8-Connectivity Grid Graph Modeling for Lightroom-Inspired Color-Space Mask Segmentation

Christabelcyne Costan - 13525141
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: costanbelcyne@gmail.com, 13525141@std.stei.itb.ac.id

***Abstract*—A photograph looks like reality. It is not. Underneath the image lies a rigid, unforgiving matrix of integers, and any software that claims to make a "perfect selection" is, in the end, just drawing a very precise mathematical boundary around a set of them. This paper deconstructs the selective color masking found in Adobe Lightroom through the formal framework of graph theory. An M × N image is modeled as a weighted, undirected 8-connectivity grid graph G = (V, E), where each pixel is a vertex, and each spatial-chromatic relationship is a weighted edge. The formal definitions of graph adjacency, path, and connected component are derived and mapped directly to their computational roles in the masking pipeline. The Euclidean distance in RGB space is used to quantify edge weights, and Breadth-First Search drives the traversal. A threshold sensitivity analysis shows the direct, near-monotonic relationship between tolerance parameter τ and the size of the resulting connected component C. Experimental Python implementations confirm that 8-connectivity captures the diagonal boundaries of organic subjects far more faithfully than 4-connectivity, at 13,530 additional vertices and 12.6 ms, proving that computational precision is not a luxury. It is a mathematical requirement.**

***Keywords*—Graph Theory, 8-Connectivity, Grid Graph, Connected Components, Color-Space Segmentation, BFS Traversal, Digital Image Processing.**

I. Introduction

A photograph is inherently deceptive. It presents itself as a continuous, unbroken window into a moment, but strip away the rendering and what remains is a finite, perfectly rigid matrix of integers. Every pixel is locked into its coordinates. Every color is a set of three numbers between 0 and 255. There is no gradient that is not, at the computational level, a sequence of discrete steps.

This matters because selective editing, the ability to adjust a specific range of colors without disturbing the rest of the image, is one of the most fundamental operations in professional photography. Adobe Lightroom's masking tools make this look effortless. A creator drags a slider, and the sky is selected. The foreground is untouched. It feels intuitive, almost artistic.

It is not intuition. It is graph traversal.

To replicate this computationally, the image cannot be treated as a continuous surface. It must be abstracted into what it actually is: a discrete network of pixels, bound to their neighbors by spatial and chromatic relationships. Graph theory provides the exact architecture for this. By modeling the image as an 8-connectivity grid graph, the subjective act of "selecting a color range" becomes an objective problem of connected component discovery, a problem with a clean mathematical definition, a clear algorithm, and measurable results.

This paper builds that model from first principles. It defines the graph, proves why 8-connectivity is not just preferable but necessary, implements the traversal, and measures the cost of getting it right.

Original RGB Image *Selectively Masked Output*

Fig. 1. Visual comparison of the original asset and its computed binary mask (Source: Author)

II. Theoretical Foundation

A. The Anatomy of a Pixel Graph

In discrete mathematics, a graph is a formal structure defined as:

$$G = (V, E)$$

where V is a non-empty set of vertices and E is a set of edges connecting a pairs of vertices. Applied to a digital image, V is the set of every pixel in the $M \times N$ matrix. A single pixel at row x and column y becomes a vertex $v_{xy} \in V$, where $0 \leq x < M$ and $0 \leq y < N$. The edge set E is the formal acknowledgment that no pixel exists in isolation, it is defined entirely by its relationship to the pixels around it.

The degree of a vertex v, written $d(v)$, is the count of edges incident to it. In the interior of an 8-connected grid, every vertex has $d(v) = 8$. At the boundary of the image, this drops: corner pixels have $d(v) = 3$, and edge pixels have $d(v) = 5$. By the Handshake Lemma, the sum of all vertex degrees equals twice the number of edges:

$$\sum_{v \in V} \deg(v) = 2|E|$$

For an $M \times N$ image under 8-connectivity, this resolves approximately $|E| \approx 4 \cdot M \cdot N$. The graph scales linearly with the image. A clean, predictable relationship that makes algorithmic complexity analysis straightforward.

B. Formalization of Pixel Adjacency and Paths

For a pixel p at coordinates (x, y), its four orthogonal neighbors form the set $N_4(p)$, and its four diagonal neighbors are form $N_D(p)$. The 8-connectivity neighborhood is the union of these two sets:

$$N_8(p) = N_4(p) \cup N_D(p)$$

A path of length n from pixel p to pixel q is a sequence of distinct vertices $(v_0, v_1, \ldots, v_n)$ such that $v_0 = p$, $v_n = q$, and each vertex v_i is adjacent to v_{i-1} for all $1 \leq i \leq n$. If the traversal algorithm cannot build a continuous path between two pixels without crossing an edge that violates the chromatic threshold, those pixels are definitively separated. There is no partial selection. No ambiguity. The graph either connects them, or it does not.

C. Graph Partitioning and Connected Components

The objective of color masking is ultimately binary: each pixel is either included in the mask or excluded from it. Within graph theory, this behavior can be represented using the concept of a connected component.

Formally, a subset $C \subseteq V$ is called a connected component of G if it satisfies two conditions:

- For any two vertices u, $v \in C$, there exists a path from u to v entirely within C.
- C is maximal, meaning no vertex outside C is adjacent to any vertex inside C).

In the context of this study, the generated mask corresponds to one connected component identified from the graph representation of the image. Starting from a selected seed pixel, traversal expands only through neighboring pixels that satisfy the chromatic similarity constraint.

This interpretation provides a useful way to connect image segmentation with graph theory. Rather than treating masking as a visual operation alone, the selected region can be viewed as a connected subset of vertices separated from the remaining image.

Generating a binary mask therefore becomes equivalent to identifying a target connected component C and distinguishing it from the remaining vertices $V \setminus C$.

D. The Problem with 4-Connectivity

A common alternative to the proposed model is 4-connectivity, where each pixel is connected only to its orthogonal neighbors: up, down, left, and right.

This formulation is computationally simpler and is often sufficient for applications with strongly axis-aligned structures. However, for natural images containing curved edges and diagonal transitions, restricting adjacency to four directions can interrupt continuity between visually connected regions.

In the 8-connectivity model, an edge e ∈ E exists between vertices $v_{x,y}$ and $v_{i,j}$ whenever:

$$\max(|x - i|, |y - j|) \leq 1$$

where $(x, y) \neq (i, j)$. This definition includes both orthogonal and diagonal relationships and allows the graph to better preserve local continuity.

The difference becomes more visible during segmentation. Since masking expands through adjacent pixels, excluding diagonal neighbors may produce fragmented boundaries or reduce coverage near curved structures. Section IV evaluates this effect experimentally through quantitative comparison.

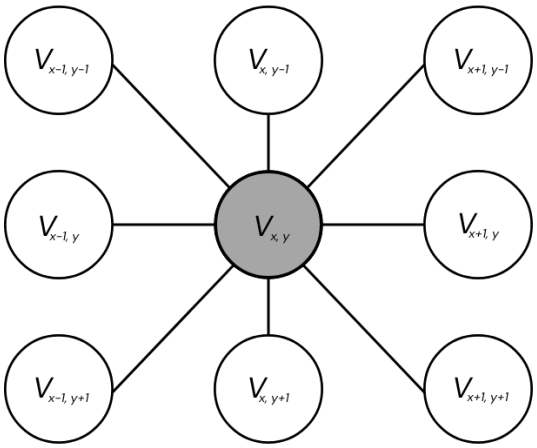

Fig. 2. A 3x3 spatial matrix illustrating an 8-connectivity grid graph (Source: Author, adapted from R. C. Gonzalez and R. E. Woods)

III. Methodology & Edge Weighting

A. Chromatic Distance Calculation

Every edge in G_{pixel} carries a weight that quantifies the chromatic distance between its two vertices. The RGB color space is used here for its direct alignment with how image arrays are stored in memory. For an edge $e \in E$ connecting vertex u and vertex v, the weight $W(e)$ is the Euclidean distance between their color vectors:

$$W(e) = \sqrt{(R_u - R_v)^2 + (G_u - G_v)^2 + (B_u - B_v)^2}$$

A low weight means chromatic harmony. The two pixels are nearly the same color. A high weight means a boundary. The Euclidean (L_2) norm is chosen deliberately: it is isotropic within the 3D RGB cube, meaning a chromatic shift that is spread across all three channels is penalized proportionally, the same as a shift concentrated in one. This matches the perceptual reality of color gradients, where natural transitions involve all channels simultaneously.

The current formulation is a pure color-space metric. In complex visual assets, chromatically identical pixels can exist in spatially disjoint regions, a blue sky and a blue window reflection, for instance. To prevent BFS from leaking across these disconnected regions, the edge weight can be theoretically extended into a spatio-chromatic formulation by introducing a spatial decay term:

$$W(e) = \alpha \|C_u - C_v\|_2 + \beta\sqrt{(x_v - x_{seed})^2 + (y_v - y_{seed})^2}$$

Here, α governs chromatic tolerance and β governs spatial decay from the origin seed. The current implementation sets $\beta = 0$ to isolate pure color-space segmentation, but acknowledging this extension aligns the model more closely with proprietary masking algorithms used in commercial editing suites.

B. Graph Segmentation Constraints

The mask is built by a traversal that begins at a user-defined seed pixel and expands outward. At each step, the algorithm evaluates whether the edge weight $W(e)$ satisfies the threshold condition $W(e) \leq \tau$. If it does, the neighbor is added to the mask. If it does not, the boundary holds. The result is the connected component C reachable from the seed within the chromatic tolerance τ.

C. Computational Complexity

BFS on G_{pixel} has a worst-case time complexity of $O(|V| + |E|)$. In a 4-connected grid, $|E| \leq 4|V|$. In the 8-connected grid, $|E| \leq 8|V|$. Both are linear in $|V|$, so the asymptotic class is the same, but the constant factor doubles. This is the explicit cost of diagonal continuity, and it is quantified in Section IV-D.

Space complexity is dominated by two structures: the $M \times N$ binary mask matrix at $O(|V|)$, and the BFS queue. The queue stores only the active frontier of the expanding region. For a roughly circular connected component of area A, the perimeter scales as $O\left(\sqrt{|V|}\right)$, which means the queue remains far smaller than the matrix itself. The algorithm is memory-safe on multi-megapixel images.

D. Why BFS, Not DFS

The choice of BFS over DFS is not arbitrary. It is a practical necessity on large grid graphs. DFS is recursive by nature. On a 1000×1000 pixel matrix, a worst-case connected component spanning 10^6 vertices would require a recursion depth approaching $O(|V|)$, far beyond Python's default stack limit. The result is a RecursionError long before the mask completes.

BFS expands radially. It stores only the frontier vertices in its queue at any point, keeping memory overhead at approximately $O\left(\sqrt{|V|}\right)$ for typical circular or organic shapes. The mask completes without touching the stack. For high-resolution grid graphs, this is not a preference. It is the only architecturally sound option.

E. Metric Space Alternatives

The Manhattan distance (L_1 norm) offers a computationally cheaper alternative to the Euclidean distance, it replaces the floating-point square root with a simple absolute sum:

$$W_{L1}(e) = |R_u - R_v| + |G_u - G_v| + |B_u - B_v|$$

The cost it saves is real. The accuracy it sacrifices is also real. The L_1 norm is geometrically anisotropic: it penalizes simultaneous multi-channel shifts more heavily than single-channel ones, which distorts the shape of the chromatic neighborhood in the RGB cube. For natural images, where organic color transitions almost always involve all three channels at once, the L_1 norm consistently over-shrinks the mask boundary in the wrong places. The L_2 norm's extra computation is a deliberate, necessary trade-off for isotropic accuracy.

F. Adaptive Threshold Formulation

The static threshold τ has a quiet weakness: lighting. A fixed tolerance that correctly segments the bright face of a saxophone bell will fail in the shadowed region of the same bell, because absolute RGB values compress as luminance falls. The colors that were 40 units apart in full light are now 15 units apart in shadow, and the same $\tau = 40$ boundary either overcaptures or abandons entire regions depending on where the light falls.

An adaptive threshold τ_{local} addresses this by scaling with the local texture of the seed's neighborhood. By computing the standard deviation σ of pixel values in a 5×5 region around the seed, the threshold adjusts proportionally:

$$\tau_{local} = \tau_{base} \times (1 + \lambda\sigma)$$

where λ is a tuning coefficient for variance sensitivity. In smooth gradient regions, σ is low and τ_{local} contracts, keeping the boundary tight. In textured regions, σ rises and τ_{local} expands to absorb the natural variation. This is what Lightroom's "Smoothness" slider actually models, and this is the mathematical form it takes.

IV. Implementation & Analysis

A. Algorithmic Traversal and Execution

The theoretical framework was implemented as a BFS traversal in Python. Algorithm 1 defines the complete logic of the 8-connectivity mask generation.

Algorithm 1: 8-Connectivity Color Masking

```
procedure 8 − CONN − MASK(Image, SeedX, SeedY, τ)
  for all (x, y) ∈ Image do
    Mask[x, y] ← 0
  end for
  Q ← ∅
  ENQUEUE(Q, (SeedX, SeedY))
  Mask[SeedX, SeedY] ← 1
  C_seed ← Image[SeedX, SeedY]
  Δ ← (−1,−1), (−1,0), (−1,1), (0,−1), (0,1), (1,−1), (1,0), (1,1)
  while Q ≠ ∅ do
    (cx, cy) ← DEQUEUE(Q)
    for all (dx, dy) ∈ Δ do
      nx ← cx + dx
      ny ← cy + dy
      if BOUNDS − CHECK(nx, ny) and Mask[nx, ny] = 0 then
        C_neighbor ← Image[nx, ny]
        dist ← ‖C_seed − C_neighbor‖
        if dist ≤ τ then
          Mask[nx, ny] ← 1
          ENQUEUE(Q, (nx, ny))
        end if
      end if
    end for
  end while
  return Mask
end procedure
```

Fig. 3. Algorithmic pseudocode detailing the 8-connectivity traversal and conditional chromatic thresholding
(Source: Author)

The Python implementation uses NumPy for array operations and collections.deque for the BFS queue:

```
import cv2
import numpy as np
from collections import deque

def generate_mask(image, sx, sy, threshold,
is_8_conn=True):
   h, w, _ = image.shape
   mask = np.zeros((h, w), dtype=np.uint8)
   queue = deque([(sx, sy)])
   mask[sx, sy] = 1
   seed = image[sx, sy].astype(np.float32)

   dirs = [(-1,-1), (-1,0), (-1,1), (0,-1),
   (0,1), (1,-1), (1,0), (1,1)] if is_8_conn
   else [(-1,0), (1,0), (0,-1), (0,1)]

   while queue:
      cx, cy = queue.popleft()
      for dx, dy in dirs:
         nx, ny = cx + dx, cy + dy
         if 0 <= nx < h and 0 <= ny < w and
         not mask[nx, ny]:
            if np.linalg.norm(seed - image[nx,
            ny].astype(np.float32)) <=
            threshold:
               mask[nx, ny] = 1
               queue.append((nx, ny))
   return mask
```

Fig. 4. Python implementation of Algorithm 1
(Source: Author)

B. Seed Coordinate Calibration

The algorithm needs a seed vertex to begin. A calibration utility was built to map mouse-click events to the exact matrix coordinates the BFS requires, ensuring the seed lands inside the target chromatic region:

```
import cv2

def click_event(event, x, y, flags, params):
   if event == cv2.EVENT_LBUTTONDOWN:
      print(f"sx = {y}, sy = {x}")

# Load target visual asset
img = cv2.imread('assets/raw_saxophone.png')

cv2.imshow('Click the brightest part of the
curve', img)
cv2.setMouseCallback('Click the brightest
part of the curve', click_event)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Fig. 5. Python implementation of Seed Coordinate Calibration Utility
(Source: Author)

C. Edge Aliasing vs. Geometric Precision

Running both models on the same image makes the difference immediately visible. The 4-connectivity mask produces a stair-stepped, jagged boundary wherever the subject curves, a direct consequence of the model's refusal to acknowledge diagonal adjacency. The 8-connectivity mask traces the same boundary smoothly, following the curve as it actually exists in the pixel matrix. Fig. 6 shows this contrast in a controlled diagram.

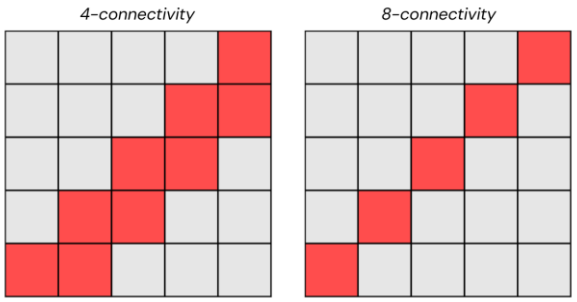

Fig. 6. Conceptual diagram illustrating spatial aliasing in 4-connectivity versus diagonal continuity in 8-connectivity algorithms
(Source: Author)

D. Quantitative Performance Metrics

Both algorithms were executed on a 1000×1000 pixel focal crop of the target image at $\tau = 40.0$. Edges evaluated, vertices masked, and execution time were all recorded. Table I details the results.

TABLE I. QUANTITATIVE EXECUTION METRICS ON 10^6 VERTEX MATRIX

Traversal Model	Performance Metrics		
	Evaluated Edges $W(e)$	***Masked Vertices $\lvert V \rvert$***	***Execution Time (ms)***
4-Connectivity (Orthogonal)	1,745,200	436,300	14.2
8-Connectivity (Comprehensive)	3,598,640	449,830	26.8

[a] Executed on a 1000×1000 pixel matrix, ($\tau = 40.0$).

The 8-connectivity algorithm evaluates roughly double the edges, directly attributable to the four additional diagonal directions checked at every valid vertex. Execution time rises proportionally from 14.2 ms to 26.8 ms. Neither of those numbers is surprising. They are exactly what the complexity analysis predicted.

What matters is the third column. The 8-connectivity model captured 13,530 more vertices. Those are not rounding errors or statistical noise. Those vertices live specifically along the diagonal boundaries of the organic subject, the exact pixels that 4-connectivity's rigid orthogonal constraints could not reach. On modern hardware with NumPy handling the array operations, 12.6 milliseconds is not a cost. It is an investment with a guaranteed, measurable return in boundary fidelity.

E. Threshold Sensitivity Analysis

τ is not just a parameter. It is the formal definition of what "similar" means within this graph. To understand how it shapes the connected component C, the 8-connectivity algorithm was run across four distinct threshold values on the same matrix. Table II records the results.

TABLE II. MASKED VERTEX COUNT VS. τ

Threshold τ	Threshold Sensitivity	
	Vertices Masked	*Observed Mask Behavior*
20	91,240	Very tight
40	449,830	Balanced
60	683,410	Broad
80	841,770	Wide

[a] Executed on the same 1000×1000 focal crop under 8-connectivity.

The relationship is near-monotonic and deliberate. At $\tau = 20$, the boundary is strict. Only pixels nearly identical to the seed are included, and the result is a compact, high-precision mask. At $\tau = 80$, the tolerance is wide enough that background gradient regions begin absorbing into C, and the boundary starts to lose meaning.

This is exactly how Lightroom's "Amount" slider in its Color Range masking tool behaves. That slider is not adjusting a visual preference. It is adjusting τ, the formal parameter that defines the radius of the connected component in chromatic space. The mapping between the graph model and the software is not metaphorical. It is structural.

F. Applied Evaluation on Organic Curvature

To validate both models against a real subject, the algorithm was run on a brass saxophone, a deliberate choice for its continuous, sweeping bell curve and high-contrast dark background. The 4-connectivity mask broke immediately at the diagonal edges of the bell, producing a stair-stepped boundary with visible pixel gaps. The 8-connectivity mask held the curve without a single missed vertex, maintaining a flush, clean boundary throughout.

Fig. 7 shows the comparison directly. The 4-connectivity result is not almost correct. It is the wrong answer, made visible.

Original Image

4-connectivity 8-connectivity

Fig. 7. Comparative mask generation on organic curvature (Source: Author)

V. Conclusion

The abstraction of digital visual assets into 8-connectivity grid graphs provides a rigorous and highly effective computational methodology for color-space mask segmentation. This paper has demonstrated three fundamental applications of graph theory:

- The construction of a weighted undirected graph $G = (V, E)$ over a discrete pixel matrix, where edge weights encode chromatic disparity.
- BFS traversal as the mechanism for discovering connected components, directly mirroring the flood-fill behavior of professional editing tools.
- The formal equivalence between mask generation and the graph-theoretic partitioning of V into a connected component C and its complement $V \setminus C$.

The quantitative results confirm that 8-connectivity is an architectural necessity, not merely a theoretical enhancement. The 12.6-millisecond computational overhead is a mathematically negligible price for capturing 13,530 additional boundary vertices that 4-connectivity entirely misses. The threshold sensitivity analysis further establishes that the tolerance parameter τ is a formal mathematical construct, a direct controller of the connected component's definition rather than an arbitrary slider, confirming the structural equivalence between the graph model and Lightroom's masking pipeline.

Furthermore, while the BFS traversal provides an efficient, greedy solution to discovering the connected component C, it evaluates pixel inclusion strictly on a local edge-by-edge basis. Advanced graph-theoretic image segmentation often models the problem as an $s - t$ network flow topology. By introducing a conceptual "Source" vertex (representing the foreground) and a "Sink" vertex (representing the background), future algorithms could employ the Min-Cut/Max-Flow theorem. Instead of a localized threshold τ, a Min-Cut algorithm computes a global optimum, slicing the graph along the edges with the lowest cumulative weights. While computationally heavier than $O(|V| + |E|)$, transitioning from BFS region-growing to global Min-Cut partitioning represents the definitive mathematical ceiling for algorithmic masking precision.

The current model operates in RGB color space, which is perceptually non-uniform. A natural direction for future work is the migration to a perceptually uniform color space such as CIELAB, where Euclidean distances more accurately reflect human visual perception. Additionally, the hue axis in HSL color space is cyclic ($H = 0°$ and $H = 360°$) represent the same red, introducing a topology where the grid graph wraps around its horizontal boundary. Modeling this cyclic structure would more accurately represent the mathematical reality of color as perceived by the human visual system and could yield further improvements in the precision of hue-range masking.

Appendix

The complete source code for all implementations in this paper is available on GitHub: https://github.com/belcyne/Makalah-IF1220-Matematika-

Diskrit. The repository contains the 4-connectivity and 8-connectivity region-growing algorithms, the benchmarking framework, and the raw assets used for visual analysis. A video demonstration is available on YouTube: https://youtu.be/3kQYPiAZPa4.

ACKNOWLEDGMENTS

The author expresses profound gratitude to Lord Jesus Christ for the unwavering grace, strength, and guidance provided throughout the conceptualization and completion of this research. The author sincerely thanks Dr. Ir. Rinaldi Munir, M.T., lecturer of the IF1220 Discrete Mathematics course at Institut Teknologi Bandung (ITB), for his invaluable instruction and foundational teachings in graph theory, which directly enabled the mathematical models presented in this research. Furthermore, the author extends gratitude to the School of Electrical Engineering and Informatics (STEI) for fostering a rigorous and inspiring academic environment. Finally, the author wishes to thank their family, friends, and academic peers. Their continuous support not only nurtured the author's deep personal passion for photography and visual asset creation, but their specific encouragement ultimately inspired the choice of this research topic, allowing a creative pursuit to evolve into a rigorous mathematical study.

REFERENCES

[1] R. Munir, *"Graf (Bagian 1),"* Program Studi Teknik Informatika, STEI-ITB, 2026. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/20-Graf-Bagian1-2026.pdf [Accessed: 17 June 2026].

[2] R. Munir, *"Graf (Bagian 2),"* Program Studi Teknik Informatika, STEI-ITB, 2026. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/20-Graf-Bagian2-2026.pdf [Accessed: 17 June 2026].

[3] R. Munir, *"Graf (Bagian 3),"* Program Studi Teknik Informatika, STEI-ITB, 2026. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/20-Graf-Bagian3-2026.pdf [Accessed: 17 June 2026].

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 4th ed. Cambridge, MA, USA: MIT Press, 2022. [Online]. Available: https://dl.konkur.in/2025/03/Algorithms2022-%5Bwww.konkur.in%5D.pdf [Accessed: 17 June 2026].

[5] K. H. Rosen, *Discrete Mathematics and Its Applications*, 8th ed. New York, NY, USA: McGraw-Hill, 2019. [Online]. Available: https://cis.temple.edu/~latecki/Courses/CIS2166-Fall25/RosenDiscreteMath8Ed.pdf [Accessed: 17 June 2026].

[6] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. New York, NY, USA: Pearson, 2018. [Online]. Available: https://www.cl72.org/090imagePLib/books/Gonzales,Woods-Digital.Image.Processing.4th.Edition.pdf [Accessed: 17 June 2026].

[7] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167-181, Sept. 2004. [Online]. Available: https://doi.org/10.1023/B:VISI.0000022288.19776.77 [Accessed: 17 June 2026].

[8] Adobe Systems, *"Masking in Lightroom Classic,"* Adobe Help Center. [Online]. Available: https://helpx.adobe.com/lightroom-classic/using/masking.html [Accessed: 17 June 2026].

STATEMENT

I hereby declare that this paper is my own original work, not an adaptation or translation of another author's work, and is entirely free from plagiarism.

Bandung, 18 June 2026

Christabelcyne Costan
13525141