Greedy Region Based Adaptive Ordered Dithering for Limited Palette Image Rendering

Steven Tan - 13524060
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: steventan6002@gmail.com, 13524060@std.stei.itb.ac.id

Abstract- Limited-palette rendering can make smooth color changes look banded, especially in gradients, fog-like images, and paintings with soft tones. Ordered dithering reduces this problem by mixing palette colors in a repeated pattern, but one fixed Bayer matrix is not always suitable for every part of an image. This paper implements a greedy region-based adaptive ordered dithering method. The image is divided into fixed-size regions. For each region, the program tries four choices: no dithering, Bayer 2x2, Bayer 4x4, and Bayer 8x8. The choice with the lowest local score is used for that region. The score combines blurred MSE, which estimates local tone preservation, and a small pattern penalty. The method is tested on Starry Night and five synthetic images using palette sizes 4, 8, and 16. The results show that nearest-palette rendering often has better raw MSE, while dithering can give better blurred MSE and smoother visual tone. The adaptive greedy method is useful when different image regions need different dithering behavior, but it is slower than fixed baselines and does not guarantee a globally optimal image.

Keywords- greedy algorithm; ordered dithering; limited palette; image rendering; local score

I. INTRODUCTION

When an image is displayed using only a small number of colors, smooth color changes can turn into visible bands. This often happens in sky gradients, dark fog, or painted areas where many nearby tones are compressed into only a few palette colors. Limited palettes are still relevant in practice for pixel art, retro-style rendering, small previews, icons, low-resource displays, and experiments that intentionally restrict the available colors.

One common way to reduce banding is dithering. Dithering does not try to match every pixel exactly. Instead, it arranges available palette colors in a small pattern so that a group of pixels looks closer to the intended tone when viewed together. Ordered dithering is simple because it uses a repeated threshold matrix, such as a Bayer matrix. However, applying one Bayer matrix to the whole image can be too rigid. A strong pattern may help a smooth gradient, but the same pattern can add unwanted texture to a flat region.

This paper explores a simple adaptive version of ordered dithering. The image is split into fixed-size regions. For every region, the program tries four choices: no dithering, Bayer 2x2, Bayer 4x4, and Bayer 8x8. It then chooses the option with the best local score for that region. The score is only a practical rule, not a perfect model of human vision. Therefore, the contribution of this paper is an implementation and analysis of a local greedy strategy, not a claim that the method always produces the best possible image.

This point is important for IF2211 Strategi Algoritma. Although the image is divided into regions, the method is not divide and conquer. There is no recursive splitting and no final combine step that guarantees an optimal whole image. The central idea is greedy selection: each region immediately takes the best available option according to a local score.

The experiments compare the proposed adaptive greedy method with nearest-palette rendering and three fixed ordered-dithering baselines. The fixed Bayer baselines are needed because the proposed method is basically an adaptive ordered-dithering method. Without them, the paper would only show whether dithering beats no dithering, not whether local selection is useful compared with using one fixed Bayer matrix.

II. THEORETICAL BASIS

A. Digital Image and Limited Color Palette

A digital raster image can be seen as a grid of pixels. In an RGB image, each pixel stores red, green, and blue values. With 8-bit channels, each channel has values from 0 to 255, so a normal image can contain a very large number of possible colors.

A limited palette restricts the output to a fixed list of colors. For example, a palette of size 8 means every output pixel must be one of only eight selected colors. This restriction makes the rendering problem harder because many original colors must be represented by the same palette color or by a local mixture of palette colors.

B. Color Quantization

Color quantization reduces a full-color image into a smaller set of representative colors. In this experiment, the palette is produced using median-cut quantization, a common method that repeatedly divides the color space so that each final group can be represented by one color [2].

The simplest baseline is nearest-palette rendering. For each pixel, the program finds the closest palette color and replaces the pixel with that color. This method is fast and often gives low raw MSE because each pixel is individually mapped to the nearest available color. The weakness is that smooth tonal changes can become visible bands.

C. Dithering and Ordered Dithering

Dithering intentionally changes some pixel values so that a small neighborhood looks closer to the original tone. Because of this, a dithered image can have worse raw per-pixel error even when it looks smoother to the eye. This is why the experiment also uses blurred MSE and visual comparison, not only raw MSE.

Ordered dithering uses a repeated threshold matrix. In a Bayer matrix, each pixel position has a threshold value. Before quantization, the pixel is shifted slightly based on that threshold, then mapped to the nearest palette color. Small matrices such as 2x2 create coarse patterns, while larger matrices such as 8x8 spread the pattern over a wider area [3].

D. Greedy Algorithm

A greedy algorithm builds a solution through a sequence of local choices. In the IF2211 material, greedy is described as a method that produces one sequence of decisions, unlike

dynamic programming which considers more than one decision sequence. A greedy choice is attractive because it is simple and fast, but it still needs analysis because a locally good choice is not always globally optimal [1].

In this paper, the local choice is the dithering setting for one image region. After a region chooses a setting, the program does not go back and change it based on later regions. This makes the method greedy: the final image is formed from many local decisions.

The greedy object in this problem is not an edge, a job, or an item. It is an image region. The region grid provides the decision positions, the candidate settings provide the possible choices, and the local score acts as the greedy criterion.

The method does not claim optimality. A complete global search would need to try every possible combination of settings across all regions. With four settings and m regions, that would be 4^m combinations. For a 256x256 image with 32x32 regions, there are 64 regions, so the global search space is 4^64 combinations. The greedy method is used because this exhaustive search is not practical.

III. ALGORITHM DESIGN

A. Overview of the Proposed Algorithm

The proposed method receives an RGB image, a limited palette, a region size, and a list of dithering choices. The implementation uses 32x32 regions. For each region, the program renders four candidate blocks: no dithering, Bayer 2x2, Bayer 4x4, and Bayer 8x8. It scores all candidates and copies the best one into the output image.

The chosen setting for each region is also stored in a setting map. This map is useful because it shows where the algorithm actually used dithering. Dark flat regions, sprite-like blocks, and already well-represented palette regions can choose no dithering, while smooth gradients can choose a Bayer matrix.

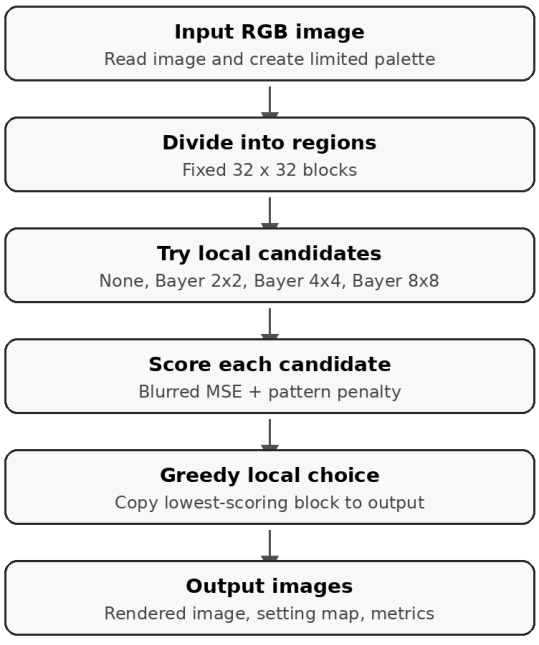

Fig. 1. Main processing flow of the proposed method.

B. Candidate Dithering Settings

The candidate set contains four choices. The first choice is no dithering, implemented as nearest-palette rendering. The other three choices use Bayer matrices of size 2x2, 4x4, and 8x8. All choices still end with nearest-palette mapping, so every output pixel remains inside the selected palette.

The number of choices is intentionally small. More choices could be added, such as different strengths, other matrices, or error-diffusion methods. However, a larger set would increase runtime and make the analysis less clear. For this paper, the goal is to test whether a simple local greedy selector can choose between no dithering and several ordered-dithering scales.

Method	Uses dithering	Region adaptive
Nearest palette	No	No
Uniform Bayer 2x2	Yes	No
Uniform Bayer 4x4	Yes	No
Uniform Bayer 8x8	Yes	No
Adaptive greedy	Optional	Yes

Table I. Compared rendering methods.

C. Local Scoring Function

Each candidate block is given a local score. The first part of the score is blurred MSE between the original block and the candidate block. Both blocks are blurred before comparison. This gives a simple estimate of local tone preservation: if a candidate keeps the local average close to the original, its blurred error should be small.

The second part is a pattern penalty. Dithering can improve local tone while also adding visible texture. To estimate this, the program compares the candidate block with a blurred version of itself. A large difference means the candidate contains stronger high-frequency pattern. The final score is blurred MSE plus a small weighted pattern penalty.

The score is local. It only compares one original region with one candidate result. It does not check neighboring regions or try to smooth the setting map globally. This keeps the method easy to understand and implement, but it also means that boundaries between regions can change suddenly.

The no-dithering choice is important. Without it, every region would be forced to use a Bayer pattern even when the nearest-palette result is already clean enough. Including no dithering turns the method into a selector rather than only a Bayer-matrix selector.

D. Greedy Selection and Complexity

The greedy step is simple. For one region, compute the score of every candidate and keep the candidate with the lowest score. The program does not compare all possible setting combinations for the whole image. It only decides the best local setting one region at a time.

A simple upper bound for the implementation is O(NKP), where N is the number of pixels, K is the number of candidate settings, and P is the palette size. Each candidate rendering maps pixels to the nearest palette color. In the experiment, K is fixed at 4 and P is only 4, 8, or 16, so the method is practical for the tested images even though it is slower than a fixed baseline.

```
function GreedyAdaptiveDithering(I: Image, C: Palette, r: Integer) → (O: Image,
M: SettingMap)
S ← {none, B2, B4, B8}
O ← empty image with the same size as I
M ← empty setting map

for each region B in I with region size r do
    best_score ← ∞
    best_setting ← none
    best_render ← empty region

    for each setting s in S do
        candidate ← RenderRegion(B, C, s)
        score ← LocalScore(B, candidate)

        if score < best_score then
            best_score ← score
            best_setting ← s
            best_render ← candidate

    CopyRegion(best_render, O, position of B)
    MarkSetting(M, position of B, best_setting)

return (O, M)
```

Algorithm 1. Greedy region-based adaptive ordered dithering.

IV. IMPLEMENTATION AND TESTING

A. Implementation Environment

The experiment was implemented in Python. NumPy is used for array operations, Pillow is used for image loading, saving, blurring, and median-cut palette generation, and pandas is used to save metric tables. The code is written in a direct style so that the algorithm in the paper can be traced to the implementation without many abstraction layers.

For reproducibility, the experiment runner generates outputs for each image, palette size, and method. The output folder stores rendered images, palette swatches, setting maps, comparison grids, and CSV files. A command block is not included in the paper because exact commands are more suitable for the README than for the main report.

The main parameters are kept fixed: region size 32 pixels, dithering strength 0.18, and pattern-penalty weight 0.02. Keeping these values fixed makes the comparison between methods easier to read. The experiment is therefore not a parameter search; it is a focused test of whether the local greedy selector gives sensible choices under one clear configuration.

The implementation handles edge regions in the same loop as normal regions. If the image size is not an exact multiple of the region size, the last block at the border is simply smaller. The same scoring rule is applied to every region.

B. Test Images

The input set contains six images: Starry_Night.jpg and five synthetic images. The synthetic images include a grayscale gradient, a color gradient, a dark fog-like gradient, a sprite-like block image, and a mixed-region image with flat areas and detailed noisy areas. These images were chosen to test different behavior: gradients test banding, flat blocks test unnecessary dithering, and mixed regions test whether different parts can choose different settings.

The full result folder contains all combinations of six images, three palette sizes, and five methods, giving 90 metric rows. To keep the paper readable, the visual discussion focuses on palette size 8. Palette sizes 4 and 16 are still included in the summary tables.

C. Compared Methods

Five methods are compared. The nearest-palette method is the no-dithering baseline. Uniform Bayer 2x2, 4x4, and 8x8 apply one ordered-dithering matrix to the whole image. Adaptive greedy chooses one of the four settings for each region.

The uniform Bayer baselines are important. Since the proposed method is an adaptive version of ordered dithering, the useful question is whether local selection improves on a single fixed Bayer matrix. If the paper only compared against nearest-palette rendering, it would not show whether the adaptive part is actually useful.

D. Evaluation Metrics

The evaluation uses raw MSE, blurred MSE, runtime, palette violation count, and region-setting percentages. Raw MSE measures per-pixel RGB difference from the original image. Blurred MSE applies a blur before computing MSE, so it better reflects local tone instead of exact pixel equality [4]. Runtime measures computational cost. Palette violation count checks whether output pixels remain inside the selected palette. Region percentages show how often the adaptive method chooses each setting.

These metrics should be read together. Low raw MSE does not automatically mean the best dithered result. Dithering deliberately changes pixels, so raw MSE can increase even when the image looks smoother. Blurred MSE and visual comparison are therefore used to complement raw MSE.

V. RESULTS AND ANALYSIS

A. Visual Result Comparison

The Starry Night comparison shows both the usefulness and the limitation of the adaptive method. Nearest-palette rendering keeps many pixels close to their nearest colors, but it also flattens some soft tonal transitions. Uniform Bayer methods add texture across the whole image. The adaptive method uses dithering only in selected regions, so the result does not cover the entire painting with one repeated pattern.

In this painting, the difference between Bayer 2x2, 4x4, and 8x8 is less obvious than in a synthetic gradient because the original image already contains many brush-like textures. This is why the setting map is useful: it shows which regions the greedy selector considered worth dithering.

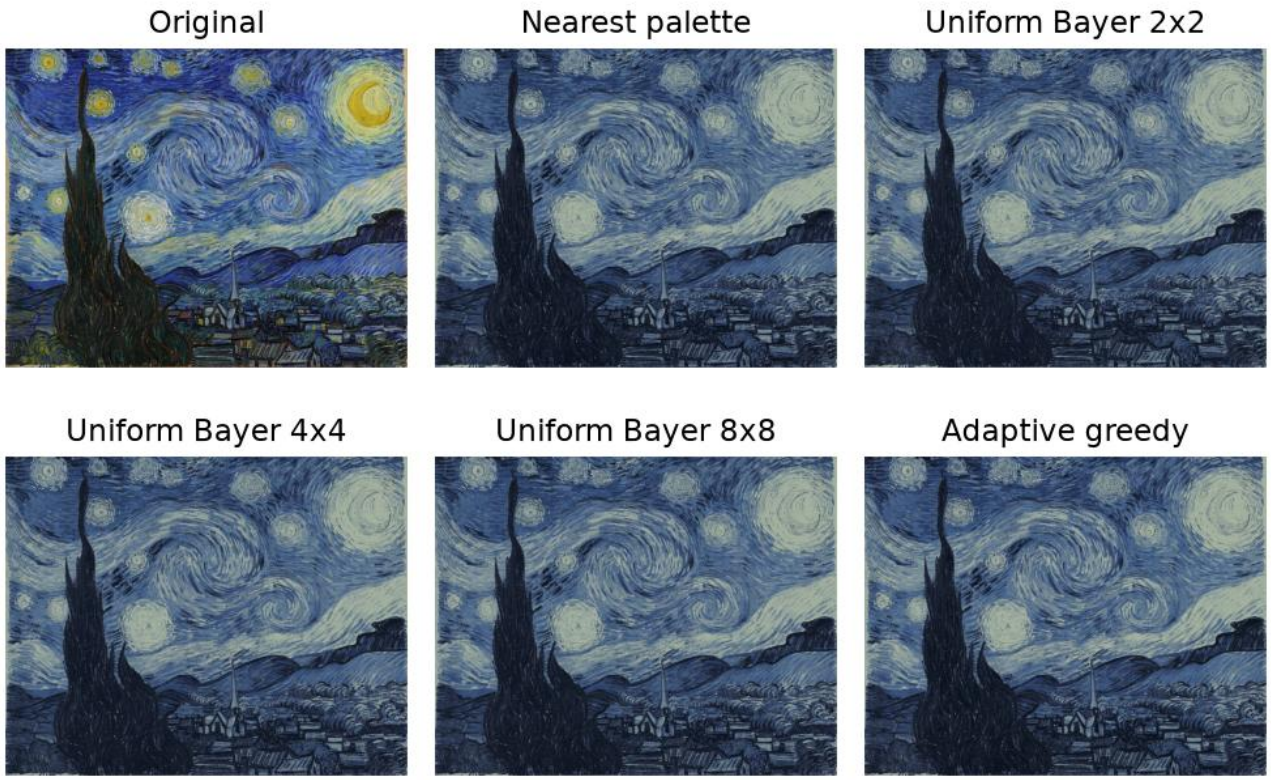

Fig. 2. Palette-8 comparison on Starry Night. This figure spans both columns because the six panels are unreadable at one-column width.

The grayscale gradient is a clearer synthetic case. Nearest-palette rendering follows the nearest color rule, but the result contains visible steps. Dithering produces a smoother local transition because adjacent pixels can use different palette colors. In this case, Bayer 4x4 and adaptive greedy give better local tone than nearest-palette rendering.

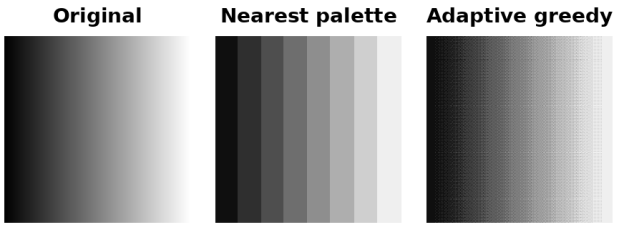

Fig. 3. Palette-8 grayscale gradient comparison.

The mixed-region image shows a different behavior. Large flat regions are often better without a visible dither pattern, while detailed or gradient-like regions may benefit from dithering. The adaptive method can choose no dithering for some regions and Bayer matrices for others. This behavior is the main reason for using a region-based greedy selector.

The sprite-block image, although not shown as a figure, is also useful. Its colors are mostly flat and close to the generated palette. A good adaptive method should avoid unnecessary texture in this case. The result confirms this behavior because most sprite-block regions choose no dithering.

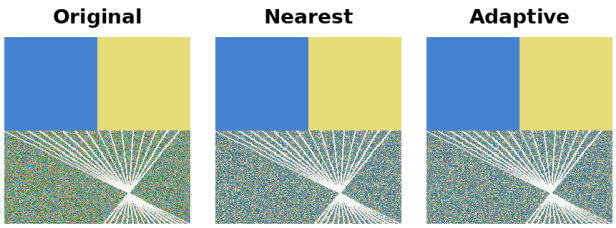

Fig. 4. Palette-8 mixed-region comparison.

B. Metric Analysis

Across the palette-8 results, adaptive greedy has the lowest average blurred MSE among the five methods, while nearest-palette rendering has the lowest average raw MSE. This matches the purpose of the method. Nearest-palette rendering is optimized for pixel closeness, while dithering is more concerned with local tone.

The selected metrics in Table II show the same pattern in individual images. On Starry Night, adaptive greedy slightly improves blurred MSE compared with nearest-palette rendering, but nearest still has lower raw MSE. On the grayscale gradient, nearest has much lower raw MSE, yet its blurred MSE is worse because the banding remains visible after local averaging. Adaptive greedy and Bayer 4x4 have much lower blurred MSE on that gradient.

Image	Method	MSE	Blurred MSE	Time (s)
Starry	nearest	318.9	181.3	0.226
Starry	adaptive	335.4	179.4	1.765
Gray grad.	nearest	85.5	69.3	0.013
Gray grad.	Bayer 4	230.4	16.6	0.013
Gray grad.	adaptive	231.1	16.4	0.162
Mixed	nearest	821.9	93.3	0.010
Mixed	adaptive	834.9	92.4	0.094
Color grad.	nearest	685.3	655.8	0.012
Color grad.	adaptive	709.4	553.9	0.178

Table II. Selected palette-8 metrics from the result CSV.

Table II also shows why the result should be discussed image by image. On the mixed-region image, adaptive greedy slightly improves blurred MSE over nearest, but the raw MSE is worse. On the color gradient, adaptive greedy again has worse raw MSE but much better blurred MSE. These cases show that the method is not simply minimizing pixel distance; it is trying to preserve local tone while controlling visible pattern.

The effect of palette size is summarized in Table III. The same general trend appears for palette sizes 4, 8, and 16. Nearest-palette rendering has lower average raw MSE, while adaptive greedy has lower average blurred MSE than nearest. This is expected because nearest-palette rendering is directly tied to per-pixel distance, while the adaptive score is based on blurred local tone.

Palette size 4 is the harshest condition because the palette is very small. No method can represent the original images well in that setting, so the trade-off between banding and patterning becomes stronger. Palette size 16 is easier because more original colors can be represented directly. The fact that the same disagreement between raw MSE and blurred MSE appears across palette sizes supports the use of multiple metrics.

Palette	Nearest MSE	Adaptive MSE	Nearest bMSE	Best uniform bMSE	Adaptive bMSE
4	760.6	797.5	492.8	435.8 (Bayer 8)	428.8
8	320.2	353.4	167.3	145.6 (Bayer 2)	141.0
16	197.2	226.3	78.4	66.7 (Bayer 2)	62.4

Table III. Average raw and blurred MSE across all six images.

Runtime is the main cost of the adaptive method. A fixed Bayer method renders the image once, while the adaptive method renders and scores four candidates for every region. For palette size 8, the recorded average runtime of adaptive greedy is about 0.410 seconds, while the fixed baselines are around 0.048 to 0.052 seconds. The exact number depends on the machine, but the relative slowdown is expected.

Palette violation count is 0 in every row of the summary CSV. This means the final mapping works correctly: even after dithering shifts pixel values before quantization, the saved output still uses only colors from the selected palette.

C. Setting Map Analysis

The setting map helps explain the greedy choices. In the Starry Night palette-8 result, most regions choose no dithering, while smaller areas choose Bayer settings. This means the algorithm does not force a visible pattern onto areas where nearest-palette rendering is already good enough.

The synthetic images make the behavior easier to see. The grayscale gradient does not choose the no-dithering setting and mostly uses Bayer 4x4, because smooth gradients benefit from local tone mixing. The dark fog gradient chooses no dithering everywhere, showing that the score can reject dithering when it would mainly add noise. The sprite-block image also chooses no dithering for almost all regions, which is desirable because crisp flat shapes should not be filled with unnecessary texture.

Image	None	B2	B4	B8
Starry	88.8%	6.2%	3.3%	1.6%
Gray grad.	0.0%	12.5%	62.5%	25.0%
Dark fog	100.0%	0.0%	0.0%	0.0%
Mixed	87.5%	9.4%	1.6%	1.6%
Sprite	98.4%	1.6%	0.0%	0.0%

Table IV. Palette-8 adaptive setting distribution.

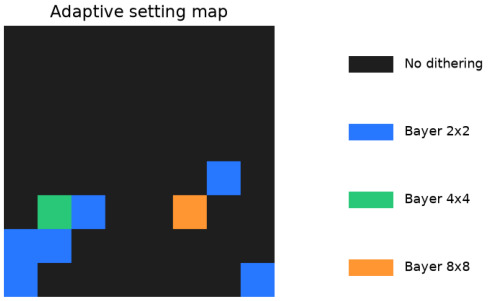

Fig. 5. Palette-8 setting map for the mixed-region image.

The Starry Night setting map gives a useful contrast to the mixed synthetic image. The map is dominated by no-dithering regions, with smaller pockets of Bayer choices. This does not mean the adaptive method failed to use dithering. It means the local score only selects dithering where the tone benefit is large enough to justify the added pattern.

For this greedy strategy, the setting map is also useful for checking the algorithm. It makes the local decisions visible. If a future version changes the score, region size, or candidate list, the setting map can show whether the change improves behavior or only changes a numeric metric.

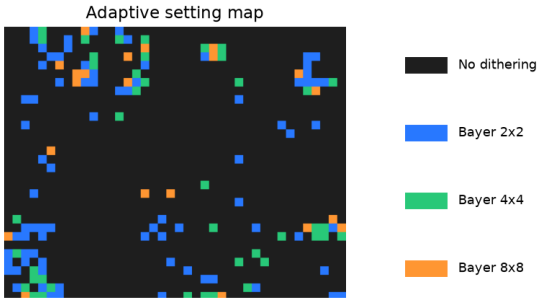

Fig. 6. Palette-8 setting map for Starry Night.

D. Strengths and Limitations

The main strength of the proposed method is practical adaptivity. It keeps ordered dithering simple, but it does not require one matrix size for the entire image. It can behave like nearest-palette rendering in flat regions, like Bayer dithering in smooth gradients, and like a mixture of both in images with varied local structure.

The first limitation is that the method is greedy and local. It does not guarantee a globally optimal output, and it does not coordinate pattern choices across neighboring regions. A region boundary may therefore separate two different Bayer settings. In some images this is not noticeable, but in others it may create block-like transitions.

The second limitation is the scoring function. Blurred MSE and the pattern penalty are only approximations. A human observer may prefer an output with a slightly worse score if the pattern looks more pleasant. The third limitation is the fixed region size. A 32x32 region is simple and works for the tested images, but real images contain structures at many different scales.

The fourth limitation is runtime. Uniform Bayer methods render the image once. The adaptive method renders every region several times and computes extra scores. Future work could test adaptive region sizes, edge-aware regions, better perceptual metrics, or a smoothing step for the setting map.

The experiment also uses a small candidate set. This is reasonable for an IF2211 paper because it keeps the greedy choice clear, but it does not explore all dithering possibilities. More strengths or matrix types could improve results, but they would also increase runtime and make the decision process harder to analyze.

Finally, the evaluation is not a user study. The visual analysis is based on selected images and two MSE variants. Human preference could differ from blurred MSE when two outputs have similar tone but different texture. Therefore, the conclusion should be read practically: the method is a useful adaptive heuristic, not a universal replacement for all dithering methods.

For practical use, region size is one of the most important parameters. A smaller region would react more quickly to local image structure, but it would create more region boundaries. A larger region would reduce boundary changes, but it might force one setting to cover both flat and gradient-like content. The 32x32 value used here is a simple middle point, not a proven best value.

Dither strength and pattern-penalty weight also affect the result. Stronger dithering can approximate intermediate tones better, but it makes the matrix pattern more visible. A larger pattern penalty makes the method more conservative and more likely to choose no dithering.

The setting map can help debug the method. If almost every region chooses no dithering on a smooth gradient, the score is probably too conservative. If almost every region chooses Bayer 8x8 on a flat sprite image, the score is probably too aggressive. This is useful because the final image alone does not always show why a decision was made.

The zero palette-violation result is also important. Ordered dithering is applied before the final nearest-palette step, so it does not create arbitrary RGB values in the saved image. The output can still be used in a truly limited-palette pipeline where every pixel must belong to the selected palette.

The method could be optimized further. The current Python implementation is intentionally direct and easy to follow. A faster version could precompute tiled Bayer thresholds, vectorize candidate scoring, or process regions in parallel because the decisions are independent after the palette is fixed.

VI. CONCLUSION

This paper presented a greedy region-based adaptive ordered dithering method for limited-palette image rendering. The image is divided into fixed-size regions, and each region chooses among no dithering, Bayer 2x2, Bayer 4x4, and Bayer 8x8 using a local score. The method is greedy because each region commits to the locally best candidate without searching all global combinations.

The experiments show that adaptive greedy is not always best in every metric. Nearest-palette rendering often has lower raw MSE because it maps every pixel to the closest palette color. Dithering can increase raw MSE because it deliberately changes pixels to improve local tone. Blurred MSE, visual comparison, and setting maps therefore give a more useful picture of the trade-off. Adaptive greedy can improve local tone in several cases and avoid unnecessary dithering in others, but it is slower and remains a heuristic. Future work can explore better perceptual scoring, adaptive region sizes, and smoother setting maps.

ATTACHMENT

Github: https://github.com/StevenT-1/Tugas-Makalah-IF2211-Strategi-Algoritma-2026

ACKNOWLEDGMENT

First of all, the author would like to thank God Almighty for His blessings, guidance, and strength throughout the completion of this paper. The author would also like to thank Pak Rila Mandala for the knowledge and guidance given during the IF2211 Strategi Algoritma course. The author also thanks Pak Rinaldi Munir for providing the course materials used as an important reference in writing the algorithmic discussion in this paper.

REFERENCES

[1] R. Munir, "Algoritma Greedy (Bagian 1)," Bahan Kuliah IF2211 Strategi Algoritma, Program Studi Teknik Informatika, Institut Teknologi Bandung, 2026.

[2] P. Heckbert, "Color image quantization for frame buffer display," Computer Graphics, vol. 16, no. 3, pp. 297-307, 1982.

[3] B. E. Bayer, "An optimum method for two-level rendition of continuous-tone pictures," in Proc. IEEE International Conference on Communications, 1973, pp. 11-15.

[4] R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th ed. New York: Pearson, 2018.

STATEMENT

I hereby declare that this paper is my own writing, not an adaptation or translation of another person's paper, and not plagiarism.

Bandung, 19 June 2026

Steven Tan - 13524060