Gradient Descent in the ALPS: Abstracted Low-Poly Stylization and Fabrication

RUBEN WIERSMA*, ALEXANDRE BINNINGER*, PEIZHUO LI, TANGUY MAGNE, ANNIKA OEHRI, AVIV SEGALL, DANIELLE LUTERBACHER, MARCEL PADILLA, JING REN and OLGA SORKINE-HORNUNG,
ETH Zurich, Switzerland

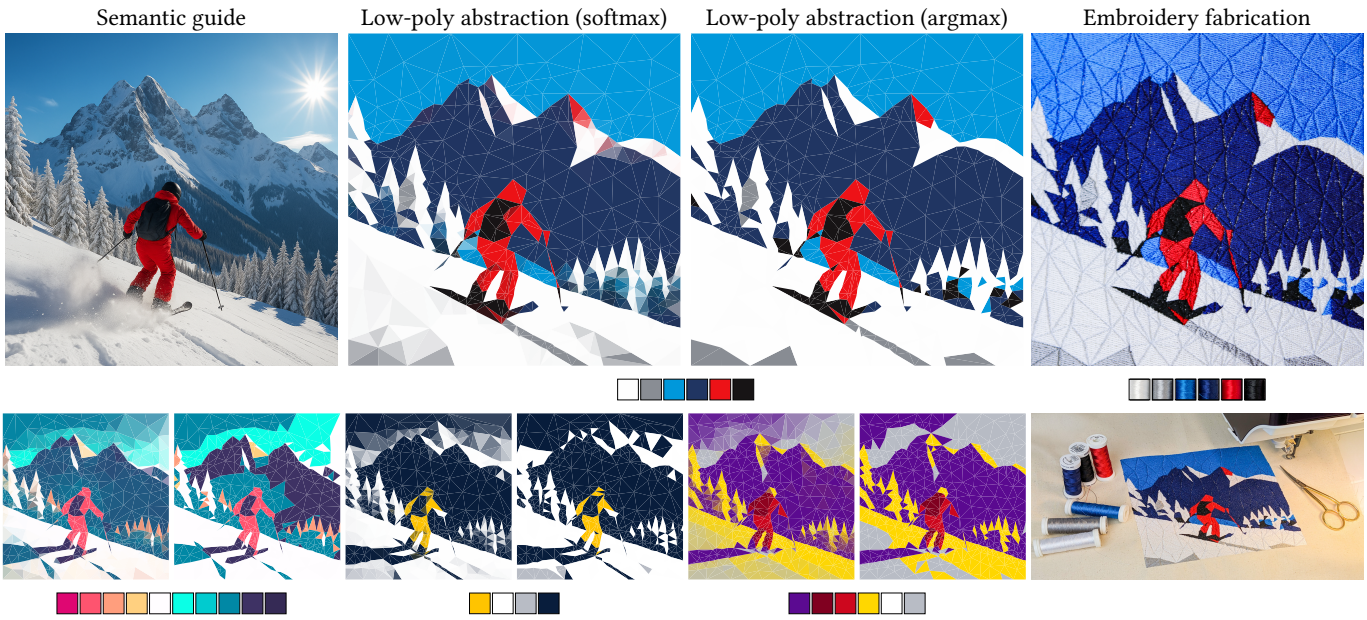


Fig. 1. ALPS takes an input image, triangle count and color palette and produces an abstracted, palette-constrained, low-poly vector output. The resulting triangulation is guaranteed to be without self-intersections. We can output a "softmax" version (every color is a combination of the palette) or an "argmax" version (colors are quantized to the exact colors in the palette). The output can easily be edited and used for fabrications, such as embroidery. Second row: results for several palettes and the fabrication. **Note:** Results are included as vector graphics. Some PDF viewers may display unintended "hairlines".

The low-poly style is a popular genre of vector graphics that depicts objects and scenes as flat-shaded meshes of low polygon count, often with a limited palette. In this paper, we propose a method to generate 2D low-poly meshes to abstract images. While it has been possible to achieve this look with general-purpose image generators and vector-based diffusion models, the resulting images are not guaranteed to be valid polygonal meshes. A key problem is that polygons overlap or intersect and, in the case of pixel-based image generators, the shapes are often not polygons. Moreover, the colors are not guaranteed to be constrained to a fixed palette. Aside from aesthetic considerations, this has practical consequences: it complicates editing and fabricating the results. We solve this problem by representing an image as a 2D polygonal mesh and optimizing the topology, geometry and coloring of the mesh using score distillation sampling, while enforcing geometric constraints, such as manifoldness and bijectivity. This presents unique challenges due to the discrete nature of the topology, which we handle using a fine-to-coarse strategy based on mesh simplification. By also constraining the colors to a fixed palette, we are able to produce various fabrications such as mosaics, embroidery, crocheting, patchwork and stencils from the resulting vector images. Code is at https://github.com/rubenwiersma/alps.

CCS Concepts: • **Computing methodologies** → **Image representations**; **Mesh geometry models**; *Texturing*; Non-photorealistic rendering.

Additional Key Words and Phrases: Image generation, Geometry, Meshing

ACM Reference Format:
Ruben Wiersma, Alexandre Binninger, Peizhuo Li, Tanguy Magne, Annika Oehri,, Aviv Segall, Danielle Luterbacher, Marcel Padilla, Jing Ren and Olga Sorkine-Hornung. 2026. Gradient Descent in the ALPS: Abstracted Low-Poly Stylization and Fabrication. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '26), July 19–23, 2026, Los Angeles, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3799902.3811050

*Both authors contributed equally to this research.

Authors' Contact Information: Ruben Wiersma, rubenwiersma@gmail.com; Alexandre Binninger, alexandre.binninger@inf.ethz.ch; Peizhuo Li, peizhuo.li@inf.ethz.ch; Tanguy Magne, tanguy.magne@inf.ethz.ch; Annika Oehri, oehria@inf.ethz.ch; Aviv Segall, avivsegall@gmail.com; Danielle Luterbacher, danielle.luterbacher@inf.ethz.ch; Marcel Padilla, padilla.marcel@gmail.com; Jing Ren, jing.ren@inf.ethz.ch; Olga Sorkine-Hornung, sorkine@inf.ethz.ch,
ETH Zurich, Zurich, Switzerland.

SIGGRAPH Conference Papers '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2554-8/2026/07
https://doi.org/10.1145/3799902.3811050

1 Introduction

Low-poly art depicts images with a small set of polygons, typically triangles, each filled with a constant color. The style originated from polygonal modeling constraints and grew into a distinct aesthetic, valued for clear geometry, crisp boundaries and scalability as vector graphics [Gai and Wang 2016]. Besides illustration and branding, low-poly art maps well to fabrication methods that require discrete parts and limited palettes, for example, stained glass, screen printing, mosaics or embroidery. In addition, as a vector graphic, low-poly art is easy to edit [Sun et al. 2007] and can be reproduced in any resolution. Our goal is to help users create abstracted vector art in the low-poly style, which can directly be used for fabrication.

To capture the visual style of low-poly images, we use a data- and semantics-driven approach based on pretrained diffusion models, rather than providing an analytical definition and corresponding algorithm as done in prior work, e.g., Lawonn and Günther [2019]. Diffusion models produce results that align remarkably well with a given style prompt, often better than hand-crafted algorithms. These results show that pixel-based diffusion models can generate images in the low-poly domain. However, creating low-poly art requires more than capturing visual style and likeness. For example, for easy editability and fabrication, the output must be a planar triangle mesh without overlaps or gaps between the faces. For fabrication- and artistic purposes, it is often required that the colors are drawn from a specified palette: for example, to fabricate an embroidery from a limited set of colored threads (Fig. 1). Existing pixel-based diffusion models can match appearance, but offer no guarantees on triangulation or editability (Fig. 3). Diffusion models conditioned on a color scheme [Shum et al. 2025] likewise lack strict palette enforcement. Existing vector-graphic generators can produce a low-poly look [Xing et al. 2024], yet often produce overlapping or poorly aligned triangles that require tedious cleanup. On the other hand, classical approaches that aim to approximate the input image with a triangle mesh fail to capture semantic information at lower resolutions and do not handle palettes well (Fig. 2). Moreover, they often contain triangle intersections and triangles with poor quality (e.g., sliver triangles). We aim to deliver a practical alternative that is semantically guided, user-controllable and provably valid and, therefore, directly usable for fabrication.

We build our solution, *Abstracted Low-Poly Stylization* (Alps), on top of Score Distillation Sampling (SDS) [Poole et al. 2023]. SDS leverages the rich generative prior of a pre-trained diffusion model to optimize the parameters of a differentiable representation, such as vector graphics, by backpropagating gradients through a differentiable renderer. Alps optimizes a 2D triangle mesh that adheres to a given palette (Fig. 4) and can be semantically guided with a text prompt and a depth map of an input image. The main technical challenge is to provide as much freedom to the diffusion process as possible, while guaranteeing the required properties: palette constraints, mesh validity and user control. To this end, we make three key contributions: First, we include a line-search strategy in the SDS optimization to guarantee that each mesh update produces non-intersecting triangles and include regularizers to support precise

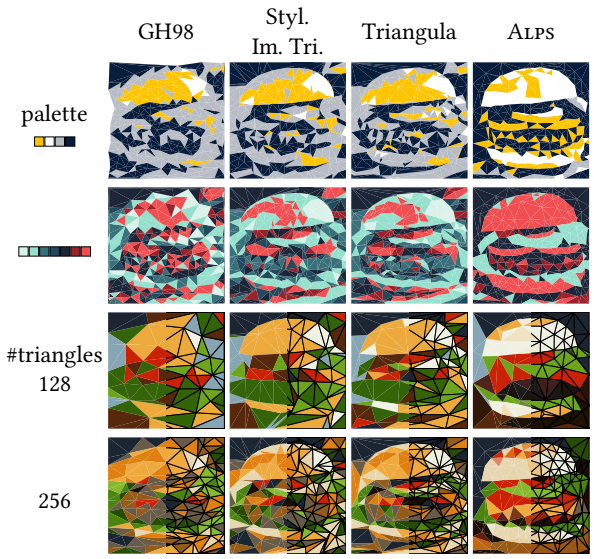


Fig. 2. Comparison of Alps against low-poly stylization methods [Garland and Heckbert 1998; Hayashi 2025; Lawonn and Günther 2019]. Top two rows: results constrained to specific color palettes. Bottom two rows: limiting the triangle budget (128 and 256 triangles).

control. Second, we optimize the mesh topology using a fine-to-coarse strategy within the diffusion process, so that the triangulation is guided by SDS. Third, we propose a palette-based color assignment of mesh faces that supports changing mesh topology and quantization.

We validate Alps in comparisons against classic image triangulation approaches and recent neural methods for generating pixel- and vector-based low-poly images. We also demonstrate specialized variants of our approach for improved user control, e.g., to achieve exact symmetry and topological guarantees. Finally, we demonstrate a wide range of fabrication applications for the resulting low-poly abstractions that highlight the relevance of our method and its properties: mosaics, embroidery, tapestry crocheting, patchworking and stencil printing. Implementation- and fabrication details, background information about score distillation sampling and additional results and evaluations can be found in the Supplemental.

2 Related work

Low-poly image stylization requires finding the right vertex placement, connectivity and face colors to approximate a given image. Early approaches address vertex placement through methods such as generation of seed points from K-means segments and Ramer-Douglas-Peucker simplification to define polygon boundaries [Uasmith et al. 2017]. Zhang et al. [2015] employ real-time Sobel edge detection to prioritize vertices on image edges, thereby preserving key structural information. Yang et al. [2003] adaptively place vertices with higher density in high-frequency regions using the Floyd-Steinberg error diffusion algorithm, while Joseph et al. [2024] use a novel boundary-aware seed placement strategy, sampling both region boundaries and interiors. These techniques prioritize basic

Fig. 3. A comparison of generative methods, illustrating the shortcomings of existing vector- and pixel-based generators: none of the generative methods produce valid triangulations, they contain curved edges and overlapping faces and most cannot produce palette-constrained outputs. More results and analysis can be found in the Supplemental.

image approximation or compression, leading to common artifacts like jagged edges [Uasmith et al. 2017] or sampled points deviating from true object boundaries [Ma et al. 2017; Zhang et al. 2015]. Other research significantly advances visual fidelity to the input image and user control: artistic rendering guided by feature edges and Voronoi diagrams [Gai and Wang 2016], energy minimization for adaptive vertex optimization and mesh adaptivity [Lawonn and Günther 2019], iterative mesh generation through point addition and deletion [Adams 2013; Mostafavian and Adams 2015] and interactive systems with adaptive thinning [Ma et al. 2017]. In contrast to these methods, our approach focuses on semantic similarity through the SDS loss and is aimed more at *abstraction* of the input, rather than direct approximation of the input. The aforementioned approaches also lack color controllability, often relying on color averaging or post-processing for aesthetic effects [Gai and Wang 2016; Joseph et al. 2024; Ma et al. 2017], thus limiting their utility for fabrication workflows with material constraints.

Image vectorization converts raster images to vector graphics [Lai et al. 2009; Li et al. 2020]. Early methods, such as gradient meshes [Sun et al. 2007], including their topology-preserving extensions [Lai et al. 2009], represent images primarily with smoothly varying colors through bicubic interpolation and optimization. Approaches employing layered linear gradients [Du et al. 2023; Favreau et al. 2017] decompose images into editable semi-transparent layers

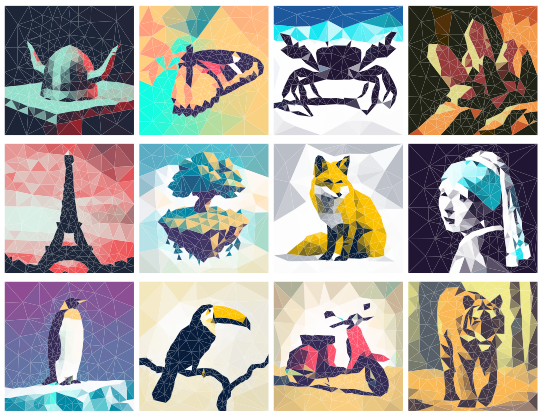


Fig. 4. Results from Alps. All results were generated with 512 triangles and softmax color assignment for different palettes.

to capture smooth color transitions and achieve abstract styles. More recently, differentiable rasterization techniques [Laine et al. 2020; Li et al. 2020] enable gradient-based optimization of vector parameters based on image-space losses [Du et al. 2023; Favreau et al. 2017; Lai et al. 2009; Sun et al. 2007]. We use differentiable rasterization as a core building block of Alps and apply it for low-poly abstraction with a fixed color palette and guaranteed geometric validity.

Several works optimize triangle meshes for different targets. FlexiCubes [Shen et al. 2023], DMesh [Son et al. 2024] and TetWeave [Binninger et al. 2025] are differentiable mesh representations for recovering 3D shapes. We share the optimization of the mesh with gradient descent and the use of regularizers to improve mesh quality, but we optimize a mesh in 2D and focus on the triangulation and per-face coloring, rather than the *3D geometry* of the triangulation. TriWild [Hu et al. 2019] finds a planar triangulation with curved elements for a given 2D Beziér curve, to be used in simulation. While the input and target application differ, we share common challenges, such as finding regular triangulations and preventing intersections.

Generative AI for stylization and vector graphics. Some art styles and fabrication methods require generated outputs to adhere to given constraints, such as the triangle-mesh requirement in low-poly art styles. A fundamental approach to adapt image-based diffusion models is score distillation sampling (SDS), introduced in DreamFusion [Poole et al. 2023]. DreamFusion generates 3D objects through a differentiable 3D representation and shows examples of constrained representations to enforce requirements such as symmetry. A downside of SDS is that it can yield oversaturated and oversmoothed results and have a lack of diversity [McAllister et al. 2024]. Building on DreamFusion, VectorFusion [Jain et al. 2023] generates vector graphics from text, optimizing path parameters with SDS. SVGDreamer and SVGDreamer++ refine SVG generation using particle-based score distillation and adaptive primitive control for improved editability and visual quality [Xing et al. 2025, 2024]. These approaches do not enforce discrete polygonal facets or strict color palette adherence, central to our low-poly generation. We also address the oversaturation problems of SDS by using a color palette. Our color palette approach is related to SD-πXL [Binninger and Sorkine-Hornung 2024], which uses a Gumbel-softmax reparameterization and score distillation on pixel-art images. We target triangle meshes as output, rather than regular pixel grids, which requires a different representation for color assignments.

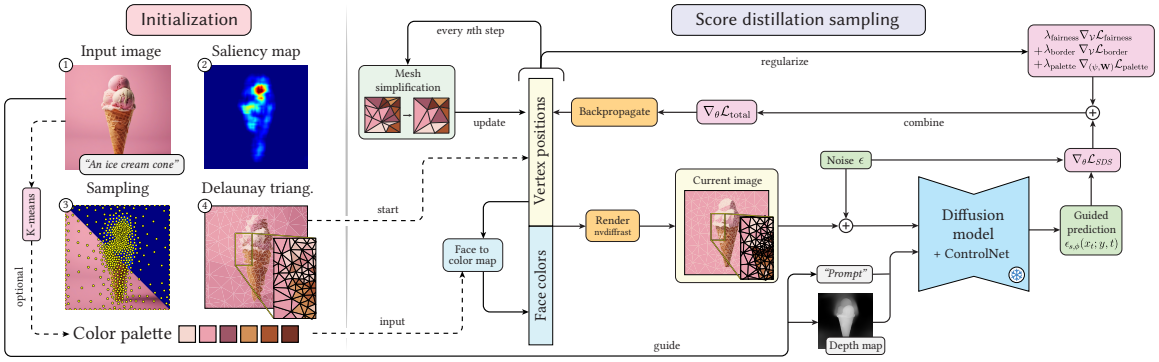


Fig. 5. Pipeline of our method. **Initialization (left):** We sample points according to saliency and compute a Delaunay triangulation. **Score distillation sampling (right):** We update vertex positions and face colors using gradients from a depth-conditioned diffusion model. Mesh simplification reduces complexity while preserving appearance every n steps. The output triangulation is intersection-free and palette-constrained, ready for fabrication.

3 Method

We present Alps, a method for generating low-poly, 2D triangular meshes with solid face colors that capture the semantic and structural content of an input image or prompt. To ensure that the resulting meshes are easily editable and fabricable, we guarantee the following properties:

Property 3.1 (Control of triangle count and quality). *A certain style or fabrication method may require specific triangle counts. A design may require fair, uniform triangles even in flat areas, or call for larger triangles in some areas to ease fabrication. A user should be able to control these properties precisely.*

Property 3.2 (Color quantization). *Many fabrication methods and artistic styles use a discrete set of materials or a discrete color palette. The approach should be able to take in any palette, which could differ from the colors of the input image.*

Property 3.3 (Manifold tiling). *To produce a tileable "mosaic", neighboring triangles must share a single edge. There can be no intersections, as they are impossible to produce physically and hurt the appearance in a digital format. This constraint also aids downstream editability: it is much easier to move a vertex and update face colors in a mesh than to edit the vertices and colors of possibly overlapping and intersecting triangles. This is a hard constraint.*

To enable Property 3.2, we allow our method to diverge from the colors in the input image. An additional benefit is that the method can more freely use the polygons to capture the semantic content. This is an important difference from prior work, which approximates the input image, including the colors. Instead, our approach generates an *abstraction* of the input image. None of these properties is currently enforceable with existing pixel-based or vector-based generative methods for low-poly art. Guaranteeing these properties, while providing as much freedom during the optimization as possible, is the main technical challenge of our work.

Overview. An overview of our approach is provided in Fig. 5. The input to our approach, used for initialization and as semantic guidance, is a pixel-based image $\mathcal{I}$ and/or a text prompt y. The user also sets a desired number of faces F and a color palette $\mathcal{P}$ of P RGB colors. Our approach outputs a planar mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$, where $\mathcal{V} \in \mathbb{R}^{V\times 2}$ and $\mathcal{F}$ is a matrix of $F \times 3$ vertex indices. Each face is assigned an RGB color $\mathcal{C} \in \mathbb{R}^{F\times 3}$, where the colors are convex combinations of $\mathcal{P}$ or restricted to the elements of $\mathcal{P}$. We want the mesh $\mathcal{M}$ to be manifold and without overlapping or intersecting triangles. Our approach optimizes the vertex positions and face colors using score distillation sampling [Poole et al. 2023]. We add regularizers to the loss function and adjust the optimization procedure to improve mesh quality, include user constraints and guarantee non-intersecting and non-overlapping triangles. To jointly improve the topology and appearance, we use projected gradient descent: the optimization is interleaved with mesh simplification, projecting the mesh to the closest appearing simpler mesh.

3.1 Initialization

We initialize the mesh to be close to the input image, if one is given, and distribute triangle density where detail is needed (Fig. 5, initialization). Our initialization is based on the importance-sampling technique proposed in [Lawonn and Günther 2019], which computes a saliency map [Hou et al. 2012] as a density and samples this density with inverse CDF [Pharr et al. 2023]. We also experimented with a density based on attention, as used in ControlSketch [Arar et al. 2025]. However, we found that saliency tends to work better when the output needs to be closer to the input image. For example, attention might be concentrated on the eyes, while we also require high detail on the contour of the face; saliency covers both. Rather than using inverse CDF for sampling, we adopt a weighted Voronoi stippling approach [Secord 2002], where samples are iteratively moved to their Voronoi-cell centers. This results in a more even distribution for a high-quality initial triangulation. Finally, evenly spaced vertices are added along the image border. Following Lawonn and Günther [2019], we compute a Delaunay triangulation on the initial vertex positions.

3.2 Color assignment using palettes

Low-poly art often limits itself to a strict color palette. This is not only visually appealing but essential for fabrication purposes, where

each piece must be produced in a single color and physically assembled. Our palette alignment approach is based on SD-πXL [Binninger and Sorkine-Hornung 2024], which was developed for quantized pixel-art image generation. SD-πXL assigns a vector λ of P logits to each pixel. The logits are mapped to coefficients through a softmax to express the final color as a convex combination of the palette $\mathcal{P} \in \mathbb{R}^{P\times 3}$. Because our mesh topology changes during optimization, we cannot assign a logit vector per face. Instead, we query a continuous 'color field' at the barycenter b_f of each face f. Similar to related work (e.g., [Hasselgren et al. 2022]), we encode the color field as a neural network $\lambda(x)$. In our approach, the network takes a 2D position x as input and returns P logits. The coefficient $s_k(x)$ for the palette entry with index k is obtained via the softmax operation

$$s_k(x) = \frac{e^{\lambda_k(x)}}{\sum_{j=1}^{P} e^{\lambda_j(x)}}. \tag{1}$$

The per-face color $c(b_f)$ is then given by the convex sum

$$c(b_f) = \sum_{k=1}^{P} s_k(b_f)\, p_k. \tag{2}$$

If a discrete palette color is required, rather than the continuous mixture given by the softmax, we simply select the single palette color with the highest logit (argmax).

The color field $\lambda(x)$ is implemented with a neural network as

$$\lambda(x) = \mathbf{W} g_\psi(\mathbf{p}_x), \tag{3}$$

where $\mathbf{p}_x$ is the sinusoidal positional encoding of x [Vaswani et al. 2023], g_ψ is an MLP with parameters ψ and q output features, and $\mathbf{W}$ is a $P \times q$ matrix of palette color embeddings. Both ψ and $\mathbf{W}$ are optimized. To encourage diversity of color assignments, we compute a Gram matrix $\mathbf{G} = \mathbf{W}\mathbf{W}^\top$ and penalize its negative log-determinant. This encourages linear independence, resulting in larger variation between logits. We stabilize the palette embeddings $\mathbf{W}$ by enforcing unit-norm constraints as a loss, so their pairwise dot products encode only angular relationships. The latent vector $g_\psi(x)$ is not normalized: its magnitude allows the model to reflect the 'confidence' for a given palette color. Both losses are summed to the palette regularization loss as

$$\mathcal{L}_{\text{palette}} = -\log\det\left(\mathbf{W}\mathbf{W}^\top\right) + \frac{1}{P}\sum_{k=1}^{P}(\|\mathbf{w}_k\| - 1)^2 \tag{4}$$

The color field is initialized for a given palette by fitting it to the target image, minimizing mean squared error with Adam [Kingma and Ba 2015].

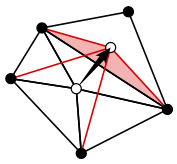
3.3 Optimization

We optimize the vertex locations $\mathcal{V}$ and color field parameters $\{\psi, \mathbf{W}\}$ using gradient descent along the SDS gradient. Additional soft constraints for user control are included by means of the loss function, and gradients are computed using automatic differentiation, resulting in the following total gradient:

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\text{total}} = \nabla_\theta \mathcal{L}_{SDS} + \lambda_{\text{fairness}} \nabla_{\mathcal{V}} \mathcal{L}_{\text{fairness}} \\ + \lambda_{\text{border}} \nabla_{\mathcal{V}} \mathcal{L}_{\text{border}} + \lambda_{\text{palette}} \nabla_{(\psi,\mathbf{W})} \mathcal{L}_{\text{palette}},\end{aligned} \tag{5}$$

where $\nabla_\theta \mathcal{L}_{SDS} = \mathbb{E}_{t,\epsilon}\left[\left(\epsilon_{s,\phi}(x_t; y, t) - \epsilon\right)\frac{\partial x}{\partial \theta}\right]$ is computed by running a denoising step with a diffusion model ϵ (see the Supplement for details). The term $\mathcal{L}_{\text{fairness}}$ measures the deviation of the triangle corner angles from those of an equilateral triangle, $\pi/3$, promoting a fair triangulation: $\mathcal{L}_{\text{fairness}} = \sum_{f\in\mathcal{F}}\sum_{v\in f}\left(\theta_v - \frac{\pi}{3}\right)^2$. The term $\mathcal{L}_{\text{border}}$ constrains border vertices to slide along the border by selecting all vertices with either an x or y coordinate close to -1 or 1 and penalizing the squared error from -1 or 1 in the corresponding coordinate. This constraint can be extended to non-border vertices, for example, vertices lying along a given line, to achieve visual effects like symmetry. The term $\mathcal{L}_{\text{palette}}$ is described in Sec. 3.2. To enforce structural similarity, the diffusion model used to compute the SDS loss is conditioned with a depth map of the input image through ControlNet alongside the original conditioning of the diffusion model with a text prompt. If only a text prompt is given, we input an all-zero depth map.

Preventing intersections. To guarantee that no step of the optimization introduces intersections, we perform a backtracking line search on the vertex positions. The detection of intersections during the line search can be simplified to finding *triangle inversions*, based on the following observation.



Observation 3.1 (Intersecting triangles). *Given that the current mesh $\mathcal{M}$, embedded in 2D, does not contain any intersections and the boundary is fixed, any newly introduced triangle intersection requires at least one triangle to be inverted (red triangle in inset).*

Because the initial triangulation is Delaunay, it does not contain intersections. Therefore, if the new positions $\mathcal{V}^{\text{new}}$ do not cause inverted faces, the step is accepted. Otherwise, we find a step size that does not introduce intersections: we first flag vertices with an inverted adjacent triangle and fix their position to the previous vertex positions $\mathcal{V}^{\text{prev}}$. Next, we find a step size $\alpha \in [0, 1]$ for all other vertices, such that

$$\mathcal{V}^{\alpha} = (1-\alpha)\,\mathcal{V}^{\text{prev}} + \alpha\,\mathcal{V}^{\text{new}} \tag{6}$$

yields no inverted triangles. Fixing the positions V^α for unflagged vertices, we find the largest step size $\beta \in [0, 1]$ such that

$$\mathcal{V}^{\beta}_{\text{flagged}} = (1-\beta)\,\mathcal{V}^{\text{prev}}_{\text{flagged}} + \beta\,\mathcal{V}^{\text{new}}_{\text{flagged}} \tag{7}$$

does not yield inverted triangles. Since $\beta = 0$ is always feasible, the search is guaranteed to terminate. This adaptive step size ensures that no triangles intersect, while allowing the optimizer to take as large a step as possible. This technique can be used to guarantee a minimum triangle size by increasing the threshold on triangle area used to detect inversion.

One could also reduce intersections by adding a Laplacian term to the objective [Lawonn and Günther 2019] or optimization [Nicolet et al. 2021]. We experimented with this approach, but did not include it in the final method. It encourages vertices to shift toward the average of their neighbors, but does not prevent intersections (Fig. 6) and also encourages triangle fairness even when not required. Our approach is guaranteed to prevent intersections and allows for separate control of triangle fairness with the $\mathcal{L}_{\text{fairness}}$ term.

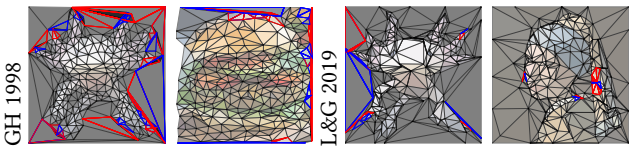


Fig. 6. The outputs with the most intersections, highlighted in red and blue, for Garland and Heckbert [1998] and Lawonn and Günther [2019]

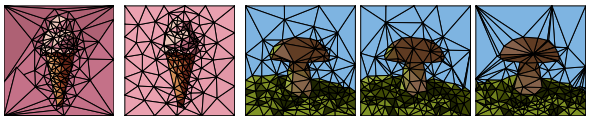


Fig. 7. Mesh simplification variants. First two columns: QEM vs. Line Quadrics. Last three columns: setting density gamma (contrast) to p.

3.4 Adaptive mesh topology

Mesh connectivity is discrete and cannot be optimized with regular gradient descent. Therefore, we use projected gradient descent in a fine-to-coarse fashion: every few iterations, the mesh is projected to the closest coarser mesh. In the in-between iterations, the SDS process can 'catch up' on any changes made to the topology by adjusting vertex positions and face colors. A key insight is that the extended quadric error metric (QEM) [Garland and Heckbert 1998] measures the least-squares difference in appearance between a mesh and its coarsening. By greedily contracting edges based on QEM, we effectively project our mesh to the closest coarser mesh. However, because we quantize colors, many vertices are contained in a 1-ring of faces that share the exact same color (e.g., the background in Fig. 7). Such vertices are 'flat' from the QEM perspective. Flat vertices are known to give poor results with QEM, because the error quadric is ill-defined there. To solve this problem and control the quality of the triangulation, we adapt the recently proposed line quadrics [Liu et al. 2025], which can improve numerics and encourage uniformly distributed vertices (Fig. 7, second column). An additional benefit of line quadrics is that we can use the line-quadric weight in combination with quadric scaling to preserve detail in prescribed areas, fulfilling property 3.1 (Fig. 7, last three columns). In our experiments, we set the quadric scaling in proportion to the density map used during initialization. Finally, we augment the algorithm to not collapse any edges that would introduce a triangle inversion (Obs. 3.1) to prevent intersections.

4 Results

We present comparisons and ablations and conclude by demonstrating optional guarantees supported by our framework. Applications related to fabrication are presented in Sec. 5. Alps runs in roughly 5 min on an RTX 4090, with peak VRAM usage of 8 GB. Implementation details and ablations are provided in the Supplemental and code is available at https://github.com/rubenwiersma/alps.

4.1 Comparisons

No existing method fulfills all the properties of our method (Table 1). Classical approaches do not support a text prompt, semantic guidance or palette control. Generative-AI methods often contain non-polygonal shapes (see Fig. 3 and Supplemental), provide no control of triangle quality, contain many triangle intersections and do not support color quantization. Here, we present a visual comparison with methods designed to produce low-poly outputs, and we provide quantitative metrics and a perceptual study to substantiate our claims. More comparisons are included in the Supplemental.

4.1.1 Visual comparisons. We compare to mesh simplification [Garland and Heckbert 1998] (implemented as *qslim* in libigl [Jacobson et al. 2018]), Stylized Image Triangulation [Lawonn and Günther 2019] and Triangula [Hayashi 2025], an online tool for image triangulation. Triangula uses a modified genetic algorithm to optimize a point cloud so that its Delaunay triangulation closely reproduces the input image. We omit additional methods included in Lawonn and Günther [2019] based on their observation that "Among [...] related work [mesh simplification (GH98)] almost always gave [the] best results". We also leave out approaches for triangulating images for compression [Wang et al. 2024; Xiao et al. 2022], as their focus is on representing the input image as accurately as possible, rather than producing a stylized, triangulated version of the image. We constrain our optimization to a palette based on K-Means clustering in RGB space with $k = 16$ and quantize the other results to the same cluster centroids. We also include results for user-specified palettes. For the competing methods, we apply the palette mapping to the input image before triangulation. Specifically, image pixels are soft-mapped to the palette using distance-based softmax weighting in the perceptually uniform Oklab color space [Ottosson 2020], which ensures smooth transitions and lightness preservation (see supplemental for details). For each method, we aim to use the same number of triangles ($F = 512$) by manually setting the only available control parameter: error tolerance for Lawonn and Günther and number of vertices $V = 300$ for Triangula.

The results are succinctly shown in Fig. 8 and Fig. 9 and an extensive comparison with 30 examples is included in the Supplemental. Alps produces vector art with higher clarity, better triangulations and a more stylized aesthetic. For the user-specified palettes (Fig. 8), Alps makes better use of the palette, resulting in clearer images that are easier to 'read'. Alps tends to use more contrast in its color

Table 1. Comparison of low-poly stylization methods. Alps is the only approach that accepts either an image or a text prompt, incorporates semantic guidance and lets the user control both triangle count and quality while guaranteeing intersection-free meshes. Its built-in color quantization further enhances suitability for downstream fabrication and editing tasks.

		Image	Prompt	Semantic	Triangle Count	Triangle Quality	Inters. Free	Color Quant.
image	Edge Collapse (GH 1998)	✓	✗	✗	✓	✗	✗	✗
	Stylized Image Triang.	✓	✗	✗	✗	✓	✗	✗
	Triangula	✓	✗	✗	✗	✗	✓	✗
gen-AI	ChatGPT, Gemini	✓	✓	✓	✗	✗	✗	✗
	Adobe Firefly 4	✓	✓	✓	✗	✗	✗	✓
	SVGDreamer++ [Xing et al. 2025]	✗	✓	✓	✓	✗	✗	✗
	Alps **(Ours)**	✓	✓	✓	✓	✓	✓	✓

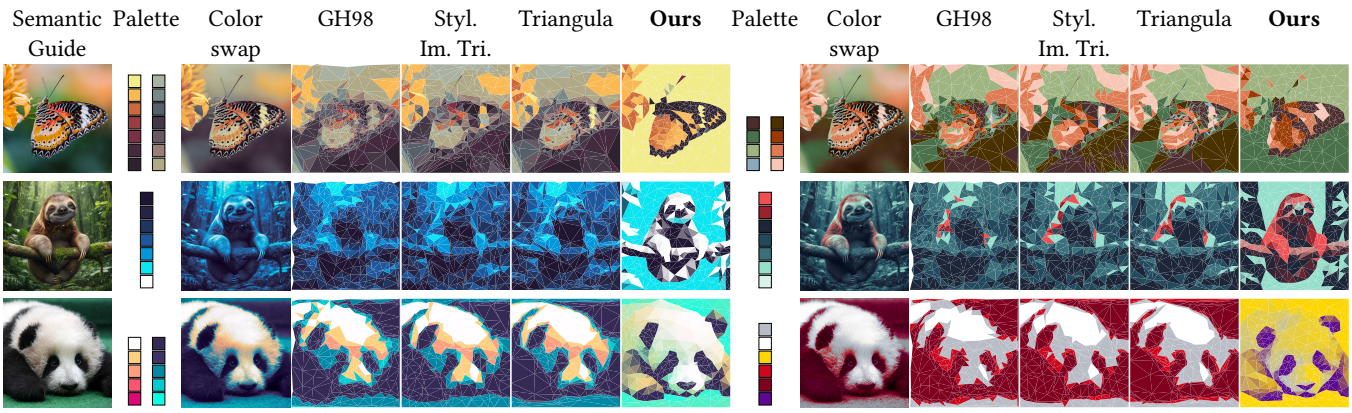


Fig. 8. To evaluate ALPS against other approaches under a constrained color-palette setting, we first apply a soft color swapping to the input image. Next, we triangulate the image and snap each triangle color to the nearest color in the palette. In contrast, our method directly optimizes the color assignment on the palette, without relying on any color swapping of the input. Easier to see when zoomed in on a digital screen.

choices, both because of the palette regularizer and since it can more freely exploit the palette colors given the semantic guidance.

4.1.2 Quantitative evaluation. We evaluate our method against baselines using semantic fidelity (CLIPScore [Radford et al. 2021]) and stylistic quality [Schuhmann 2023] metrics. The results in Table 2 show that our approach achieves the highest score across both metrics. The superior CLIPScore indicates that our method preserves the semantic content of the input image more robustly than competing approaches. The higher Aesthetic Predictor score confirms that our semantic-driven pipeline better captures the low-poly style. This demonstrates that ALPS successfully balances abstraction with content retention while maintaining high aesthetic quality.

We also conduct a perceptual study with 75 participants, comparing our method to baselines on the semantic and aesthetic criteria. The survey results (Table 2) reveal that respondents find ALPS to be the best method for both criteria in roughly 60% of cases, significantly more than any other method: the runner-up, Triangula, wins in roughly 25% of cases. More details of the perceptual study are provided in the Supplemental.

Table 2. Quantitative results. We use CLIPScore [Radford et al. 2021] for semantic similarity and the predictor of Schuhmann [2023] for aesthetics. The user column ($n = 75$) shows the percentage a method was rated best in our user study. We also provide a fairness measure and the percentage of triangles whose smallest angle is less than 10°. An extended table and further details are provided in the Supplemental.

	Semantic		Aesthetic		Triangle quality	
	CLIPScore ↑	Users (%) ↑	Predictor ↑	Users (%) ↑	Fairness ↑	SA<10° (%) ↓
GH98	0.6231	0.0	4.751	0.7	0.5829	22.08
Styl. Im. Tri.	0.6807	12.1	4.996	16.1	0.6335	14.41
Triangula	0.6869	25.3	5.043	26.5	0.6588	8.36
ALPS	**0.7075**	**62.6**	**5.137**	**56.8**	**0.8208**	**1.39**

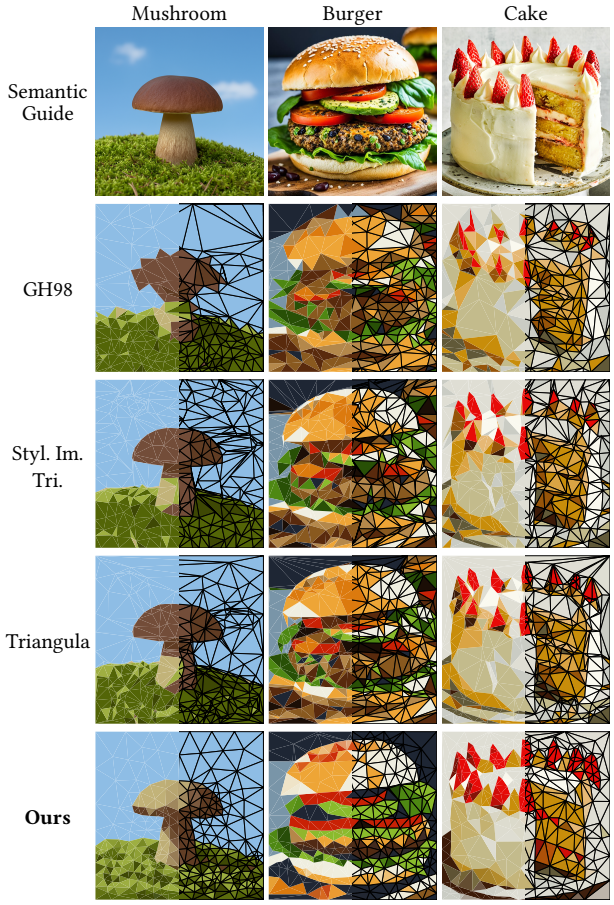


Fig. 9. Comparison with previous methods. Rows: Semantic Guidance, GH98 [Garland and Heckbert 1998], Stylized Image Triangulation [Lawonn and Günther 2019], Triangula [Hayashi 2025], and ALPS. Easier to see when zoomed in on a digital screen.

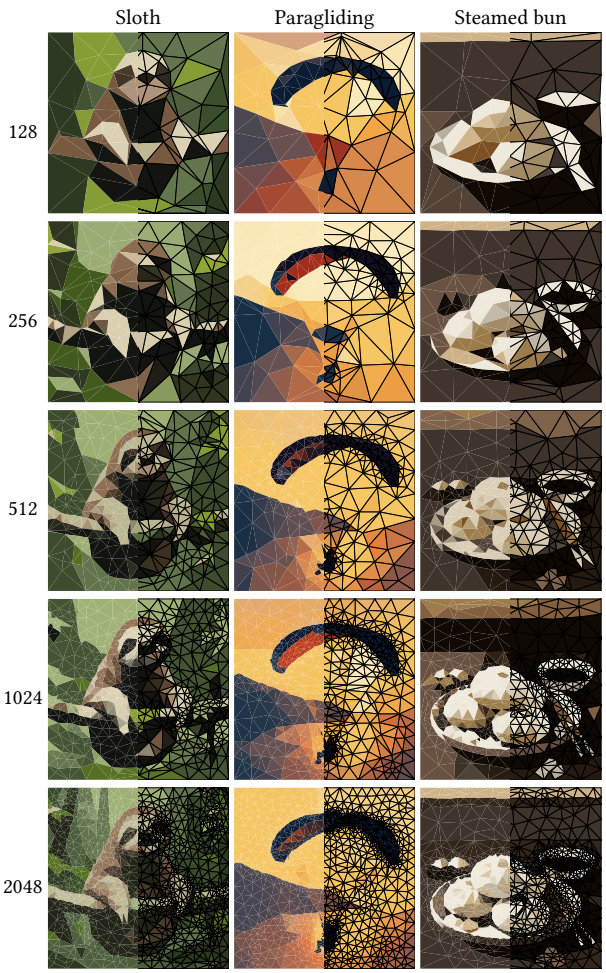


Fig. 10. Effect of triangle count (rows) on abstraction level. From top to bottom: 128, 256, 512, 1024, and 2048 triangles. Easier to see when zoomed in on a digital screen.

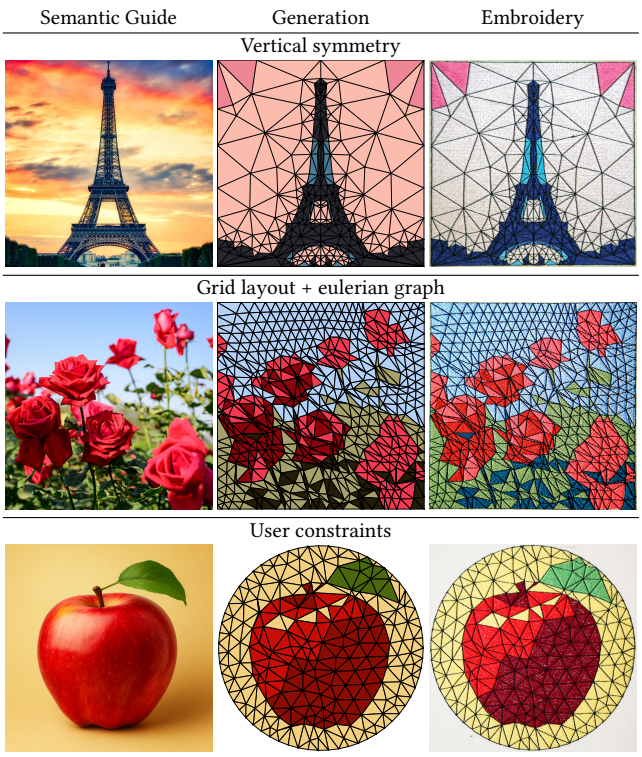


Fig. 11. Our framework can impose high-level constraints: it can make outputs vertically symmetric even from asymmetric guidance. With a grid layout and Eulerian boundary, it yields a regular low-poly look and a single continuous stitching path. Using fixed points on a circle and blue-noise sampling inside, then optimizing only interior points, it produces a circular, highly regular embroidery pattern.

4.2 Control of palette, triangle count and mesh quality

We test our approach for a range of palettes and triangle counts, with results shown in Fig. 2 and Fig. 10. Additional results are in the Supplemental. At low triangle counts, classic triangulation methods often struggle to maintain recognizability, particularly when restricted to diverse color palettes. Low triangle counts are highly relevant because they produce a clearer low-poly style, are easier to fabricate and reduce material waste. Alps demonstrates superior abstraction capabilities by leveraging semantic guidance. Furthermore, our results can be refined by adjusting default parameters and incorporating user-defined density annotations. Additionally, the triangulations are more even for Alps than for the other methods. This high geometric quality is confirmed by the metrics in Table 2. Our generated meshes exhibit significantly higher fairness scores and a lower percentage of sharp angles compared to baseline methods. For Garland and Heckbert [1998], all 15 triangulations contain intersections with 22 intersecting triangles on average, and Lawonn and Günther [2019] get intersections on 8 out of 15 triangulations with 10 intersections on average (see Fig. 6). Some of our results show jagged lines and have trouble with sharp features. This can be adjusted by (locally) tuning the weight for the triangle fairness loss.

4.3 Structural constraint control

Alps can impose explicit geometric and combinatorial constraints during generation, such as exact symmetry and Eulerian graphs for fabrication. We showcase these properties in Fig. 11.

Symmetry. To enforce a perfectly symmetric triangulation and color field, we optimize the mesh only on the left half of the domain and mirror the mesh across the symmetry axis before rendering. Vertices on the symmetry axis are snapped to the axis using $\mathcal{L}_{\text{border}}$.

Eulerian graphs. Some fabrication methods, such as continuous-path embroidery or single-stroke plotting, are greatly simplified when every edge of the mesh can be traversed exactly once in a single path (Eulerian graph). Enforcing this property during optimization is non-trivial, so in Fig. 11, we start from a grid layout with the Eulerian graph property. This property is preserved by fixing the mesh topology during the optimization.

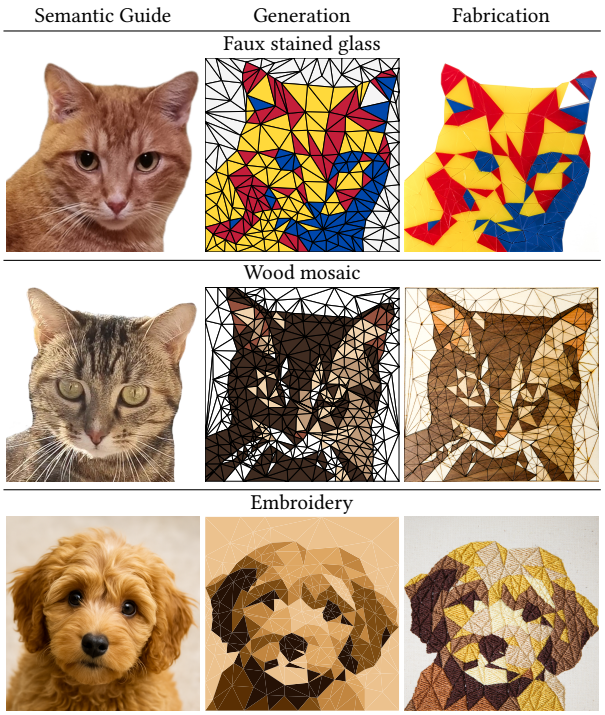


Fig. 12. The fixed color palette and overlap-free planar triangulation make our output suitable for fabrication such as embroidery, faux stained glass or wood mosaics. Each distinct color is assigned to a separate thread color or cutting layer, and we can convert it to an embroidery format, or the individually cut pieces can be assembled to reconstruct the generated image.

User inputs. Our method naturally accommodates user-specified constraints. For example, in the circular embroidery case of Fig. 11, we fix a ring of evenly spaced vertices on the boundary and initialize the interior with blue noise sampling. During optimization, only the interior points move, preserving the outer circle and yielding a clean, regular embroidery pattern.

5 Applications

Our algorithm generates low-poly meshes from images or text prompts using specific color palettes, offering three key advantages for fabrication: color quantization, vector output with a valid triangle mesh, and geometric abstraction. The resulting straight edges and convex triangles simplify cutting and assembly, while the triangulation quality can be tuned for fabrication needs. This section demonstrates several processes that benefit from our approach. Additional details are provided in the Supplemental.

Mosaics. Traditional stained glass and wood marquetry rely on assembling discrete, colored segments, making our low-poly output an ideal basis for design. We realize these crafts by merging triangles of the same color and generating vector cutting paths. We merge connected edges into polylines and assign single cut lines to shared edges, preventing redundant cutting. The resulting pieces, cut from acrylic or wood, are assembled to reconstruct the image (Fig. 12).

Fig. 13. ALPS generates a low-poly simplification of the input image that is easier to crochet while preserving key shape features. Unlike the commonly used pixel-grid approach (left), low-poly crocheting (right) reduces aliasing artifacts.

Embroidery. Our designs can be fabricated using standard embroidery machines. We export the mesh to SVG and generate stitch plans using Ink/Stitch [The Ink/Stitch Project 2026]. To optimize fabrication, we minimize jump stitches by computing a vertex-disjoint path cover on the dual graph of the triangulation and we optimize outlines using Eulerian trails to allow continuous stitching (see Supplemental for details). The resulting satin-stitched patterns closely reproduce the low-poly aesthetic (Fig. 11 and Fig. 12).

Tapestry Crocheting. Standard tapestry crocheting (pixel-grid crocheting) relies on row-by-row rasterization, which suffers from aliasing and requires frequent, error-prone and time-consuming color changes. Our method enables a patch-based approach: each polygonal face is crocheted independently using a single yarn color and then stitched together. This avoids mid-row color switches entirely, significantly reducing fabrication time and complexity. For example, the low-poly panda in Fig. 13 required less than half the time of its pixel-grid counterpart (14 vs. 30 hours) while producing a cleaner, stylized aesthetic.

Patchworking. Our mesh provides a robust framework for patchwork, where fabric panels are stitched along straight seams. The mesh edges define these seams, ensuring they are of equal length for easy alignment. To reduce assembly effort, we merge adjacent triangles of the same color into larger polygonal patches. The resulting sewing patterns are efficient to cut and assemble, as shown in the Rosy Maple Moth example (Fig. 14).

Stencil Printing. Stencils require connectivity so that inner regions do not detach. Our mesh naturally solves this: by retaining material along the edges of the triangulation (insetting the triangles), the mesh structure acts as a supporting bridge network. This

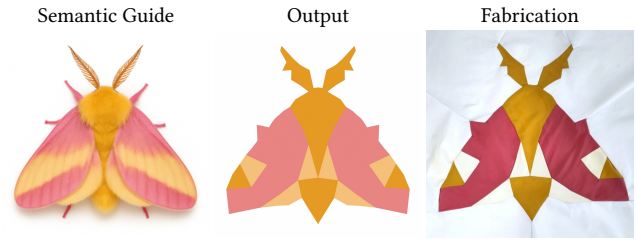


Fig. 14. Patchwork result: we generate a color-quantized Rosy Maple Moth with low polygonal count to use it for patchworking. The resulting polygons are used as a sewing pattern to be joined together into the final fabrication.

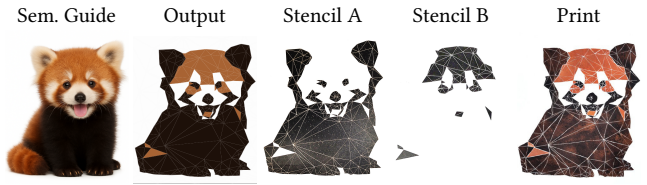


Fig. 15. We generate a color quantized version of a red panda, which we then transform into stencil patterns that can be produced by a laser cutter. Stencil A is used for the darker brown, while stencil B is for the orange-brown areas. The final print was obtained by sponge painting.

ensures structural stability without the need for complex multi-layer splitting or manually placed bridges. We demonstrate this in Fig. 15, where a laser-cut stencil was used to sponge-paint a panda design.

6 Discussion and conclusion

This paper introduces Alps, a method to generate low-poly vector graphics. Alps integrates three main contributions: an optimization procedure that guarantees non-intersecting triangles with precise control of quality through regularization and initialization; mesh-topology optimization using a fine-to-coarse strategy, guided by SDS; and a palette-based color assignment of mesh faces that supports changing mesh topology and quantization. We compared the results to prior methods, showing that our approach guarantees the desired properties, where others fail, while producing visually appealing results. Finally, we highlighted the relevance of our approach in fabrication-oriented examples.

Limitations. Alps guarantees valid low-poly vector output but shares some drawbacks of score distillation sampling. It is slower and requires more memory than feed-forward generators would, but does not require training a network or assembling a substantial dataset. The tendency of SDS to produce oversaturated colors is mitigated by our palette-based approach. The semantic guidance that drives our optimization can also pull the result away from the exact image input, so fidelity to a source image is lower than with purely pixel-based approaches. We expect advances in diffusion models to reduce both the runtime and quality gap. As a gradient-descent-based approach, Alps can land in local minima. In practice, most triangulations were satisfactory, but addressing this could improve the quality of our outputs.

Future work. One direction for future work is to further explore the use of arbitrary polygons, which we achieved through post-processing by merging adjacent triangles with equal colors for the stencil fabrication. A related direction is to relax the straight-edge requirement and allow curved edges in the mesh. This would require careful optimization to avoid self-intersections between boundaries. Outside of low-poly stylization, the ideas in this work could contribute to image compression and 3D mesh simplification by including semantic information to guide the removal of certain geometry, e.g., simplifying a cluttered table to a table without clutter.

Acknowledgments

We thank the anonymous reviewers for their constructive feedback. This work was supported in part by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 101003104, ERC CoG MYCLOTH) and SNSF BRIDGE PoC (grant agreement No. 40B1-0_239640).

References

Michael D. Adams. 2013. A highly-effective incremental/decremental Delaunay mesh-generation strategy for image representation. *Signal Process.* 93, 4 (2013), 749–764. doi:10.1016/J.SIGPRO.2012.09.017

Ellie Arar, Yarden Frenkel, Daniel Cohen-Or, Ariel Shamir, and Yael Vinker. 2025. SwiftSketch: A Diffusion Model for Image-to-Vector Sketch Generation *(SIGGRAPH Conference Papers '25)*. Association for Computing Machinery, New York, NY, USA. doi:10.1145/3721238.3730612

Alexandre Binninger and Olga Sorkine-Hornung. 2024. SD-πXL: Generating Low-Resolution Quantized Imagery via Score Distillation. (2024). doi:10.1145/3680528.3687570

Alexandre Binninger, Ruben Wiersma, Philipp Herholz, and Olga Sorkine-Hornung. 2025. TetWeave: Isosurface Extraction using On-The-Fly Delaunay Tetrahedral Grids for Gradient-Based Mesh Optimization. *ACM Transactions on Graphics* 44, 4 (8 2025). doi:10.1145/3730851 SIGGRAPH 2025 issue.

Zheng-Jun Du, Liang-Fu Kang, Jianchao Tan, Yotam I. Gingold, and Kun Xu. 2023. Image vectorization and editing via linear gradient layer decomposition. *ACM Trans. Graph.* 42, 4 (2023), 97:1–97:13. doi:10.1145/3592128

Jean-Dominique Favreau, Florent Lafarge, and Adrien Bousseau. 2017. Photo2clipart: image abstraction and vectorization using layered linear gradients. *ACM Trans. Graph.* 36, 6 (2017), 180:1–180:11. doi:10.1145/3130800.3130888

Meng Gai and Guoping Wang. 2016. Artistic Low Poly rendering for images. *Vis. Comput.* 32, 4 (2016), 491–500. doi:10.1007/S00371-015-1082-2

Michael Garland and Paul S. Heckbert. 1998. Simplifying surfaces with color and texture using quadric error metrics. In *9th IEEE Visualization Conference, IEEE Vis 1998, Research Triangle Park, North Carolina, USA, October 18-23, 1998, Proceedings*, David S. Ebert, Holly E. Rushmeier, and Hans Hagen (Eds.). IEEE Computer Society and ACM, 263–269. doi:10.1109/VISUAL.1998.745312

Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. 2022. Shape, Light, and Material Decomposition from Images using Monte Carlo Rendering and Denoising. 35 (2022), 22856–22869. https://proceedings.neurips.cc/paper_files/paper/2022/file/8fcb27984bf16ca03cad643244ec470d-Paper-Conference.pdf

Ryan Hayashi. 2025. Triangula: Generate high-quality triangulated and polygonal art from images. https://github.com/rh12503/triangula. GitHub repository, MIT License.

Xiaodi Hou, Jonathan Harel, and Christof Koch. 2012. Image Signature: Highlighting Sparse Salient Regions. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 1 (2012), 194–201. doi:10.1109/TPAMI.2011.146

Yixin Hu, Teseo Schneider, Xifeng Gao, Qingnan Zhou, Alec Jacobson, Denis Zorin, and Daniele Panozzo. 2019. TriWild: Robust Triangulation with Curve Constraints. *ACM Trans. Graph.* 38, 4, Article 52 (July 2019), 15 pages. doi:10.1145/3306346.3323011

Alec Jacobson, Daniele Panozzo, et al. 2018. libigl: A simple C++ geometry processing library. https://libigl.github.io/.

Ajay Jain, Amber Xie, and Pieter Abbeel. 2023. VectorFusion: Text-to-SVG by Abstracting Pixel-Based Diffusion Models. 1911–1920. https://openaccess.thecvf.com/content/CVPR2023/html/Jain_VectorFusion_Text-to-SVG_by_Abstracting_Pixel-Based_Diffusion_Models_CVPR_2023_paper.html

Philumon Joseph, Binsu C Kovoor, and Job Thomas. 2024. A Delaunay Triangulation-Based Low-Poly Image Abstraction. *Journal of Image and Graphics* 12, 4 (2024).

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6980

Yu-Kun Lai, Shi-Min Hu, and Ralph R. Martin. 2009. Automatic and topology-preserving gradient mesh generation for image vectorization. *ACM Trans. Graph.* 28, 3 (2009), 85. doi:10.1145/1531326.1531391

Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. 2020. Modular Primitives for High-Performance Differentiable Rendering. *ACM Transactions on Graphics* 39, 6 (2020).

Kai Lawonn and Tobias Günther. 2019. Stylized Image Triangulation. *Computer Graphics Forum* 38 (2019), 221–234. Issue 1. doi:10.1111/cgf.13526

Tzu-Mao Li, Michal Lukác, Michaël Gharbi, and Jonathan Ragan-Kelley. 2020. Differentiable vector graphics rasterization for editing and learning. *ACM Trans. Graph.* 39, 6 (2020), 193:1–193:15. doi:10.1145/3414685.3417871

Hsueh-Ti Derek Liu, Mehdi Rahimzadeh, and Victor Zordan. 2025. Controlling Quadric Error Simplification with Line Quadrics. In *Computer Graphics Forum*, Vol. 44. Wiley Online Library, e70184.

Yiting Ma, Xuejin Chen, and Yu Bai. 2017. An interactive system for low-poly illustration generation from images using adaptive thinning. In *2017 IEEE International Conference on Multimedia and Expo, ICME 2017, Hong Kong, China, July 10-14, 2017*. IEEE Computer Society, 1033–1038. doi:10.1109/ICME.2017.8019369

David McAllister, Songwei Ge, Jia-Bin Huang, David Jacobs, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. 2024. Rethinking Score Distillation as a Bridge Between Image Distributions. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (Eds.). http://papers.nips.cc/paper_files/paper/2024/hash/3b62bca132cf5c8973b09a2fc6dc8ca6-Abstract-Conference.html

Seyedali Mostafavian and Michael D. Adams. 2015. An optimization-based mesh-generation method for image representation. In *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, PACRIM 2015, Victoria, BC, Canada, August 24-26, 2015*. IEEE, 234–239. doi:10.1109/PACRIM.2015.7334840

Baptiste Nicolet, Alec Jacobson, and Wenzel Jakob. 2021. Large steps in inverse rendering of geometry. *ACM Trans. Graph.* 40, 6 (2021), 248:1–248:13. doi:10.1145/3478513.3480501

Björn Ottosson. 2020. A perceptual color space for image processing. https://bottosson.github.io/posts/oklab/. Accessed: 2025-01-13.

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2023. *Physically based rendering: From theory to implementation*. MIT Press.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. 2023. DreamFusion: Text-to-3D using 2D Diffusion. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/forum?id=FjNys5c7VyY

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. arXiv:2103.00020 [cs.CV] https://arxiv.org/abs/2103.00020

Christoph Schuhmann. 2023. Improved Aesthetic Predictor. https://github.com/christophschuhmann/improved-aesthetic-predictor. Last accessed on January 19, 2024.

Adrian Secord. 2002. Weighted Voronoi Stippling. In *Proceedings of the 2nd International Symposium on Non-photorealistic Animation and Rendering* (New York, NY, USA) *(NPAR '02)*. Association for Computing Machinery, 37–43. doi:10.1145/508530.508537

Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp, and Jun Gao. 2023. Flexible Isosurface Extraction for Gradient-Based Mesh Optimization. *ACM Trans. Graph.* 42, 4 (July 2023). doi:10.1145/3592430 Place: New York, NY, USA.

Ka Chun Shum, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. 2025. Color Alignment in Diffusion. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*. 28446–28455.

Sanghyun Son, Matheus Gadelha, Yang Zhou, Zexiang Xu, Ming C. Lin, and Yi Zhou. 2024. DMesh: A Differentiable Mesh Representation. http://arxiv.org/abs/2404.13445 arXiv:2404.13445 [cs].

Jian Sun, Lin Liang, Fang Wen, and Heung-Yeung Shum. 2007. Image vectorization using optimized gradient meshes. *ACM Trans. Graph.* 26, 3 (2007), 11. doi:10.1145/1276377.1276391

The Ink/Stitch Project. 2026. Ink/Stitch: Open source machine embroidery design. https://inkstitch.org/ Version 3.2.2.

T. Uasmith, T. Pukkaman, and Peeraya Sripian. 2017. Low-poly image stylization. *Journal for Geometry and Graphics* 21 (01 2017), 131–139.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL] https://arxiv.org/abs/1706.03762

Wanyi Wang, Zhonggui Chen, Lincong Fang, and Juan Cao. 2024. Curved Image Triangulation Based on Differentiable Rendering. *Comput. Graph. Forum* 43, 7 (2024), i–xxii. doi:10.1111/CGF.15232

Yanyang Xiao, Juan Cao, and Zhonggui Chen. 2022. Image Representation on Curved Optimal Triangulation. *Comput. Graph. Forum* 41, 6 (2022), 23–36. doi:10.1111/CGF.14495

Ximing Xing, Qian Yu, Chuang Wang, Haitao Zhou, Jing Zhang, and Dong Xu. 2025. SVGDreamer++: Advancing Editability and Diversity in Text-Guided SVG Generation. *IEEE Trans. Pattern Anal. Mach. Intell.* 47, 7 (2025), 5397–5413. doi:10.1109/TPAMI.2025.3547889

Ximing Xing, Haitao Zhou, Chuang Wang, Jing Zhang, Dong Xu, and Qian Yu. 2024. SVGDreamer: Text Guided SVG Generation with Diffusion Model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*. IEEE, 4546–4555. doi:10.1109/CVPR52733.2024.00435

Yongyi Yang, Miles N. Wernick, and Jovan G. Brankov. 2003. A fast approach for accurate content-adaptive mesh generation. *IEEE Trans. Image Process.* 12, 8 (2003), 866–881. doi:10.1109/TIP.2003.812757

Wenli Zhang, Shuangjiu Xiao, and Xin Shi. 2015. Low-poly style image and video processing. In *International Conference on Systems, Signals and Image Processing, IWSSIP 2015, London, UK, September 10-12, 2015*. IEEE, 97–100. doi:10.1109/IWSSIP.2015.7314186