated is not mere 3D shapes, but parametric 3D models coded in the **`OpenSCAD`** programming language. This paper describes the functionality and algorithms used in the new **`Trace2SCAD`**.

Development of `Trace2SCAD`

- Built in 2015 to help 3D print tactile versions of the 2D diagrams used in the Computer Organization course for a blind student
- Originally a shell script using ImageMagick, **`potrace`**, & **`awk`**
- In 2025, reimplemented in **C++** using OpenCV and new algorithms



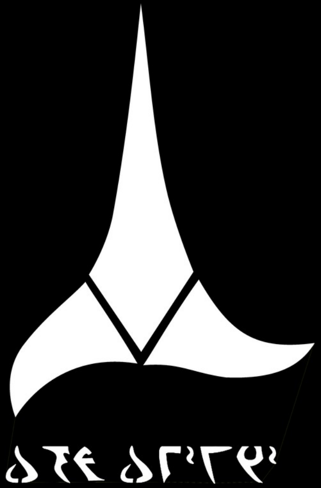
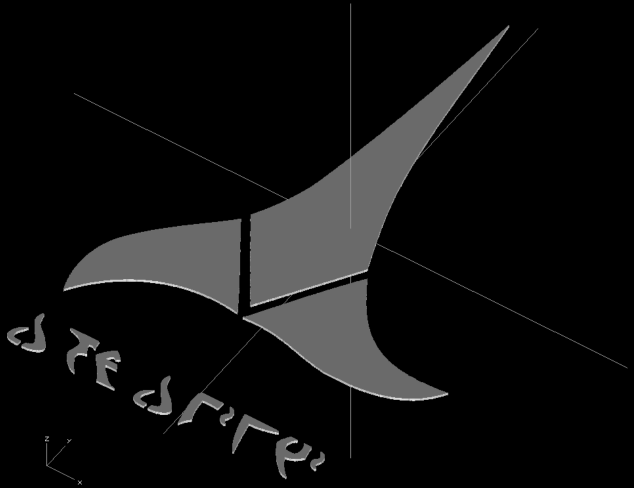
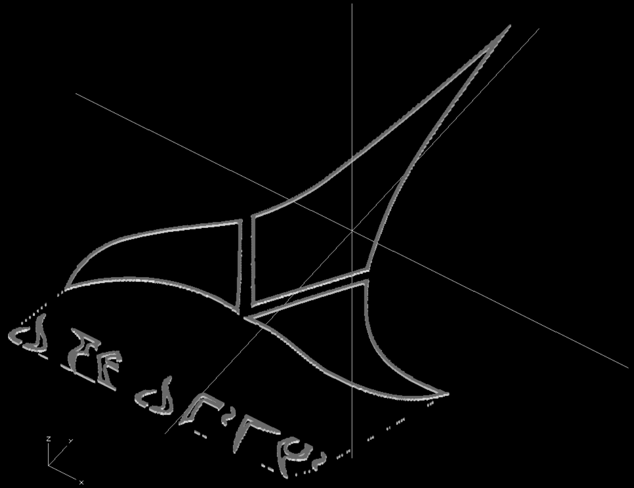
Why `Trace2SCAD`?

- Trace refers to the idea of finding edges in an image
- SCAD refers to `OpenSCAD`
  - A portable, open source, 3D modeling system
  - 3D models are expressed as programs using `union`, `difference`, `intersection`, `hull`, `minkowski` and other operations on basic 3D objects
  - Programs can be fully parametric and the built-in `customizer` interface allows interactive parameter adjustment
  - 3D models are *rendered* into `.stl` files for 3D printing, etc.

Pixel-Map to Vector Conversion

- Most digital imaging operates on pixel maps
  - Each image has a fixed pixel count (spatial resolution)
  - Each pixel samples a spatial region’s appearance
  - Vectorized images have scalable resolution of boundaries
- This is a well-studied problem with many algorithms
  - The “standard” is polygon-based `potrace`
  - Output is optimized Bezier curve segments, not vectors
  - Union of curves for different colors can have gaps and overlaps

Original `Trace2SCAD` Processing

Original	Vectorized `OpenSCAD`	Vectorized Edges
		

Trace2SCAD Handling of Issues Using `potrace`

- Nested polygons need to be ordered for differencing
- No control over number of line segments output
  - Simplify line segments by merging lines with same angles
  - Reduce complexity by iterative rescaling of image
- Polygons can be non-printable
  - Single-pixel structures can produce 0-width segments
  - Each threshold is handled separately, so layers can have overlapping polygons

Aggregate.Org UNBRIDLED COMPUTING
electronic IMAGING 2026 MARCH 1–5
University of Kentucky

Trace2SCAD 2025 C++ Reimplementation

- Don’t need `potrace` and **ImageMagick** helpers… does require **OpenCV** library
- New edge walking algorithm gives pixels unit width rather than treating a pixel as a point
- Can use printer nozzle size and print dimensions
- Still does ordering of nested polygons and line merging

Lithophanes

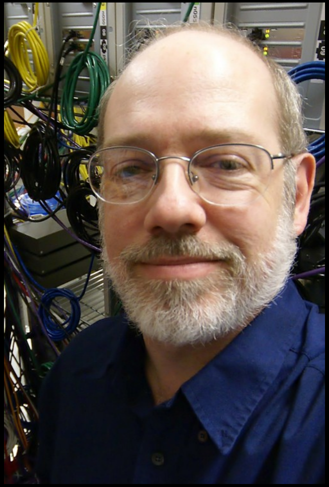
- A lithophane is an image that becomes visible when backlit
  - In 1800s, made as artwork using cast porcelain
  - Classically monochrome using white material, but can be color
- Digital lithophane creation
  - Essentially treats image pixel luminance as a bump map: darker pixels become taller bumps (thicker material)
  - Transitions between bumps of differing heights?
  - 3D printed using 100% infill
  - Convenient to have one side (bottom as printed) flat

Lithophane Software

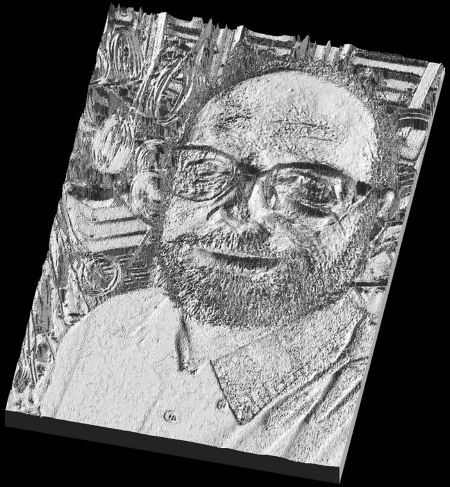
- Dedicated software including
  - PNG23D `http://kyllikki.github.io/png23d/`
  - Img2scad `https://gitlab.com/victor-engmark/img2scad`
  - Image to lithophane `https://3dp.rocks/lithophane/`
  - Kittl – a free AI image vectorizer `https://www.kittl.com/`
- Embedded in software including
  - OpenSCAD `https://openscad.org/`
    `surface(file="image.png", center=true, invert=true);`
  - Cura `https://ultimaker.com/software/ultimaker-cura/`

Monochrome Lithophane

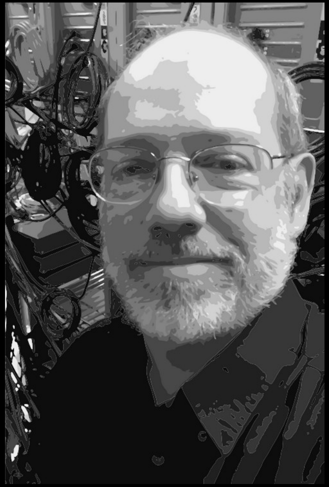
High-res. image



OpenSCAD surface



Kittl 16 level vector

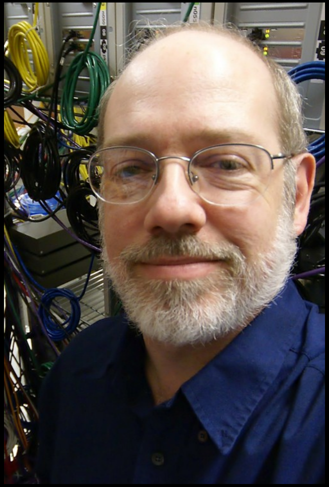


Trace2SCAD Monochrome Lithophanes

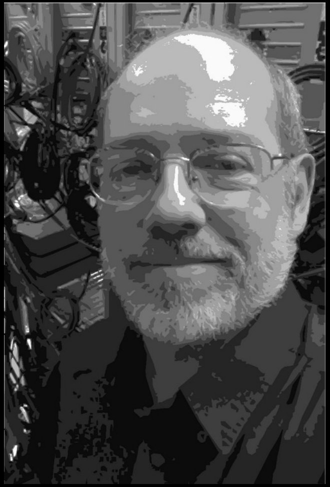
- Original scheme stacked simplified polygons
- Now handled as a subset of voxel-based color stacking analysis
- Honestly, the OpenSCAD `surface()` mechanism does well enough without any help – if you don’t mind a huge stl file

`Trace2SCAD` Monochrome Lithophane

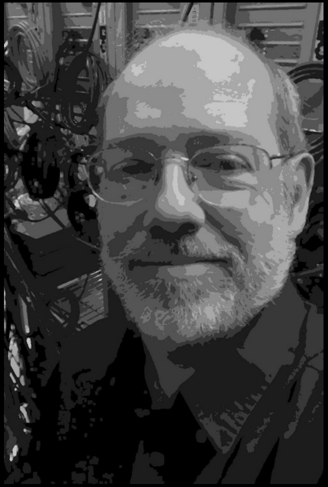
High-res. image



Original Lithophane



New Lithophane



Filament Painting, as in `HueForge`

- Every filament has a color and a Transmission Distance (TD)
  - TD is the extruded thickness that light penetrates
  - Color is the RGB of white light *reflected* or *transmitted*
  - Layers of extrusion blend to make different colors, although blending is different lighting front vs. rear (i.e., lithophane)
- Extrusion-based 3D printers slice to layers
  - It is relatively easy to swap filaments at specific heights, so a bump map with color change heights is efficient to print
  - Alternatively, filaments can occupy specific 3D volumes

Aggregate.Org UNBRIDLED COMPUTING
electronic IMAGING 2026 MARCH 1–5
University of Kentucky

Filament Painting in Trace2SCAD

- This was not in the original Trace2SCAD…
- There are two types of filament stacking supported:
  - True stacking (i.e., like `HueForge`) with **one color per layer**
  - Arbitrary **volumetric stacking with a maximum of s filaments** used in the stack – ideal for 3D printers that have filament changers with a capacity of *s* filaments
- Trace2SCAD attempts to fully automate both

Filament Stacking Analysis in Trace2SCAD

- 32-bit color values with CIE DeltaE 2000 metric
- Uses a database of filaments
  - Blended color of a stack is computed much like in **`HueForge`**
  - Either K-means or a Genetic Algorithm optimizes color palette selection using JND (Just Noticable Difference) filtering to reject stacks that are too similar to a less costly stack
  - Error propagation optionally used to improve area color
- **Volumetric stacking uses a Genetic Algorithm to try options**

Aggregate.Org UNBRIDLED COMPUTING
electronic IMAGING 2026 MARCH 1–5
University of Kentucky

Filament Stack Output in Trace2SCAD

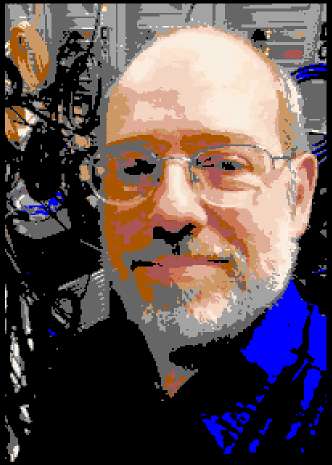
- Output is an OpenSCAD program that *generates* the voxel set for each filament color
- Note: default front-lit assumption is a black background

```
// Color palette is:
// filament 0: 000000 0.1 TD (PLA) Black
// filament 1: ffffff 5 TD (PLA) White
// filament 2: 0000ff 2 TD (PLA) Blue
// filament 3: ff7f00 3 TD (PLA) Orange
// color 0: 000000 by stacking 3 0
// color 1: 0000ff by stacking 2
// color 2: aa5500 by stacking 3
// color 3: 666666 by stacking 1
// color 4: aa5555 by stacking 3 3 3 3 2 3
// color 5: cc7755 by stacking 3 3 3 2 1 3
// color 6: cc9966 by stacking 3 1
// color 7: a3a3a3 by stacking 3 3 3 0 1 1
// color 8: e0ad99 by stacking 3 3 2 1 3 1
// color 9: f4d5c2 by stacking 2 1 1 3 1 1
// color 10: f9e0ce by stacking 2 3 3 1 1 1
// color 11: fbe5cf by stacking 3 1 3 1 1 1
// color 12: fbecde by stacking 3 3 1 1 1 1
// color 13: f8eee2 by stacking 1 3 1 1 1 1
```

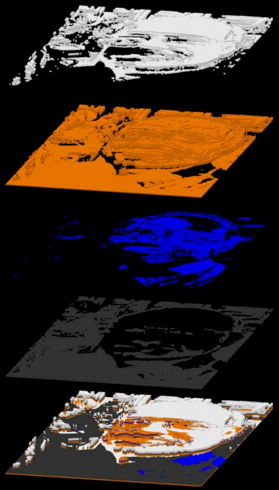
`Trace2SCAD` Filament Painting

Low-res. image

Filament Painting



Filament Voxels



Aggregate.Org UNBRIDLED COMPUTING

electronic IMAGING 2026 MARCH 1–5

University of Kentucky

Conclusions

- Trace2SCAD became fairly popular in 2015
  - **Produces OpenSCAD output that is small & editable**
  - **LOTS of complaints about difficulty installing it**
- Things have changed (credit to HueForge for filament painting)
  - Monochrome lithophanes are easy
  - Multi-color 3D printers are cheaply available
- The new, self-contained, Trace2SCAD uses novel algorithms
  - **Improved resolution-aware vectorization**
  - **GA-optimized color stacking for filament painting**

The Trace2SCAD Basic Command Line

```
Usage: ./t2s {options} inputimagefile
-b #    set base height in layers (default 0)
-c #    set target max complexity (default 0)
-d file read filament database from file
-E #    set error propagation shading (default 0)
-e #    set max line-combining error in pixels (default 0)
-F #    set frame thickness in units (default 0)
-f #    set highpass filter radius (default -1)
-g #    set gamma (default 1)
-h #    set extrusion layer height
-i      try to improve local contrast (tonemapping)
-j #    set just noticeable difference (default 0 parts per 256)
-l levs set level values by:
        a single integer number of layers
        one or more ,-separated floats containing .
        one or more ,-separated hexadecimal color values
        default is 000000, ffffff
-n      negate the image (default no)
-m file mask to only non-black (0x000000) pixels in this image
-o file set OpenSCAD output filename to file (default from input filename)
-p pre  set prefix for OpenSCAD module names (default from input filename)
-t #    set turd size (default 1)
-s #    set maximum number of filaments to use enabling pure color stacking
-u #    set resolution unit (nozzle diameter in mm, default 1)
-v      enable verbose output, keep temp files
-X #    set X resolution int times input columns
-x #    set X resolution (columns)
-Y #    set Y resolution int times input rows
-y #    set Y resolution (rows)
```