Journal of Innovative Image Processing (ISSN: 2582-4252)
www.irojournals.com/iroiip/

A Two-Level Simplification Framework for Efficient Low-Poly Image Abstraction from AI-Generated Images

Philumon Joseph[1], Binsu C. Kovoor[2], Job Thomas[3]

School of Engineering, CUSAT, Kerala, India.

E-mail: [1]philumon@cusat.ac.in, [2] binsu@cusat.ac.in, [3]job_thomas@cusat.ac.in

Abstract

Low-polygon image representation is commonly employed into simplify an image to reduced geometrical data. This paper proposes a method to generate a low-poly image from an AI-generated image. The proposed framework is based on a two-level simplification technique to achieve a considerable reduction in resolution while maintaining good visual quality. First, significant feature points are extracted from the segmented portions of the AI-generated image to obtain essential structural information. Subsequently, hexagonal grid-based sampling, which allows for the identification of crucial seed points while respecting significant visual elements, is employed. The Low-poly style is achieved through Delaunay triangulation, and the colors are taken directly from the original AI image to ensure visual consistency. Furthermore, we have compared the performance of entropy and saliency maps when used in the selection process for the hexagonal grid. We present experimental results demonstrating a pixel-to-point reduction of over 99.1%, as well as the ability to compress high-resolution images into a few simple points in a way that powerfully preserves perceptual integrity. Both qualitative and quantitative analyses were conducted on the AI-generated images at multiple resolutions, observing the sensitivity and scalability of the method for generating low-poly images.

Keywords: Low Poly Image Abstraction, AI-Generated Images, Hexagonal Grid-based Sampling, Delaunay Triangulation.

DOI: https://doi.org/10.36548/jiip.2025.2.012
Received: 13.05.2025, received in revised form: 11.06.2025, accepted: 25.06.2025, published: 04.07.2025

1. Introduction

The convergence of Artificial Intelligence (AI) and digital art has ushered in a new era of creative expression, where advanced algorithms enable the generation of highly detailed and photorealistic images. Among these advancements, low-poly image generation has emerged as a compelling approach to simplifying AI-generated content while retaining its core aesthetic and structural qualities. Low-poly art, characterized by the use of geometric shapes, typically triangles, to create simplified representations of complex images, offers a minimalist yet visually striking alternative to more detailed representations.

AI-generated images exhibit a harmonious integration of smooth regions and intricate high-frequency details, with significant objects distinctly delineated by sharp edges and contours. Flat regions that are sufficiently uncomplicated to preserve visual coherence coexist with intricate textures. Blurred backgrounds create a depth-of-field effect that accentuates sharper, more intricate foreground elements, while vivid color variations and subtle gradients enhance visual appeal. The image's richness and simplicity are harmonized by the geometric rhythm established by structural recurrence. AI-generated images contrast with natural images by exhibiting crisper and more defined edges. AI outputs typically exhibit less biological variability, enhanced color vibrancy, and more seamless transitions. Due to the absence of intrinsic imperfections in depth-of-field effects, structural repetition may appear too precise. A significant problem in the low-poly abstraction of AI-generated images is the intentional selection of key spots to simplify complex pictures while maintaining essential aspects.

AI-generated image tools can utilize several input formats, including text prompts, reference photographs, and sketches, to generate a broad spectrum of outputs. Reference images and sketches aid the AI in integrating specific visual attributes, styles, or color palettes, whereas language prompts enable users to specify the scenario or object they envision. Among the well-known AI image generation programs are Adobe Firefly [1], DALL-E 3 [2], Canva [3], Meta AI [4], and Stable Diffusion [5]. Diffusion models, neural networks, and other advanced approaches are used to transform text prompts, reference images, and sketches into a range of stylized graphics.

The process of converting an AI-generated image into low-poly image abstraction is examined in this paper, along with the creative and technological difficulties associated with the endeavor. This study's main goal is to create a new hybrid form of digital art by bridging

the gap between classic artistic abstraction approaches and AI-driven image generation. Our goal is to provide artists and designers with a new tool for creative expression by reducing complex AI outputs to low-poly forms. This will allow the richness of AI-generated content to be reinterpreted through the prism of geometric simplicity. In order to preserve important visual components during simplification, the process of creating low-poly images from AI-generated content requires careful handling of intricate textures and subtle color changes. This paper examines how to apply Delaunay triangulation, edge preservation tactics, and color averaging approaches to preserve visual accuracy while accomplishing significant simplification. Widely used in resource-efficient rendering, low-poly AI-generated images are ideal for environments with limited computing power, such as mobile devices and virtual reality. By simplifying visual content while not significantly affecting perceptual quality, they provide effective data representation, crucial for quicker transmission and reduced storage needs. Through the simplification of complex models, low-poly AI visuals enhance game development speed, enabling faster loading times and more fluid gameplay. Additionally, real-time applications like augmented reality use these visuals, where quick rendering and a polished look are essential for creating immersive experiences.

This effort aims to reliably construct low-poly image abstractions from images created by artificial intelligence. Direct simplification of high-resolution AI-generated images poses a considerable challenge: a direct reduction either eliminates critical edges and textures, diminishing visual fidelity, or excessively clusters vertices in complex areas, leading to uneven mesh density that reduces performance and consistency. We propose a two-level simplification framework: the initial stage utilizes feature-based extraction to retain essential edges and prominent regions, while the second stage implements hexagonal-grid refinement to uniformly reduce those points, resulting in low-poly meshes that are both accurate and evenly distributed.

This paper's primary contributions are as follows:

- To provide a novel technique for creating low-poly representations from AI-generated images. The method allows for effective abstraction while preserving crucial visual characteristics through a two-level image simplification and accurate reconstruction process.

- Introduce a two-level simplification strategy to strategically preserve important visual information using region-based, entropy-based, and saliency-based approaches, each of which uses a hex grid for point selection prior to Delaunay triangulation.
- The suggested technique compresses high-resolution images into a small number of seed points while preserving visual quality and perceptual integrity, exhibiting a notable 99.1%pixel count decrease.
- Exhibiting the effectiveness of the suggested methodology through a series of quantitative and qualitative assessments, including comparisons to current techniques.

2. Related Work

Most research works dealing with low-poly image synthesis are related to Delaunay triangulation, which is able to generate well-distributed triangle meshes that closely approximate the meaningful features of complex images. In this paper, we review various methods and refinements for approaches to selecting seed points and coloring triangles that are necessary for maintaining visual fidelity during the simplification process.

M. Gai et al. [6] employ an Edge Drawing-based technique [7] and a modified Douglas-Peucker algorithm to simplify dense edge vertices while preserving only the most significant information. Delaunay triangulation is applied to the L component of the Lab color space, resulting in triangles with median colors for balanced abstraction, while a saliency detection algorithm facilitates non-uniform sampling to focus on foreground areas. W. Zhang et al. [8] utilize a Sobel edge-detection shader to identify edges within the image, subsequently generating triangles through Delaunay triangulation, with each triangle colored based on the color of its center of mass in the original image. C. J. Qian et al. [9] apply a Laplacian filter to locally detect edges following the integration of horizontal and vertical blurs by interpolation, so achieving a comprehensive blur effect. Delaunay triangulation is employed to enhance color clarity and definition in the low-poly output by generating triangles, with each triangle's color sampled at its centroid instead of being averaged over all pixels. T. Uasmith [10] proposes a seed point selection technique that uses a bilateral filter to smooth the input image, preserving significant edges while reducing noise. After that, K-means clustering is used to segment the image, producing areas with comparable hues or intensities. The Ramer-Douglas-Peucker (RDP) technique is then used to simplify contours in order to preserve key shape attributes

while lowering complexity. Following the selection of seed points, Delaunay triangulation is used to build a mesh over the image. The low-poly effect is then achieved by averaging the coloring of each triangle. Y. Ma et al. [11] use an over-segmentation technique to extract several vertices from the corners of notable regions in order to capture the image's structural information. Adaptive thinning is used to rank these vertices, giving high saliency vertices priority. Furthermore, the technique uses Delaunay triangulation and lightness difference to produce numerous low-poly representations from a single image by allowing users to interactively change the number of points, triangle contrast, and local regions. To create a triangular mesh, R. Ng et al. [12] proposed a method that first combines edge, saliency, and face detection to identify important features. Next, it uses Delaunay triangulation to assign colors to triangles based on the average pixel color. K. Lawonn et al. [13] present image triangulation as an optimization issue and uses an interactive system to partition triangles according to refinement criteria and modify vertex placements in order to minimize the root mean squared approximation error. The system facilitates initial triangulation through either a traditional layout or Delaunay triangulation informed by importance sampling (e.g., saliency maps), and allows for seamless editing and re-triangulation via efficient GPU-based rasterization, offering various rendering styles such as textures, tonal maps, gradients, and solid colors. In order to identify important characteristics and produce seed points, the Pic2PolyArt [14] framework integrates saliency, edge, and face identification, providing a single method for geometric abstraction that allows for both triangle-based and polygon-based representations. O. X. Laske et al. [15] sharpen the image and use the Sobel operator to find conspicuous edges. Delaunay triangulation is then used to triangulate critical edge pixels. To produce a refined triangulated image, each triangle is colored according to its centroid, which is determined by averaging its vertex coordinates. P. Joseph et al. [16] proposed an entropy-based method that smooths the input image using an anisotropic diffusion filter, then computes an entropy map to record fine-grained pixel changes. Selecting seed points for Delaunay triangulation from high-entropy regions enables an optimal geometric arrangement in which a specialized color selection module colors each triangle, resulting in a refined low-poly image through rendered triangulation. P. Joseph et al. [17] suggested a region-based seed point technique that preserves key image elements and improves visual attractiveness by precisely capturing object boundaries, hence enhancing low-poly image abstraction. By minimizing triangle data, combining randomly assigned points within regions with boundary points to form seed points for Delaunay triangulation, allowing for controlled point density based on segment

area, and using the average color of the pixels within each triangle to create a cohesive visual representation, this technique minimizes the complexity of the original image while optimizing storage efficiency and reconstruction. Low-poly abstraction has become increasingly popular in recent years, particularly in the fields of graphic design and video game creation.

A few games that use low polygonal models are Don't Mess with Texas [18], Superhot 19], Morphite [20], and Astroneer [21]. Low poly abstraction has been used in everything from animated movies to advertising campaigns, and it has emerged as a crucial tool for creative professionals seeking to create distinctive and appealing visuals [22]. The low-poly image style has garnered considerable popularity across multiple domains, especially in digital art, design, and interactive systems. This style employs a limited quantity of triangles to produce abstract representations, rendering it aesthetically pleasing and efficient for both artistic and functional purposes.

Table 1. Summary of Related Works

Study & Year	Feature Point Selection	Coloring	Image Resolution	Dataset
[6], 2015	Gaussian Filtering, Gradient Magnitude	L component, median part of the colors	954 × 637	Custom dataset
[8], 2016	Sobel Edge-detection	Center of mass	1024 × 1024	Custom dataset
[9], 2016	Laplacian Filter	Center of gravity	NA	Custom dataset
[10], 2017	Bilateral Filter, K-means, RDP	Average	NA	Custom dataset
[11], 2017	Over-segmentation, Adaptive Thinning	Lightness difference	717 × 537	Custom dataset
[12], 2017	Edge Detection, Saliency, and Face Detection	Average	NA	Custom dataset
[13], 2018	Regular Layout, Saliency Map	Constant, Linear	512 × 512	Inkscape [23], Photoshop [24]
[14], 2021	Edge Detection, Saliency, and Face Detection	Average	1024 × 1024	Benchmark image set [25] and Natural Image [26]

[15], 2024	Laplacian Filter, Sobel Operator	Centroid	NA	Custom dataset
[16], 2024	Anisotropic Diffusion Filter, Entropy map	Average	1024 ×1024	Benchmark image set [25] and Natural Image [26]]
[17], 2024	Region Segmentation	Average	1024 ×1024	Benchmark image set [25] and Natural Image [26]

Neural rendering combines conventional graphics and deep learning to create photorealistic images from scene descriptions or observations. C. Zeng et al. [27] developed RenderFormer, which converts triangle tokens with reflectance attributes into pixel patches under full global illumination without per-scene training or fine-tuning. Q. Wang et al. [28], developed IBRNet, which learns view synthesis from multi-view posed images using fully differentiable volume rendering, while V. Sitzmann et al. [29] proposed, Light Field Networks, which model scenes as a four-dimensional light field using a neural implicit representation for single-evaluation rendering that captures geometry and appearance. C. Wu et al. [30] proposed differentiable rasterizers provide dependable end-to-end gradient flow without human tweaking by rewriting hard visibility criteria as continuous “soft” functions. T. M. Li et al. [31] present differentiable vector graphics rasterization for editing and learning that computes unbiased pixel-to-curve gradients with quick analytical prefiltering and high-quality multisample anti-aliasing. H. Yuan et al. [32] apply differentiable rendering to CSG models using boolean-primitive rasterization with intersection anti-aliasing, while Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning, proposed by S. Liu et al. [33], sends silhouette, shading, and color signals directly to mesh attributes.

According to the literature, edge detection, entropy maps, saliency maps, and region segmentation are prominent techniques for choosing feature points from images to generate low-poly images. Most methods employ average triangle coloring and are based on bespoke datasets, natural image datasets, and benchmark image datasets, with input dimensions typically limited to 1024 × 1024 pixels. Notably, high-resolution AI-generated images have not yet been used to create low-poly image abstractions; this paper tries to fill that gap by using high-resolution AI-generated images to produce low-poly abstractions.

3. Proposed Work

The proposed approach for generating a low-poly image from an AI-generated image uses a two-level simplification procedure to select seed points and utilizes Delaunay triangulation to achieve a visually pleasing low-poly representation. The detailed framework is shown in Figure 1.

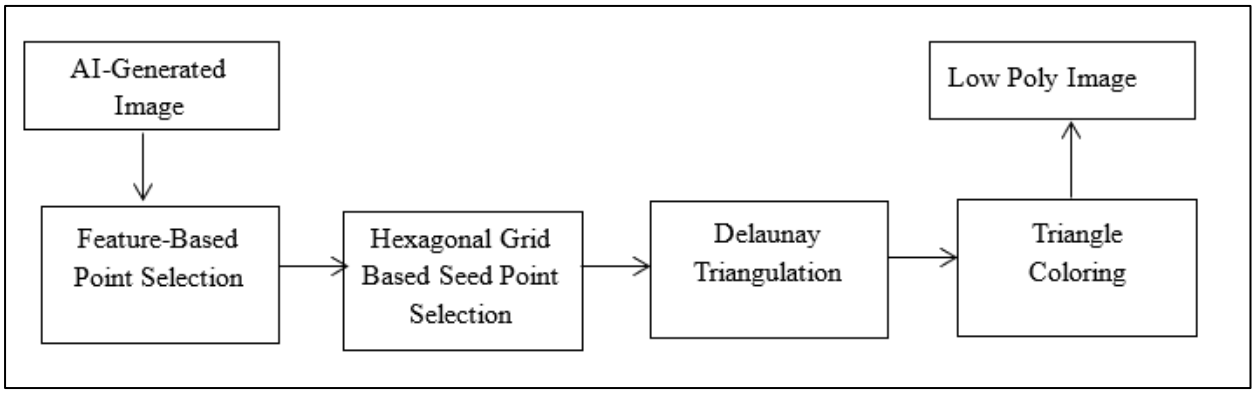

Figure 1. The Proposed Framework for Generating Low Poly Image from AI-Generated Image

In general, the two-level framework proceeds as follows: the first level of simplification is based on Feature-Based Point Selection; this could be edges, regions of high deviation (entropy), salient areas, or mixtures of border and internal region sampling. The second level of simplification is on Hexagonal Grid-Based Seed Point Selection. Superimpose a uniform hexagonal grid and, inside each cell, identify one or more representative points (e.g., the most prominent feature response) to uniformly reduce and distribute the seed points. Finally, execute Delaunay triangulation on the simplified point set, and color each triangle by sampling the original image to get the low-poly result.

3.1 First-Level Simplification: Region-based Feature Point Selection

For the simplification of low-poly images, emphasizing regions with properly selected seed points on the boundaries and inside the regions enhances the visual coherence and structure of the result. Such a selection retains major contours and captures fine details, so the simplified image keeps the main visual features, which is the base for realism and aesthetics of the abstraction. P. Joseph et al. [17] proposed a method that segment the image using k-means clustering to group pixels by color similarity, forming distinct regions. Mark boundary points by detecting color differences between neighboring pixels, then select points along each

boundary at set intervals and add random points within each region for a balanced distribution. Figure 2 shows the twolevel simplification process. The AI-generated image, which has a resolution of 2304 × 1792 pixels, is shown figure 2(a) and a region-based seed point generation approach was used to generate the seed points in figure 2(b), yielding 527,314 points. These points are from region boundaries and within the regions. This many points frequently resulted in the formation of clusters in specific places, which is undesirable for the production of low-poly images since it adds needless complexity. Due to the high density of points, a two-level simplification procedure was required. A hexagonal grid-based method was used to eliminate point clusters in order to remedy this and guarantee a more even distribution of points throughout the image.

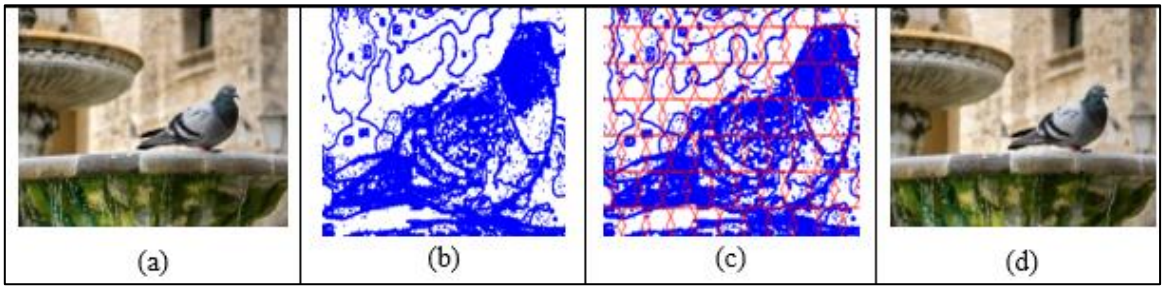

Figure 2. Two-Level Simplification Process. (a) Input AI Image, (b) Region-based Feature Point Selection, (c) Hexagonal Grid-based Seed Point Selection (d) Low-Poly AI Image

3.2 Second-Level Simplification: Hexagonal Grid-based Seed Point Selection

AI-generated images contain a large number of feature points and improper selection of points that lead to clusters of points in some areas. This irregular distribution of points creates some noise or non-aesthetic visual patterns when Delaunay triangulation is applied to the data. Long, skinny triangles generated by an unstructured set of feature points can degrade the quality of the low-poly image. As the points are spread evenly across the entire region, oversampling in some areas and undersampling in others are reduced through grid-based seed point selection. In the second step, a hexagon grid is overlaid, and the seed points are automatically selected based on feature density in each hexagon to ensure geometric accuracy and consistent representation. Since hexagonal grids have 6 equally spaced surrounding points, they are more efficient for spatial coverage than square grids. The coverage is maintained or enhanced by the seed points, which are picked from a hexagonal lattice but with fewer points. By decreasing the number of data points while improving representation and preserving detail,

this will simplify the process. Once the seed points are selected, Delaunay triangulation is applied to generate a mesh of triangles that represent the image geometry. Each triangle is colored with an average color taken from the input image, resulting in a visually striking low-poly image.

In algorithm 1, the first step is to plot input point set P_{set} on the image plane, and then a hexagonal grid H_{Grid} is superimposed on the plane with the seed points. A subset of the original set P_i is contained in each hexagonal cell H_i. The density of each cell is determined by counting the number of points clustered in high-density, medium-density and low-density groups. Point selection criteria are as follows: low-density cells retain reduced points (one quarter of the maximum points), while medium-density cells retain half of the maximum points, and high-density cells keep the user-defined maximum number of points. Finally, these conditions are employed to randomly sample the right number of points in each hexagonal cell, generating a reduced set of points and resulting in more uniform coverage. After this secondary reduction, the point set along with the hex grid is displayed in Figure 2(c). Delaunay triangulation is then applied, and each triangle is filled with an average color to create the final low-poly image. The proposed method preserves the original image's fundamental structure and color distribution while lowering its visual complexity. Figure 2(d) displays the low-poly image that was produced. This suggested approach makes the procedure scalable to larger images or more intricate designs by lowering the computational complexity and memory utilization.

Algorithm 1: Hex Region Low Poly

Input:

$P_{set} = \{p1, p2, \ldots . pn\}, High_{Threshold}, Medium_{Threshold}, Low_{Threshold}, I_{Height}, I_{Width}, Hex_{Density}, Max_{points}$

Output: $Points_{Combined}$

Step 1: Plot the points set P_{set} on the image plane

Step 2: Determine Hexagon Size

- Total Area of the Image $I_{Area} = I_{Height} \times I_{Width}$
- Desired area of a hexagon $Desired_{Hex\,Area} = I_{Area} \times Hex_{Density}$

- Area of a regular hexagon $Hex_{Area} = \frac{3\sqrt{3}}{2} \times Side^2$
- Side length(radius)

$$(\ [\![Side]\!]\ \hat{}2 = Hex(\ [\![\_Area\hat{}\ ]\!]\ )/((\sqrt[3]{(3\ )}\ )/2)$$

$$Side = \sqrt{(Hex((\ [\![\_Area\hat{}\ ]\!]\ )/((\sqrt[3]{(3\ )}\ )/2))}$$

$$Side = \sqrt{I_{Area} \times \frac{Hex_{Density}}{\frac{3\sqrt{3}}{2}}}$$

- Final hexagonal size $\text{Side} = \sqrt{\text{I\_(Height)} \times \text{I\_Width }/((\sqrt[3]{(3\ )}\ )/2))}$

Step 3: Create a hexagonal grid H_{Grid} over the image plane

Each hexagon $H_i \in H_{Grid}$ contains a set of points P_i

Density Calculation for Each Hexagon

$$D_i = \frac{|P_i|}{Hex_{Area}}$$

where $|P_i|$ is the number of points in H_i and Hex_{Area} is the area of the hexagon.

Step 4: Choose points for each hexagonal cell using density thresholds

$$N_{Points_{Selected}} = \begin{cases} Max_{points} & if\ D_i > High_{Threshold} \\ \frac{Max_{points}}{2} & if\ Medium_{Threshold} < D_i \leq High_{Threshold} \\ \frac{Max_{points}}{4} & if\ Low_{Threshold} < D_i \leq Medium_{Threshold} \\ Default_{points} & if\ D_i \leq Low_{Threshold} \end{cases}$$

Step 5: Choose an appropriate number of points at random from each hexagonal cell.

Step 6: Combine the points that were chosen from each hexagonal cell

Step 7: Return $Points_{Combined}$

3.3 Entropy-Guided Hexagonal Grid Sampling for Seed Point Generation

The entropy map of an image provides a measure of the level of local complexity or variability in pixel intensities throughout the entire image. It is often used to highlight areas with a lot of texture, complexity, or unpredictable fluctuations. High entropy areas are those with more pixel intensity variations, while low-entropy regions are linked to more homogenous or smooth sections. Directly analyzing pixel-level entropy over a large image is computationally expensive and slower.

P. Joseph et al. [16] presented a technique that generates seed points iteratively using a Gaussian suppression mask after first creating an entropy map. The mask subtracts the Gaussian-distributed values around the selected point, which are determined by an amplitude and sigma (spread). By preventing adjacent sites from being selected again, this ensures a more diverse and uniform distribution of seed points over the entire image. Only small-sized images are a good fit for this technique. We suggest an approach called Entropy-Based Hexagonal Grid Sampling for Seed Point Generation in low-poly Image Abstraction (Hex Entropy Low Poly) for larger images, such as AI-generated images.

Algorithm 2: Hex Entropy Low Poly

Input: I_{AI}: AI generated Image, N_{Points}: Number of Points to select, $P_{Minimun}$: Minimum points, H_{size} : Size of the Hexagonal cell

Output: $P_{Selected}$:Selected seed points

Step 1: Preprocess I_{AI} using Anisotropic Diffusion

$$D_{image} = \text{anisotropic}_{\text{diffusion}(I_{AI})}$$

Step 2: Compute the Entropy Map of the diffused_image D_{image}

$$E_{map} = Entropy_{Map(D_{image})}$$

Step 3: Create a hexagonal grid over E_{map}

$$Hex_{center} = Generate_{hex_{grid(E_{map},H_{size})}}$$

Step 4: Calculate average entropy $Avg_{Entropy}$ for each hexagon in the grid:

$$Avg_{Entropy} = Average_{Entropy(E_{map}, Hex_{center},H_{size})}$$

Step 5: Call Algorithm 3: Hexagonal Grid Based Sampling

Step 6: Return $P_{Selected}$

This algorithm uses anisotropic diffusion, which enhances significant features and reduces noise while preserving edges. An entropy map is subsequently computed to quantify the texture and complexity in various regions of the image. In the sliding window technique for calculating entropy, a predefined disk size controls the neighborhood for study. The entropy map is then initially filled with hexagonal cells. To account for local complexity, the average entropy of each cell is calculated. Use Algorithm 3 to choose seed points from each hexagonal cell.

Algorithm 3: Hexagonal Grid based Sampling

Input: $Feature_{map}$: Entropy map/Saliency map, $Avg_{\text{feature}_{\text{value}}}$: Average Feature value, H_{size} : size of the hex cell, H_T : High Threshold , M_T :Medium Threshold, L_T: Low Threshold

Output: $P_{Selected}$: Selected seed points

Step 1: Create a hexagonal grid over $Feature_{map}$

$$Hex_{center} = Generate_{hex_{grid(Feature_{map}, H_{size})}}$$

Step 2: Select points based on feature values

- Choose minimum points $P_{Minimun}$ to each hex cell in hex_centers
  - For each hex cell Hex_{cell} in Hex_{center}

$$P_{Selected} = P_{Selected} \cup \text{select}_{\text{points}_{\text{from}_{\text{cell}(Hex_{cell}\ \ Feature_{map}, Hex_{center}, P_{Minimun})}}}$$

- Determine the Remaining Points

$$P_{Remaining} = N_{Points} - (|Hex_{center}| \times P_{Minimun})$$

Step 3: Selecting Remaining Points Based on average feature value

- High feature value Cells

$$Hex_{center}(high) = \left[Hex_{cell} \in Hex_{center} : Avg_{\text{feature}_{\text{value}}}[Hex_{cell}] > H_T\right]$$

- Medium feature value Cells:

$$Hex_{center}(Medium) = \left[Hex_{cell} \in Hex_{center} : M_T \leq Avg_{\text{feature}_{\text{value}}}[Hex_{cell}] \ \leq H_T\right]$$

- Low feature value Cells:

$$Hex_{center}(Low) = \left[Hex_{cell} \in Hex_{center} : L_T \leq Avg_{\text{feature}_{\text{value}}}[Hex_{cell}] \leq M_T\right]$$

- Determine Points to Select from each Priority Group

$$〚Total〛\_(High\ \ ) = int(P\_Remaining \times 0.5)$$

$$\llbracket Total \rrbracket\_(Medium\ ) = int(P\_Remaining \times 0.3)$$

$$Total_{Low} = P_{Remaining-}\left(Total_{High} - Total_{HMedium}\right)$$

Step 4: Select Points from each cell

$$P_{Selected} = P_{Selected} \cup \text{select}_{\text{points}_{\text{from}_{\text{cell}}\left(Hex_{center}(High),\ Feature_{map},\ Hex_{center},Total_{High}\right)}}$$

$$P_{Selected} = P_{Selected} \cup \text{v}_{\text{points}_{\text{from}_{\text{cell}}(Hex_{center}(Medium),\ Feature_{map},\ Hex_{center},Total_{Medium})}}$$

$$P_{Selected} = P_{Selected} \cup \text{select}_{\text{points}_{\text{from}_{\text{cell}}(Hex_{center}(Low),\ Feature_{map},\ Hex_{center},Total_{Low})}}$$

Step 5: Return $P_{Selected}$

Algorithm 3 takes an entropy map, average entropy value, hex cell size, high threshold, medium threshold and low threshold, values as input. First create a hexagonal grid over the entropy map, and then select a minimum number of points from every hex cell to confirm the coverage. The remaining points are selected based on the following criteria: 50% of remaining points are selected from cells with high entropy, 30% of the remaining points are selected from medium entropy cells, 20% of the remaining points are selected from low entropy cells. The function select_points_from_cells selectsa proportionate number of points based on the calculated $Total_{High}$, $Total_{Medium}$ and $Total_{Low}$ values. The selected points from all cells are grouped to $P_{Selected}$, the final selected points. By balancing coverage and feature relevance, the approach enhances seed point selection for low-poly image generation.

3.4 Saliency-based Hexagonal Grid Sampling for Seed Point Generation

The geometric abstraction work Pic2PolyArt [14], has focused on saliency, edge, and face detection methods to determine an image's primary subject and produce a set of seed points. Based on the predetermined saliency ranges (0–63, 64–127, 128–191, and 192–255). seed point selection assigns the same amount of seed points to each saliency region. This type of seed point selection leads to inefficient distribution. The detailed regions continue to be underrepresented, and the less prominent regions receive more points. Issues with scalability and structural accuracy make this method unsuitable for large resolution images or AI-generated images. When working with bigger or more complex images, the accuracy of subject detection is particularly important in defining the quality of the final geometric abstraction. We propose an approach Saliency-Based Hexagonal Grid Sampling for Seed Point Generation in Low poly Image Abstraction (Algorithm 4: Hex Saliency Low Poly) that takes an AI-generated

image, the number of points to select, minimum points for each cell and size of the hexagonal cell as input parameters. By distributing a fixed number of points across hexagonal cells based on each cell's saliency rating, this algorithm ranks the visually important portions of an image in order of significance. Initially, the algorithm computes the saliency map of the input image and places the hexagonal grid over the saliency map. Then, it calculates the average saliency value in each hex cell.

Algorithm 4: Hex Saliency Low Poly

Input: AI generated Image, :Number of Points to select, :Minimum points, : Size of the Hexagonal cell

Output: $P_{Selected}$:Selected seed points

Step 1: Compute Saliency Map of input image I_{AI}

$$S = \text{StaticSaliencySpectralResidual}(I_{AI})$$

Step 2: Average Saliency Calculation

$$Avg_{Saliency}(Hex_{Cell_i}) = \frac{1}{P_i}\sum_{p \in P_i} S(p)$$

where P_i is the set of pixels within cell Hex_{Cell_i}

Step 3: Determine Saliency Thresholds

- High threshold:

$$H_T = Percentile(Avg_{Saliency}, 75)$$

- Medium threshold:

$$M_T = Percentile(Avg_{Saliency}, 25)$$

- Low threshold:

$$L_T = 0 \quad \text{\# Low threshold (all values <= 25th percentile)}$$

Step 4: Call Algorithm 3: Hexagonal Grid Based Sampling

Step 5: Return $P_{Selected}$

Using the average saliency values across all cells, cells are divided into three groups: high, medium, and low saliency. High saliency cells are those with saliency values above the 75th percentile, showing the most visually important areas. Cells that fall between the 25th and 75th percentiles are called medium saliency cells. Low saliency cells are those below the 25th

percentile, indicating they are in areas that are less visually prominent. Then, use Algorithm 3: Hexagonal grid-based sampling to choose seed points from each hexagonal cell. There is a hexagonal grid on the saliency map. The length of each side is defined by hex_size, and the location of each cell center is indicated by hex_centers. Each cell is initially assigned a minimum number of points to give more weight to areas that are very important. The remaining points are distributed based on importance: high saliency cells receive 50% of the points, medium saliency cells receive 30%, and low saliency cells receive 20%. This ensures that the map has both meaning and even coverage across all areas. The function uses select_points_from_cells for each category to draw a number of points proportional to the high_points, medium_points, and low_points that were found. When these points are added to the selected_points list, there are more points in areas with higher saliency values.

4. Results and Discussion

To test the effectiveness of the two-level simplification method, we use AI generated high-resolution images from different diffusion models. The goal of the experiment is to assess how effective the reduction processes are in the generation of low-poly images. Our particular aim is to investigate how a two-level simplification strategy can be applied to reduce the size of the point set used to Delaunay triangulate the image while maintaining the fundamental structure and visual content of the original image.

We collected AI images from the benchmark dataset "Synthbuster"[34] and other websites. Figures 3(a) through 3(d) display AI images from the Dale2 dataset. Each of these images has a resolution of 1024 by 1024 pixels and around 10 lakh pixels. The website pixabay.com [35] contain AI generated images with resolutions of 3(e) 1477 x 1920, 3(f) 1920 x 1280, 3(g) 1920 x 1097 and 3(h) 1920 x 1076, which are illustrated in the figure 3(e) to 3(h). Each of these images is about 20 – 28 lakh pixels. Also, Figures 3(i) through 3(k) are sourced from the pixabay.com website and they are all of the resolution 1920 x 1920 pixels, with approximately 36 lakh pixels included in each. Figures 3(l) through 3(p) are taken from the firefly dataset. Their sizes are (i) 2688 x 1536, (m) 2304 x 1792, (n) 2688 x 1536, (o) 2304 x 1792 and (p) 2304 x 1792. Each of these images is about 41 lakh pixels.

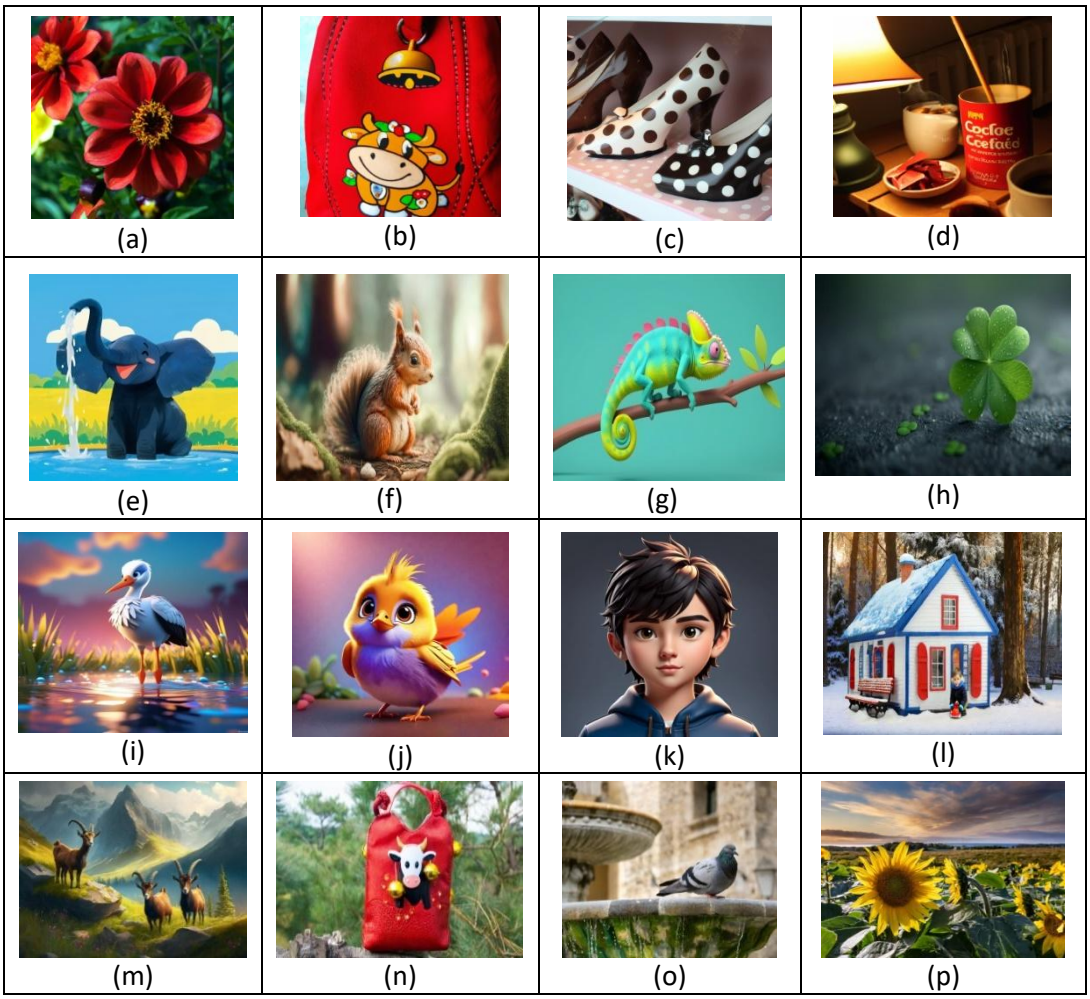

Figure 3. Sample AI Images

4.1 Performance Analysis by Simplification Rate

The reduction of data from high resolution AI image to low poly representation is analysed by simplification rate. First Level Simplification (FLS) involves the selection of feature points from the original AI image. Second Level Simplification (SLS) involves the selection of points using hexagonal grid. The cumulative line plot, figure 4, illustrates the relationship between the number of pixels originally present in the image and the degree of simplification that was accomplished. We can see that effective simplification is achieved while preserving discriminative visual characteristics. The number of pixels in the input AI images is in between 10 lakhs to 42 lakhs. The two-level simplification reduces the data from lakhs to a few thousand points.

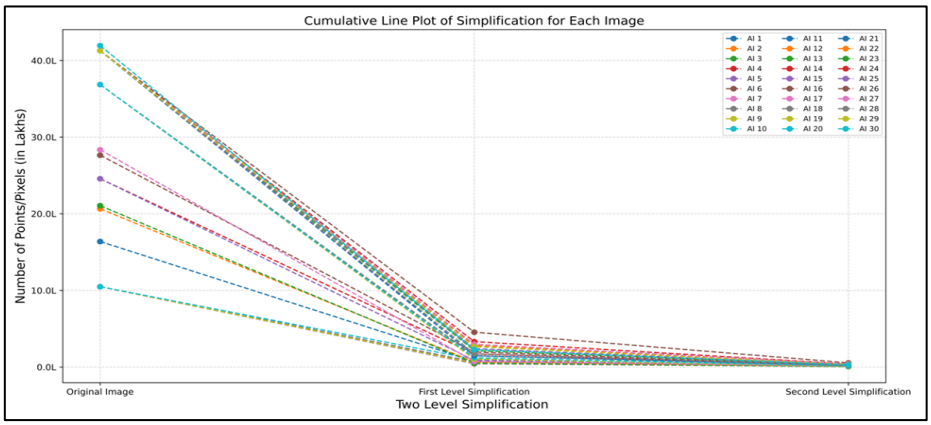

Figure 4. Cumulative Line Plot of Two-Level Simplifications

FLS rate is calculated as:

$$FLS\ Rate = \left(\frac{Number\ of\ points\ in\ FLS}{Number\ of\ pixels\ in\ AI\ image}\right) \times 100 \quad (1)$$

The SLS rate is calculated as

$$SLS\ Rate = \left(\frac{Number\ of\ points\ in\ SLS}{Number\ of\ pixels\ in\ AI\ image}\right) \times 100 \quad (2)$$

First Level Simplification (FLS) and Second Level Simplification (SLS) findings are shown together with the original AI image size and number of pixels. The FLS and SLS rates illustrate how successfully the simplification process strikes a balance between abstraction and visual correctness by displaying the percentage of visual information retained at each level in the two-level simplifications. Table 2 shows the FLS and SLS rates of the two-level simplifications.

Table 2. FLS and SLS Rate of Two-Level Simplifications

Image Id	AI Image Size	Number of Pixels in AI Image	First Level Simplification (FLS)	FLS Rate (%)	Second Level Simplification (SLS)	SLS Rate (%)
AI 1	1024 x 1024	1048576	63570	93.94	13970	98.67
AI 2	1024 x 1024	1048576	42911	95.91	8082	99.23
AI 3	1024 x 1024	1048576	63043	93.99	10811	98.97
AI 4	1024 x 1024	1048576	52065	95.03	8371	99.20
AI 5	1024 x 1024	1048576	42855	95.91	10067	99.04

AI 6	1024 x 1024	1048576	56123	94.65	11020	98.95
AI 7	1024 x 1024	1048576	61736	94.11	12898	98.77
AI 8	1024 x 1024	1048576	68033	93.51	12470	98.81
AI 9	1024 x 1024	1048576	48802	95.35	9996	99.05
AI 10	1024 x 1024	1048576	101157	90.35	15578	98.51
AI 11	1280 x1280	1638400	75886	95.37	13064	99.20
AI 12	1920 x 1076	2065920	67333	96.74	7525	99.64
AI 13	1920 x 1097	2106240	50877	97.58	5638	99.73
AI 14	1920 x 1280	2457600	170670	93.06	20481	99.17
AI 15	1920 x 1280	2457600	88774	96.39	10176	99.59
AI 16	1920 x 1440	2764800	196855	92.88	23115	99.16
AI 17	1477 x 1920	2835840	73126	97.42	12949	99.54
AI 18	1920 x 1920	3686400	146060	96.04	17416	99.53
AI 19	1920 x 1920	3686400	104289	97.17	11126	99.70
AI 20	1920 x 1920	3686400	120944	96.72	14950	99.59
AI 21	2688 x 1536	4128768	148205	96.41	22333	99.46
AI 22	2304 x 1792	4128768	282333	93.16	38407	99.07
AI 23	2304 x 1792	4128768	215439	94.78	31041	99.25
AI 24	1792 x 2304	4128768	331651	91.97	39634	99.04
AI 25	2304 x 1792	4128768	240601	94.17	32556	99.21
AI 26	2688 x 1536	4128768	454943	88.98	54485	98.68
AI 27	2304 x 1792	4128768	297305	92.80	39677	99.04
AI 28	1792 x 2304	4128768	174088	95.78	24726	99.40
AI 29	2688 x 1536	4128768	266825	93.54	33684	99.18
AI 30	2048 x2048	4194304	227475	94.58	30211	99.28
Average				94.61		99.19

The average FLS and SLS rates are 94.61% and 99.19%, respectively. This huge reduction achieved by carefully selecting feature points in FLS and using a hexagonal grid in SLS. From the small set of points, the proposed method successfully generates a low poly image. The two-level simplification keeps important featurepoints that ensure the low poly image maintains visual and geometric information from the input AI image.

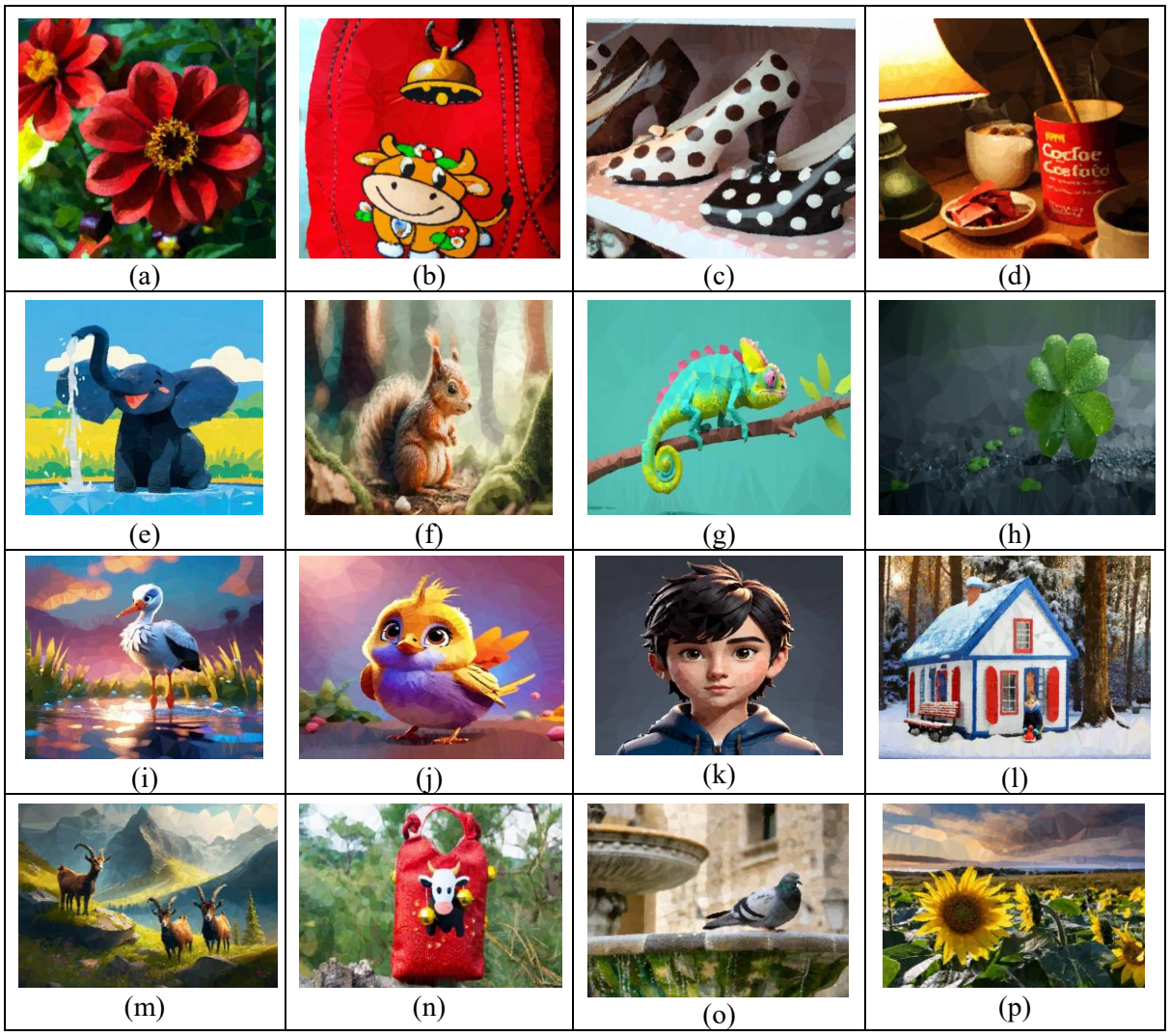

Figure 5. Low Poly Images from AI Images (Zoom in to Observe the Detailed Abstraction)

The first level of simplification reduces the pixel count greatly, and in the second level of simplification, it is further refined by selecting key spots using a hex grid that preserves the structure of the image. The Delaunay triangulation uses the seed points that correspond to the second level simplification points. The results of the proposed two-level simplification and the created low poly images are shown in Figure. We conclude that very few seed points are enough to create a low poly image.

We can see from Table 3, that the input AI images range in size from 3.0 MB to 12.0 MB, with a size of 7.55 MB. The memory usage of the resultant triangle representation falls between 0.52 MB and 3.34 MB, with an average size of 1.41 MB, the outcome of this method

leads to a decline in memory usage, with an average reduction of 79.96% and some images dropping by as much as 90.23%. In the worst case, the method still cuts memory use by 65.01%.

Table 3. Summary of Memory Usage Statistics

	AI Image Size (MB)	**Low Poly Representation (MB)**	**Memory Reduction (%)**
Min	3.0	0.52	65.01
Max	12.0	3.34	90.23
Average	7.55	1.41	79.96

The results show that the low-poly representation is a good way to simplify images because it saves a lot of memory while keeping important visual features.

4.2 Comparison of Approaches based on Region, Entropy, and Saliency Using Hexagonal Grid Sampling

The qualitative evaluation of region-based (Pro HRLP), entropy-based (Pro HELP), and saliency-based (Pro HSLP) approaches, as well as research methods region-based (RLP) [17] and Pic2PolyArt [14] for low-poly image generation is being compared. For the experimental study, we have taken second-level simplification points for Delaunay triangulation.

Figure 6 illustrates the AI-generated images and the resultant low-poly images of various techniques. The proposed hexagonal region-based low-poly (Pro HRLP) method generates a low-poly image that keeps the structure intact, clearly triangulates flat regions, and creates small triangles in edge regions, resulting in an output that is both balanced and well suited. The region based (RLP) method does not produce results as good as those of Pro HRLP because random point elimination in simplification breaks spatial coherence, resulting in huge, uneven triangles in smooth backdrop areas. This technique fails to preserve important structural elements, resulting in poorly defined edges and an uneven low-poly abstraction. In the entropy-based hexagonal grid sampling method (Pro HELP), the first level of simplification's entropy-based feature extraction method and the second level's hexagonal grid sampling method do not yield good results.

Figure 6. Low Poly Images from AI Images (Zoom in to Observe the Detailed Abstraction)

In a certain way, the hexagonal grid sampling preserve's structure while the entropy-based feature extraction yields ill-defined boundaries, creating tiny triangles along edges. All the details of smoothing and face regions get oversimplified; even some areas are not triangulated, meaning that the whole area looks immensely smooth with no definition. With Pro HSLP and Pic2PolyArt, one gets a low-poly image that is exceedingly simplified. The triangles produced are too tiny, taking out the visibility of abstraction and reducing low poly image abstraction. A detailed study of the proposed methods reveals that, for achieving low-poly abstraction of AI generated images, the hexagonal-grid region-based sampling paradigm

(Pro HRLP) consistently produces results which are optimal in terms of both efficiency and appearance. It is a technique that accurately triangulates smooth regions and preserves structural integrity and sharp edges and boundaries, thus producing a very well-balanced and high-quality low-poly result.

4.3 Quantitative Evaluation of Structure and Pixel-Level Accuracy

The proposed Hexagonal Region Low Poly (Pro HRLP), Hexagonal Entropy Low Poly (Pro HELP) and Hexagonal Saliency Low Poly (Pro HSLP), for generating low-poly image abstraction are compared to various recent techniques such as Region Low Poly (RLP) [17] and Pic2PolyArt [14].

SSIM (Structural Similarity Index Measure) assesses how effectively the fundamental structural elements of the original image—such as edges, textures, and shapes—are preserved in this abstraction. High SSIM: A low-poly image that, despite simplified features, successfully maintains important edges, shapes, and contrast. Low SSIM: A low-poly image that creates structural discrepancies by warping or misrepresenting key features.

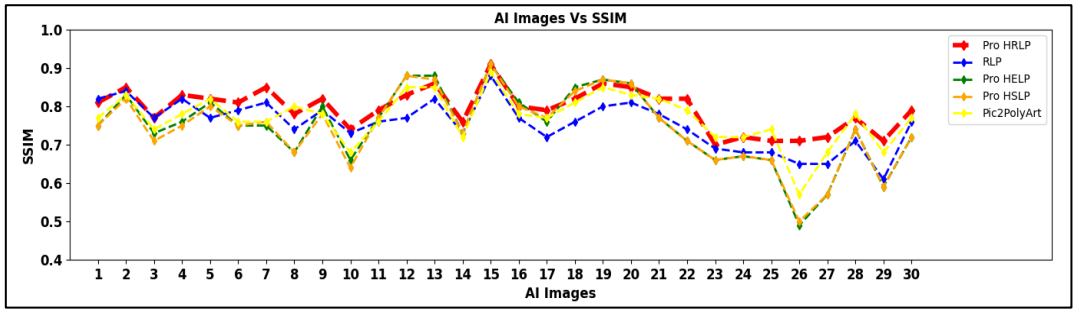

Figure 7. Graphical Representation of SSIM Measure.

Figure 7 shows the SSIM values with respect to 30 distinct numbers of AI images that are simplified into points indicated in the table column named “Second Level Simplification”. The SSIM measure shows that the proposed Pro HRLP yields higher SSIM scores compared to other algorithms. The technique maintains a high SSIM of about 0.80 when 99% of a high-resolution AI image of about 40 lakh pixels is reduced to an average of 35,000 points. A high SSIM score means that the suggested approach more fully maintains the structural details and general resemblance between the original and simplified images. The method is ideal for high-resolution images when structural fidelity is essential since it effectively strikes a compromise between size reduction and preservation of perceived quality

MSE (Mean Squared Error) calculates the average squared difference between the original image and the corresponding pixels of the low-poly image. This provides a numerical figure for pixel-wise deviation, which is helpful for monitoring how much the simplification changes the values of individual pixels.

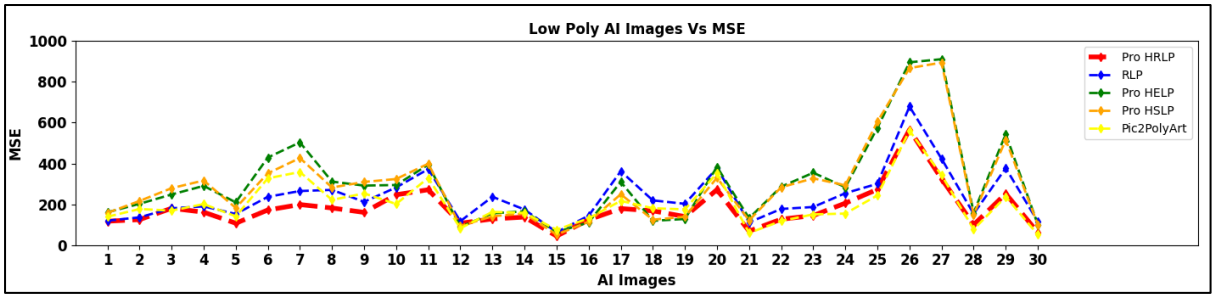

Figure 8. Graphical Representation of MSE Measure

Figure 8 shows that the proposed hexagonal region low poly (Pro HRLP) method outperforms RLP, Pro HELP, Pro HSLP and Pic2PolyArt by generating lower MSE values, which indicate superior accuracy in visual representation. Hexagonal grids over region-based points provide an organized framework for important seed point selection. The second-level simplification in each hex cell eliminates fewer important points, leaving the point set rather thin but clearer. This selective simplification, together with Delaunay triangulation, preserves edge continuity and detail in the final low-poly image while reducing the overall point count. It achieves much lower MSE values by significantly reducing point count while preserving information, thereby indicating high structural similarity to the original image and effective minimization of the error. Entropy maps highlight variations in intensity at the pixel level, often concealing noise or imperceptible textures, which may lessen the quality of point selection. Salience maps do emphasize visually salient areas; however, they may miss fainter edges and curves that are essential for an accurate structure in low-poly abstraction. This means that these methods can underrepresent other structurally significant areas while overrepresenting a number of points in entropy or saliency-intense areas.

Table 4. Average SSIM, MSE and Computational Time of Various Methods

Average	Pro HRLP	RLP	Pro HELP	Pro HSLP	Pic2PolyArt
SSIM	0.79	0.76	0.75	0.75	0.77
MSE	178.09	236.38	303.39	293.31	201.56
Time(s)	6.21	3.55	15.08	13.78	53.61

From Table 4, the mean squared error and structural similarity index measure reflect better results when using the low polygon method in hexagonal regions. In light of all the factors, the region-based approaches provide more reliable results, mainly for preserving small features and boundaries relevant to the image structure's integrity. Even if entropy- or saliency-based selections can highlight the most visually emphasized portions of an image, this is not the case with the region-based methods. This method is a real advantage for large images because seeds defined by regions pick out major contours to create low-polygon representations that finely detail structural elements. According to the framework, the average SSIM is 0.79 and the average MSE is 178.09, with 6 a computation time of 6.21 seconds,meaning that the structures are well preserved, with relatively low pixel-level error and efficient computation speed. The two-level system has quite good visual quality, with most edges and textures still recognizable despite significant vertex reduction. In comparison with single-stage methods, it can produce more consistent meshes with fewer clustering artifacts.

4.4 Robustness Evaluation Against Noise, Occlusion, and Compression Artifacts

In our robustness experiments, we assessed the proposed two-level simplification (Pro HRLP) under three perturbations: additive Gaussian noise ($\sigma = 0.5$), 2% random occlusion, and JPEG compression at quality level Q = 10, to evaluate how it responds to real-world aberrations.

Figure 9. Robustness Evaluation of the Two-Level Simplification Framework

Figure 9 (a) shows the original AI-generated image. Figure 9 (b) shows the low poly image of the proposed method, which attains SSIM = 0.77 and MSE = 40.67, establishing our benchmark. Under mild Gaussian noise, Figure 9 (c), SSIM remains 0.77 and MSE rises modestly to 48.10, indicating preserved structure despite slight pixel-level errors in flat regions. With 2% random masking, Figure 9 (d) shows that SSIM remains at 0.77 and MSE decreases to 40.27, as seed points shift to visible edges without compromising overall fidelity.In spite of JPEG Compression artifacts (Q = 10), Figure 9 (e) indicates that SSIM remains at 0.77 and

MSE is 42.68, demonstrating the method's robustness against compression distortions. The two-level simplification approach reliably maintains structural integrity and uniform mesh distribution despite noise, occlusion, and compression artifacts, showcasing its strong applicability to high-resolution AI-generated images. Despite its robustness, the two-level architecture may exhibit suboptimal performance on intricate or congested images, as static feature extraction could overlook complex patterns. It is sensitive to edge noise and misdetections, which may dislocate seeds and generate jagged artifacts.

5. Conclusion

In this paper, we introduced an innovative methodology for generating low-poly abstractions from AI-generated images. By using a two-level simplification mechanism, this method dramatically lowers resolution without sacrificing visual quality. Using hexagonal grid-based seed point selection, the suggested methodology investigates region-based, entropy-based, and saliency-based feature extraction. Following this, the Delaunay triangulation and average coloring methods were applied, successfully reducing image resolution while maintaining crucial visual elements. Compared to the entropy and saliency map-based techniques, the region-based feature extraction using a hexagonal grid-based seed point selection strategy produced lower Mean Squared Error values and higher Structural Similarity Index Measure values. This suggests that distortion has decreased and structural similarity has been preserved more successfully. According to the results, the approach is resilient and scalable, making it appropriate for a range of applications using AI-generated solutions and content. Potential avenues for further research include adaptive simplification techniques that adjust the level of abstraction in AI-generated graphics based on their content. By employing machine learning techniques to optimize triangulation and seed point selection, low-poly images can also be made more productive and visually appealing. The development of real-time low-poly generation frameworks is another fascinating area that may lead to interactive digital art, gaming, and virtual reality applications.

References

[1] [Online], Available: https://www.adobe.com/products/firefly.html

[2] [Online], Available: https://openai.com/index/dall-e-3/

[3] [Online], Available: https://www.canva.com/

[4] [Online], Available: https://www.meta.ai/

[5] [Online], Available: https://stablediffusion.com/

[6] Gai, Meng, and Guoping Wang. "Artistic low poly rendering for images." The visual computer 32 (2016): 491-500.

[7] Topal, Cihan, and Cuneyt Akinlar. "Edge drawing: a combined real-time edge and segment detector." Journal of Visual Communication and Image Representation 23, no. 6 (2012): 862-872.

[8] Zhang, Wenli, Shuangjiu Xiao, and Xin Shi. "Low-poly style image and video processing." In 2015 International Conference on Systems, Signals and Image Processing (IWSSIP), IEEE, 2015, 97-100.

[9] C. J. Qian and D. Dobkin, “Generating low-poly abstractions,” [Online], Available: https://cjqian.github.io/docs/tri_iw_paper.pdf

[10] Uasmith, Thitiwudh, Tantikorn Pukkaman, and Peeraya Sripian. "Low-poly image stylization." Journal for Geometry and Graphics 21, no. 1 (2017): 131-139.

[11] Ma, Yiting, Xuejin Chen, and Yu Bai. "An interactive system for low-poly illustration generation from images using adaptive thinning." In 2017 IEEE International Conference on Multimedia and Expo (ICME), IEEE, 2017, 1033-1038.

[12] Ng, Ruisheng, Lai-Kuan Wong, and John See. "Pic2geom: A fast rendering algorithm for low-poly geometric art." In Pacific Rim Conference on Multimedia, pp. 368-377. Cham: Springer International Publishing, 2017.

[13] Lawonn, Kai, and Tobias Günther. "Stylized image triangulation." In Computer graphics forum, vol. 38, no. 1, 2019, 221-234.

[14] Low, Pau-Ek, Lai-Kuan Wong, John See, and Ruisheng Ng. "Pic2PolyArt: Transforming a photograph into polygon-based geometric art." Signal Processing: Image Communication 91 (2021): 116090.

[15] Laske, Olivia, and Lori Ziegelmeier. "Image Triangulation Using the Sobel Operator for Vertex Selection." arXiv preprint arXiv:2408.16112 (2024).

[16] Joseph, Philumon, Binsu C. Kovoor, and Job Thomas. "Balancing Simplification and Detail Preservation in Low Poly Image Abstraction through Edge-Preserved Seed Point Generation." International Journal of Image, Graphics and Signal Processing (IJIGSP) 16, no. 2 (2024): 43-57.

[17] Joseph, Philumon, Binsu C. Kovoor, and Job Thomas. "A Delaunay Triangulation-Based Low-Poly Image Abstraction." Journal of Image and Graphics 12, no. 4 (2024).

[18] Dontmesswithtexas.org, “Don’t mess with Texas,” [Online], Available: https://www.dontmesswithtexas.org/.

[19] Superhotgame.com, “SUPERHOT,” [Online], Available: https://superhotgame.com/.

[20] Steampowered.com, “STEAM,” [Online], Available: https://store.steampowered.com/app/661740/Morphite/.

[21] Astroneer.space, “ASTRONEER,” [Online], Available: https://astroneer.space/.

[22] M. M. Keleşoğlu and D. GüleçÖzer, “A Study on Digital Low Poly Modeling Methods as an Abstraction Tool in Design Processes,” in Civil Engineering and Architecture, vol. 9, no.7,2021, 2570 – 2586.

[23] Inkscape.org, “Inkscape,” [Online], Available: https://inkscape.org/.

[24] Adobe.com, “Photo shop,” [Online], Available: https://www.adobe.com/products/photoshop. html.

[25] Mould and Rosin, “A Benchmark Set for Evaluating Image Stylization,” Expressive 2016, [Online], Available: https://gigl.scs.carleton.ca/benchmark.html

[26] Kaggle.com, “Natural Images,” [Online], Available: https://www.kaggle.com/code/navidrashik/object-detection-and-classification/data.

[27] C. Zeng et al, “RenderFormer: Transformer-based Neural Rendering of Triangle Meshes with Global Illumination,” ACM SIGGRAPH 2025.

[28] Q. Wang, Z. Wang, K. Genova, P. P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser. “Ibrnet: Learning multi-view image-based rendering”. In CVPR. 2021, 4690–4699.

[29] V. Sitzmann, S. Rezchikov, W. T. Freeman, J. B. Tenenbaum, and F. Durand. “Light field networks: neural scene representations with single-evaluation rendering.” In Proceedings of the 35th International Conference on Neural Information Processing Systems (NIPS '21). Curran Associates Inc., Red Hook, NY, USA, Article 1477, 2021, 19313–19325.

[30] C. Wu, H. Mailee, Z. Montazeri, and T. Ritschel, “Learning to Rasterize Differentiably”. Computer Graphics Forum. vol.43(4), 2024.

[31] T. M. Li, M. Lukač, M. Gharbi, and J. Ragan-Kelley, “Differentiable vector graphics rasterization for editing and learning”, ACM Transactions on Graphics (TOG), vol. 39(6), 2020, 1-15.

[32] H. Yuan, A. Bousseau, H. Pan, Q. Zhang, N. J. Mitra ,L. Changjian , “DiffCSG: Differentiable CSG via Rasterization”, SIGGRAPH Asia 2024 Conference Papers Article No.: 9, 2024, 1 – 10.

[33] S. Liu, W. Chen, T. Li and H. Li, "Soft Rasterizer: A Differentiable Renderer for Image-Based 3D Reasoning," IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, 7707-7716.

[34] Q. Bammey, "Synthbuster: Towards Detection of Diffusion Model Generated Images," IEEE Open Journal of Signal Processing, vol. 5, 2024, 1-9.

[35] Pixabay.com, [Online], Available: https://pixabay.com/