SLIC Superpixel Algorithm for Pixel Artwork

Abstract—Pixel art stands apart from other digital image manipulations, such as simple blur filters, due to its emphasis on deliberate design, sharpness, color constraints, and aesthetic intent. While a blur filter often washes details together, pixel art preserves the character of each carefully chosen block of color. This paper explores a method of generating stylized pixel art by leveraging the SLIC (Simple Linear Iterative Clustering) superpixel algorithm. By intelligently grouping pixels into meaningful clusters and then "printing" them as larger, more uniform blocks, the method yields a pixelated rendition that is more thoughtfully structured than a brute-force downscaling or smoothing operation would provide. Through detailed explanations and Python-based experimentation, this work demonstrates how SLIC can serve as a powerful stepping stone from raw imagery to a visually appealing, intentionally pixelated design.

1 Introduction

IMAGE Pixel art is more than just a nostalgic callback to early gaming consoles or a quirky art style. It relies on reducing complex imagery into small, square segments of uniform color, giving each pixel greater significance as a visual building block. Unlike simple filters that blur or distort images to a uniform softness, pixel art seeks to maintain distinct outlines and recognizable shapes. The final piece should feel handcrafted, where every block of color has a purpose.

While creating pixel art from scratch can be time-consuming, adapting an existing image into a pixelated form is still challenging. Naively shrinking an image often introduces unwanted artifacts or muddy transitions. We need a technique that not only reduces resolution but also groups colors and edges into coherent areas. This is where superpixels come into play. Superpixels group pixels that share similar characteristics—such as color intensity or texture patterns—into more manageable regions. The SLIC algorithm [11] stands out for its simplicity, efficiency, and strong adherence to natural image boundaries.

This paper will illustrate how to use the SLIC superpixel algorithm to segment an image and then transform each superpixel into a uniform block of color, achieving a stylized pixel art effect. We will discuss the underlying concepts, the methodological steps, parameter choices, and present experimental results that demonstrate how SLIC-based segmentation preserves essential features while providing a structured, pixelated rendition.

2 Background

Pixel art is a distinctive art form originating from constraints in older hardware, yet now celebrated for its aesthetic. Each pixel is like a tiny "tile" placed intentionally to convey shape, expression, and depth using a limited color palette. Unlike methods that simply reduce resolution or apply blur, pixel art demands careful arrangement of pixels so that even at low resolution, the image remains recognizable and visually appealing. Pixel art can also be seen as a form of minimalist painting where every pixel is chosen thoughtfully. Instead of representing every detail, it aims to capture the essence of forms and textures through simplified blocks of color. This results in images that are not only visually distinct but also resonate with a sense of craftsmanship and intention.

Figure 1 [14] shows an example of pixel art inspired by anime figures, capturing intricate facial expressions and clothing folds with only a handful of carefully placed colored squares.

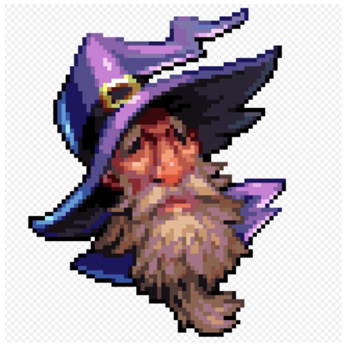

Fig. 1. Pixel Art of an Anime Figure

Superpixels have become a crucial concept in image processing and computer vision because they reduce the complexity of subsequent tasks. By replacing disorganized pixel grids with meaningful clusters, algorithms run faster and often produce more human-like interpretations of image content. Early approaches to superpixel generation focused on complex graph structures or spectral clustering, but SLIC provides a cleaner approach. It adapts a K-Means style algorithm to operate in a combined color and spatial space, using perceptually uniform LAB color coordinates to maintain visually meaningful distinctions.

In practice, SLIC starts by dividing an image into roughly even patches and placing initial cluster centers. It then refines these centers by considering both color similarity and location proximity. The result is a set of superpixels that contour objects and regions more naturally than uniform grids. Once these superpixels are defined, we can manipulate them as the building blocks of pixel art, "printing" each superpixel as a single uniform color block to form a stylized image that retains recognizable features while looking distinctly pixelated.

3 Related Work

Clustering-based approaches have been extensively explored for superpixel generation before the introduction of SLIC. These

• *Yuchen Song in the major of Applied Math/Computational Science track, Duke Kunshan University, Suzhou, China.*
E-mail: ys396@duke.edu

methods aim to group pixels into coherent regions based on similarity metrics, typically using iterative optimization techniques. For instance, Mean Shift (MS02) [8] applies a mode-seeking procedure to locate the local maxima of a density function in the feature space, resulting in irregularly shaped superpixels without direct control over their size or compactness. Another prominent method, Quick Shift (QS08) [9], uses a medoid shift strategy that clusters pixels based on Parzen density estimates. While QS08 achieves reasonable boundary adherence, it suffers from computational inefficiency. Additionally, the Watershed approach (WS91) [10] generates superpixels by simulating water flow from local minima, producing irregular and noncompact regions. Although these techniques laid the foundation for superpixel segmentation, they are generally limited by either irregular superpixel shapes or high computational costs, prompting the development of more efficient algorithms like SLIC.

There are many deep learning based methods facilitating pixel art generation these days. For example, Coutinho and Chaimowicz [5] introduced a conditional Generative Adversarial Network (GAN) that translates character images between poses, streamlining sprite sheet creation for game development. Similarly, Saravanan and Guzdial [6] developed the Pixel VQ-VAE model, which effectively learns discrete latent representations of pixel art, outperforming traditional models in embedding quality and performance on downstream tasks. Additionally, AI-powered tools like DeepAI's [7] Pixel Art Generator enable users to convert images into pixel art through advanced algorithms, democratizing the creation of pixel art for users without extensive artistic backgrounds.

4 METHODOLOGY

This section elucidates the methodology employed to realize pixel art through the Simple Linear Iterative Clustering (SLIC)[11] superpixel algorithm. We delve into the mathematical underpinnings of the SLIC algorithm, followed by the post-processing steps that transform the segmented superpixels into a pixelated artistic rendition. The parameterization involving step size, iterations, stride, and weight is also discussed in detail. Finally, pseudocode is presented to encapsulate the procedural flow of the implemented method.

4.1 SLIC Superpixel Segmentation

The SLIC algorithm partitions an image into compact and homogeneous superpixels by clustering pixels in a five-dimensional space composed of color and spatial coordinates. Some of the important parameter are : S controls the density of cluster centers, I affects convergence, s sets the pixelation level, and m balances the influence of spatial versus color distance. The mathematical formulation of the SLIC algorithm is as follows:

4.1.1 Color Space Transformation

To ensure perceptual uniformity, the input image is converted from the RGB color space to the CIELAB (LAB) [12] color space. The LAB space decouples luminance (L) from chromaticity (a and b), facilitating more effective color distance calculations.

4.1.2 Initialization of Cluster Centers

Given an input image I with dimensions $H \times W$, the desired number of superpixels K, and the compactness parameter m, the initial grid interval S is computed as: $S = \sqrt{\frac{N}{K}}$ where $N = H \times W$ is the total number of pixels. Cluster centers are then initialized uniformly across the image grid at intervals of S pixels. Each center c_k is represented as a five-dimensional vector $(L_k, a_k, b_k, x_k, y_k)$, where (x_k, y_k) denote the spatial coordinates.

4.1.3 Distance Metric

The distance D between a pixel p with LAB values (L_p, a_p, b_p) and spatial coordinates (x_p, y_p), and a cluster center c_k, is defined as: $D = \sqrt{\left(\frac{d_c}{m}\right)^2 + \left(\frac{d_s}{S}\right)^2}$, where:$d_c = \sqrt{(L_p - L_k)^2 + (a_p - a_k)^2 + (b_p - b_k)^2}$ and $d_s = \sqrt{(x_p - x_k)^2 + (y_p - y_k)^2}$

This composite distance metric balances color similarity (d_c) with spatial proximity (d_s), controlled by the compactness parameter m.

4.1.4 Iterative Clustering

The algorithm iteratively refines the cluster assignments through the following steps:(1)Assignment Step: For each cluster center c_k, pixels within a $2S \times 2S$ search window are evaluated. Each pixel p within this window is assigned to the cluster c_k that minimizes the distance D. (2)Update Step: After all pixels are assigned, the new cluster centers are recalculated as the mean LAB values and spatial coordinates of the pixels belonging to each cluster. This process is repeated for a predefined number of iterations I to ensure convergence.

4.1.5 Connectivity Enforcement

Post clustering, it is imperative to enforce connectivity to avoid fragmented superpixels. This is achieved by traversing the labeled image and ensuring that each superpixel forms a single connected component. Small disconnected regions are merged with adjacent superpixels based on proximity and similarity.

4.2 Post-Processing for Pixel Art

After enforcing connectivity to ensure that each superpixel remains a single, contiguous region, the algorithm proceeds to initialize the final pixelated image I_p by setting all its pixel values to zero. This blank canvas serves as the foundation for constructing the pixel art representation. The algorithm then iterates through each identified superpixel c_k. For each superpixel, it first calculates the centroid, which comprises the mean LAB color values (L_k, a_k, b_k) and the spatial coordinates (x_k, y_k) of all pixels within c_k. This centroid represents the average color and position of the superpixel, effectively summarizing its visual characteristics. Next, a representative color $C_k = (L_k, a_k, b_k)$ is selected based on the computed centroid, ensuring that each superpixel is uniformly colored. The algorithm then determines the block coordinates within the pixelated image I_p using a predefined stride s. This stride defines the size of each block, typically resulting in an $s \times s$ pixel area that will be uniformly colored. Finally, the algorithm fills each $s \times s$ block in I_p with the representative color C_k, effectively translating the clustered superpixels into a stylized pixel art format. This process results in an image that maintains the essential color regions and spatial structure of the original image while adopting a distinct, pixelated aesthetic.

The algorithm iteratively assigns each pixel to the nearest cluster based on a combined distance metric that balances color

Algorithm 1 SLIC-Based Pixel Art Generation

```
Require: Input image $I$, desired number of superpixels $K$, compactness $m$, number of iterations $I_{\max}$, stride $s$
Ensure: Pixelated image $I_p$
0: Convert $I$ from RGB to LAB color space
0: Compute grid interval: $S \leftarrow \sqrt{\frac{H \times W}{K}}$
0: Initialize cluster centers uniformly across the image grid
0: for each iteration from 1 to $I_{\max}$ do
0:   for each cluster center $c_k$ do
0:     Define search window $[x_k - 2S, x_k + 2S] \times [y_k - 2S, y_k + 2S]$
0:     for each pixel $p$ in the search window do
0:       $d_c = \sqrt{(L_p - L_k)^2 + (a_p - a_k)^2 + (b_p - b_k)^2}$
0:       $d_s = \sqrt{(x_p - x_k)^2 + (y_p - y_k)^2}$
0:       $D = \sqrt{\left(\frac{d_c}{m}\right)^2 + \left(\frac{d_s}{S}\right)^2}$
0:       if $D < D(p)$ then
0:         Assign pixel $p$ to cluster $c_k$
0:         Update $D(p) \leftarrow D$
0:       end if
0:     end for
0:   end for
0:   Recompute cluster centers as the mean LAB and spatial coordinates of assigned pixels
0: end for
0: Enforce connectivity to ensure spatial coherence of superpixels
0: Initialize pixelated image $I_p$ with zeros
0: for each superpixel $c_k$ do
0:   Compute centroid $(L_k, a_k, b_k, x_k, y_k)$ for $c_k$
0:   Select representative color $C_k = (L_k, a_k, b_k)$
0:   Define block coordinates based on stride $s$
0:   Fill the $s \times s$ block in $I_p$ with color $C_k$
0: end for=0
```

similarity and spatial proximity, controlled by the weight parameter m. Each iteration refines the cluster centers, ensuring better grouping of pixels into superpixels. After segmentation, a connectivity enforcement step ensures that each superpixel forms a single, contiguous region.

To generate the pixel art, each superpixel is replaced with a uniform color block determined by the cluster's average LAB color. The block size, defined by the stride s, determines the resolution of the pixelated output.

5 Experimental Results

6 Dataset

The dataset used in this study comprises a subset from Hugging Face, selected to eliminate the need for extensive training. Only a small subset of the dataset is utilized, as referenced in [13]. While qualitative analysis through visual inspection provides valuable insights, quantitative metrics such as Boundary Recall, Undersegmentation Error, and Computational Time offer a more objective evaluation. However, this report primarily focuses on qualitative assessments due to scope constraints.

7 Evaluation (Qualitative)

To evaluate the performance of the SLIC algorithm, experiments were conducted on various images with differing complexities and content. Each parameter's impact was meticulously analyzed to understand its influence on the segmentation results. The visual comparisons highlight how adjustments to parameters affect the granularity and accuracy of the segmentation, while the mathematical controls ensure that these adjustments are systematically managed to achieve optimal performance.

7.1 Results Overview

Figure 3 and Figrue 2 showcase the comparsion of original image, original image with boundaries after pixelation, and segmentation result.

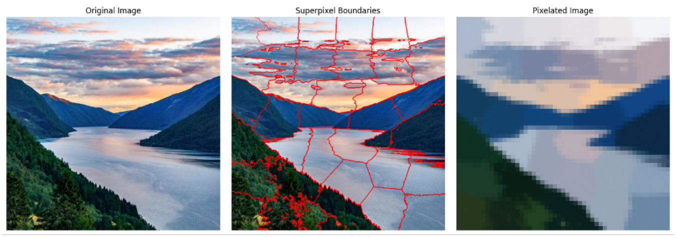

Fig. 2. Results of the original image with boundaries after pixelation

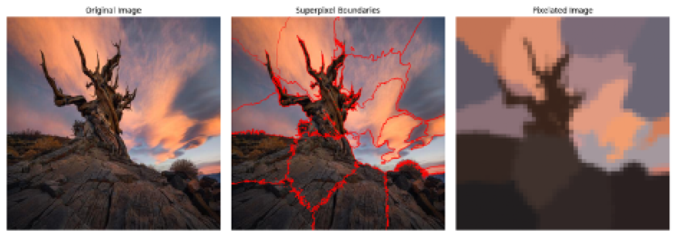

Fig. 3. Results of the original image with boundaries after pixelation

7.2 Parameter Impact Analysis

The SLIC algorithm's performance is significantly influenced by key parameters. This section examines the impact of pixel scale, compactness, and the number of superpixels, providing visual comparisons and discussing the underlying mathematical controls.

7.2.1 Number of Superpixels (k)

The number of superpixels, denoted by k, determines the segmentation granularity. As illustrated in Figure 4, increasing k results in finer segmentation, capturing more detailed structures within the image. Mathematically, k sets the target number of clusters, influencing the spatial distribution of superpixels. A smaller k leads to larger superpixels, suitable for images with less detail, while a larger k captures intricate details and edges, beneficial for tasks requiring precise boundary delineation. Visually, varying k alters the segmentation's appearance, with higher values providing more detailed and accurate representations of the image's regions. However, excessively high k may cause over-segmentation, complicating subsequent analysis.

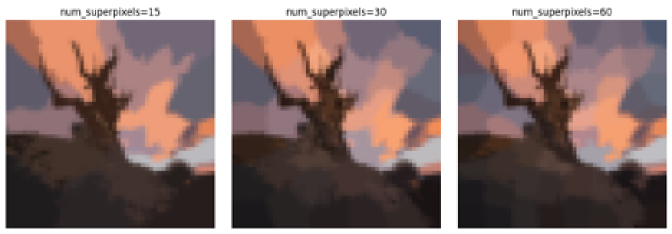

Fig. 4. Impact of varying numbers of superpixels

7.2.2 *Compactness Parameter* (m)

The compactness parameter, m, controls the balance between color similarity and spatial proximity. As shown in Figure 6, higher m values lead to more regular and compact superpixel shapes by emphasizing spatial distance in the clustering process. Mathematically, m scales the spatial component in the distance metric: $D = \sqrt{d_c^2 + \left(\frac{m}{S}\right)^2 d_s^2}$. where D is the distance measure, d_c is the color distance, d_s is the spatial distance, and S is the grid interval. A higher m ensures uniform superpixel sizes but may overlook important color boundaries, potentially merging distinct regions. Conversely, a lower m allows superpixels to adhere closely to color variations, enhancing boundary accuracy but resulting in irregular shapes. Balancing m is crucial for achieving segmentation that is both spatially coherent and color-consistent.

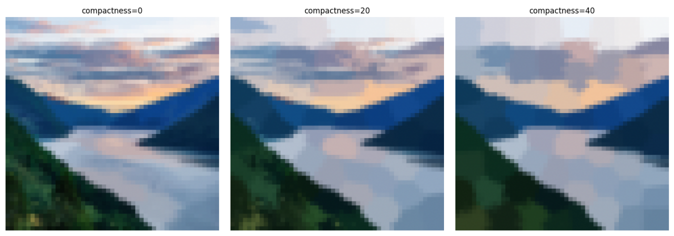

Fig. 5. Impact of different number of compactness

7.2.3 *Pixel Scale*

Pixel scale refers to the spatial resolution at which the image is processed, directly affecting the size of superpixels. Adjusting the pixel scale influences the algorithm's sensitivity to image details. A finer pixel scale (higher resolution) allows the SLIC algorithm to capture more precise boundaries and intricate structures, resulting in smaller and more numerous superpixels. Conversely, a coarser pixel scale reduces the number of superpixels, leading to larger regions that may merge similar areas. Mathematically, pixel scale determines the grid interval S, which sets the initial spacing between cluster centers. Proper control of pixel scale ensures that superpixels are appropriately sized relative to the image's features, balancing detail preservation with computational efficiency. Visually, varying the pixel scale affects the segmentation's fidelity, with finer scales providing detailed and accurate representations, while coarser scales offer a more generalized overview of the image.

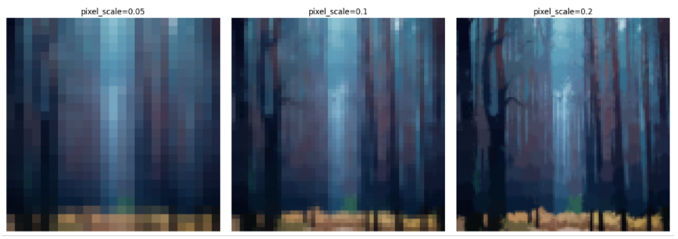

Fig. 6. Impact of different pixel scale values.

8 Conclusion and Future Work

This work explored how SLIC superpixels can be harnessed to produce pixel art, bridging the gap between raw imagery and intentional pixelation. By leveraging SLIC's ability to produce compact, visually meaningful segments, we can create stylized pixel art that retains important image features without resorting to naive downscaling methods.

Future work could integrate more sophisticated color quantization, incorporate adaptive stride sizes, or combine SLIC with learning-based methods. Evaluations involving user studies or perceptual metrics could also help refine the process.

References

[1] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," in *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[2] J. C. Brandt, T. Pock, and C. Rother, "Linear spectral clustering for superpixel segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8, 2011.

[3] J. Ren and J. Malik, "Semantic grouping with multiscale superpixels," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2743–2751, 2015.

[4] B. Dai, M. Ren, Y. Wu, X. M. Wang, L. Shao, and M. H. Yang, "Superpixels extracted via energy-driven sampling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2984–2991, 2012.

[5] F. Coutinho and L. Chaimowicz, "Generating pixel art character sprites using GANs," *arXiv preprint arXiv:2208.06413*, 2022.

[6] A. Saravanan and M. Guzdial, "Pixel VQ-VAEs for improved pixel art representation," *arXiv preprint arXiv:2203.12130*, 2022.

[7] DeepAI, "Pixel Art Generator," *DeepAI*, 2024. [Online]. Available: https://deepai.org/machine-learning-model/pixel-art-generator. [Accessed: 10-Dec-2024].

[8] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach toward Feature Space Analysis," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pp. 603-619, May 2002.

[9] A. Vedaldi and S. Soatto, "Quick Shift and Kernel Methods for Mode Seeking," Proc. European Conf. Computer Vision, 2008.

[10] L. Vincent and P. Soille, "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 13, no. 6, pp. 583-598, June 1991.

[11] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 11, pp. 2274-2282, Nov. 2012.

[12] "Colorimetry - Part 4: CIE 1976 Lab* Colour Space," International Commission on Illumination (CIE), Standard CIE S 014-4/E:2007, 2007.

[13] Qwertyforce, "Scenery Watermarks Dataset," Hugging Face, 2024. [Online]. Available: https://huggingface.co/datasets/qwertyforce/scenery_watermarks. [Accessed: 11-Dec-2024].

[14] "Pixel Art." Wikipedia. Last modified November 27, 2024. https://en.wikipedia.org/wiki/Pixel_art.