Parametric-Based Models for Artistic Representations

PhD Thesis

Manuel Rodríguez Ladrón de Guevara

Doctoral Candidate in Computational Design

School of Architecture, Carnegie Mellon University

Advisor:

Ramesh Krishnamurti

Professor, PhD Computational Design Track, School of Architecture, Carnegie Mellon University

Co-advisor:

Daragh Byrne

Associate Teaching Track, School of Architecture, Carnegie Mellon University

Committee members:

Jun-Yan Zhu

Professor, Robotics Institute, Carnegie Mellon University

January 27, 2024

CARNEGIE MELLON UNIVERSITY

School of Architecture

College of Fine Arts

Dissertation

Submitted in Partial Fulfillment of the requirements for the degree of

Doctor of Philosophy

TITLE:

Parametric-Based Models for Artistic Representations

PRESENTED BY: **Manuel Rodríguez Ladrón de Guevara**

ACCEPTED BY ADVISORY COMMITTEE:

Ramesh Krishnamurti	Advisor	Jan 29, 2024 DATE
Daragh Byrne	Co-Advisor	DATE
Jun-Yan Zhu	Committee Member	Jan 27, 2024 DATE
Joshua Bard	Observer	1.29.2024 DATE

I hereby declare that I am the author of this dissertation

I authorize Carnegie Mellon University to lend this dissertation to other institutions or individuals for the purpose of scholarly research.

I further authorize Carnegie Mellon University to reproduce this dissertation by photocopy or other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_

Manuel Rodríguez Ladrón de Guevara

Copyright © 2024 Manuel Rodríguez Ladrón de Guevara
All rights reserved

Abstract

Driven by the popular adoption of AI for artistic purposes, this research examines its technical and ethical implications, and presents new approaches for parametric algorithms that generate different type of artistic representations, based on optimization methods and learning models. In the midst of the success of machine learning algorithms in image generation and other *creative* tasks of pixel-based diffusion and large language models, my dissertation is framed within the field of parametric representations, which is closer to human art than pixel-based methods. An array of computational methods including procedural, optimization, and machine learning, are analyzed and proposed with the intention of disseminating the generation of artwork by computers.

To understand and situate the current AI models used for art, we provide a comprehensive review and implementation of artistic computational methods, ranging from classical procedural hand-made, rule-based algorithms, to the most advanced AI methods. After analyzing the technical possibilities of existing methods, we present new algorithms that focus on particular gaps in the literature.

Particularly, this research studies three main problems that exist in the literature: *stylization, controllability*, and *identity preservation*. Stylization, the process of applying a particular artistic style to an image, is a common subject within generative algorithms, and key to artistic success. Controllability enables intentional painting processes and thus, it is a steppingstone for further stylization. Identity preservation is the ability for a learning model to preserve key content features from the original image through the artistic representation process.

Finding the right combination of stroke primitives for a particular artistic style and, by extension, for a painting strategy that leads to certain styles is not fully resolved under a parametric framework, as there exists a trade-off between reconstructions of the input image and a controllable stylistic variation. Existing works that address style are still limited in style variations and controllability, and principally use different stroke models and textures to output styles. State-of-the-art algorithms normally output strokes in an uncontrollable manner without a planned strategy that might help stylization. However, human artists employ painting techniques such as "blocking in", grouping by semantics or colors, "background-foreground" or "color-then-contours" that help them convey artistic styles.

Throughout the different algorithms presented, I demonstrate that we can achieve new ways to find stylization, controllability and identity preservation. We disentangle such a complex landscape of artistic styles and strategies, and leverage some artistic vision under some perception of art. We finally tap into computational creativity, whether algorithms can be creative, and discuss future steps in the field of machine learning and art.

Acknowledgements

As I embarked on this PhD journey, I was a licensed architect with some design experience, freshly graduated from a Master's in Advanced Architectural Design at Carnegie Mellon. With no published papers and only a nascent grasp of academic writing, it was Professor Ramesh Krishnamurti who saw potential in me and took me under his wing as his doctoral student. Thank you, Professor, for not only guiding me to where I am today but also for shaping who I have become. Your invaluable advice, encouragement to step out of my comfort zone, lessons in scholarship, enlightening conversations, and constant guidance mean more than words can express. You are much more than a mentor to me; you are a role model.

I owe immense gratitude to Daragh Byrne, my co-advisor, for your unwavering trust and support. Your rigorous feedback has been instrumental in my growth as a student. Thank you for always being available for a conversation, attentively listening to my concerns, and helping me navigate through this program.

My sincere thanks also go to Jun-Yan Zhu, a member of my committee. Your masterclasses were a source of inspiration and learning, and your support has been invaluable.

I would also like to thank my PhD colleagues and friends, who shared this journey with me. Thank you, Ardavan Bidgoli, Pedro Veloso, Jinmo Rhee, Emek Erdolu, Leo Liu, Yuning Wu, Varvara Toulkeridou, Sanaz Saadatifar, Javier Argota, for making this path both easier and more enjoyable.

This journey would not have been possible without a very special person, Jeremy Ficca, the catalyst for my journey to the U.S. and the MAAD program, which led to my PhD. Your guidance as my previous advisor, mentor, and friend has been a cornerstone

in my academic and personal life. I also extend my gratitude to Steve Lee, for believing in me and supporting my transition to the PhD program, and to Omar Khan, for his ongoing support. My appreciation goes to Josh Bard, Daniel Cardoso, Kai Gutschow, and Dave Koltas for their valuable contributions to my journey.

Teaching at CMU has been an integral part of my PhD experience. I am thankful to all the faculty members I've had the pleasure of working with, especially Dana Cupkova, Sarah Rafson, Tommy Yang, Hal Hayes, Laura Garofalo, and the rest of the faculty peers. A special mention to Mary-Lou Arscott, whose support has been pivotal at many levels; your influence is profoundly appreciated.

I would like to thank Aaron Hertzmann, who gave me the opportunity to intern at Adobe Research, and for mentoring me and significantly shaping my research. My time at Adobe Research was transformative thanks to you. Your generous assistance, insightful feedback, and guidance have left a permanent mark in my work. I also thank Matt Fisher for his mentorship and critical feedback. I would also like to thank Daichi Ito, Yannick Hold-Geoffroy, Jose Echevarria, Yijun Li, and Cameron Smith for the enriching time at Adobe Research during a second internship, and for all the inspiring conversations and contribution to my research.

It is hard to find words to express how lucky I am to have shared this journey with Luis Borunda. We started this together when I was in the Master's program, and ever since, we have unconditionally supported each other. Without you this could not have been possible. I am forever grateful for all the moments we've shared.

Sinan, my buddy, thank you. You are my Pittsburgh. Your friendship, support, conversations, understanding, and experiences have added immeasurable value to my journey. I am eternally grateful for the time we have spent together, and for always helping me out ever since we met in 2017.

To Ramoncillo, my other half, my best friend, without you, I would not be where I am now. You brightened my life, a continuous source of inspiration, and the person who introduced me to the field of AI. I will be forever grateful for sharing my life with you. Our continuous support throughout our PhD journeys has been truly empowering.

Our dreams keep the machine going. Thank you for influencing me the way you did. I am grateful for how inspiring this has been. Here's to continuing the journey together, side by side, always laughing, always inquiring.

Finally, to my family—my mom Manuela, my dad Alfonso, my brother Miguel Angel—you are my foundation and driving force. Thank you for your unconditional love and support. For the efforts and sacrifices you've made for me to be here and achieve my dreams. Thank you for understanding me when I've been stressed or unresponsive. Thank you for helping me at every single level in life. Thank you for believing in me and for being the source of my motivation, my inspiration. For helping me sort the bad moments, and celebrate the good ones. For all the FaceTime calls, for your guidance, advice, and for the freedom you have given me. Mamá, papá, Miguel Angel, this achievement is as much yours as it is mine.

Contents

1 Introduction **31**

1.1 Research Method 35

1.2 Research Motivation, Objectives and Contributions 36

1.3 Thesis Structure 38

2 Historical and Computational Foundations of Art **42**

2.1 Technological Evolution in Art 43

2.1.1 Prehistoric Pigment Technology 43

2.1.2 Material Innovations in Ancient Art 44

2.1.3 Renaissance Innovations: Perspective and Oil Paint 46

2.1.4 Daguerreotype 46

2.1.5 Computational Art 47

2.1.6 Artificial Intelligence in Art 49

2.1.7 Parametric-Based Models 54

2.2 Literature Review 55

2.2.1 Introduction to Non-Photorealistic Rendering 55

2.2.2 Non-Learning-Based SBR Methods 58

2.2.3 Neural-Based SBR Methods 65

2.3 Foundations for Parametric-Based Models 69

2.3.1 The SBR Model 69

2.3.2 Overview of Loss Functions for Gradient Methods 72

2.4 Conclusion 74

3 Im2Painting: Economical Painterly Stylization 86

3.1 Introduction 88
3.2 Related Work 90
3.3 Painting Framework 92
3.3.1 Model Architecture 93
3.3.2 Stroke Parameterization and Differentiable Renderer 94
3.3.3 Loss Functions and Optimization 95
3.4 Economical Painterly Styles 97
3.4.1 Comparison with State-of-the-Art Methods 103
3.5 Conclusion and Discussion 105

4 Stroke Optimization Methodology 111

4.1 Introduction 112
4.2 SOBP Methodology 113
4.2.1 Loss Functions for Parameter Optimization 113
4.2.2 Differentiable Rendering Engine 114
4.2.3 Overview of Optimization Framework 116
4.2.4 Preliminary Approach: Naïve Implementation 118
4.2.5 Hierarchical Coarse-to-Fine Algorithm 124
4.3 Conclusion 131

5 Segmentation-Based Parametric Painting 136

5.1 Introduction 137
5.2 Related Work 139
5.3 Method 141
5.3.1 Layered Painting and Patch-Based Optimization 142
5.3.2 Optimization and Loss Functions 144
5.4 Experiments 145

5.4.1 Style Variations . . . 146
5.4.2 Comparison with State-of-the-Art . . . 150
5.5 Conclusion . . . 155
5.5.1 Limitations and Discussion . . . 155

6 Cross-modal Latent Space Alignment for Image to Avatar Translation 162
6.1 Introduction . . . 164
6.2 Related Work . . . 165
6.3 Dataset . . . 167
6.4 Method . . . 169
6.4.1 Unpaired Modality-Specific Representations . . . 170
6.4.2 Cross-Modal Alignment . . . 171
6.4.3 Hair Pipeline . . . 173
6.4.4 Rendering Engine . . . 173
6.4.5 Implementation Details . . . 174
6.5 Experiments . . . 175
6.5.1 Comparison to State-of-the-art Methods . . . 176
6.5.2 User Study . . . 178
6.5.3 Ablation Study . . . 180
6.6 Conclusion . . . 182
Interlude: Reflections on AI and the Nature of Creativity . . . 188

7 Conclusion 210
7.1 Research Summary . . . 211
7.1.1 Contextual Relevance . . . 216
7.2 Limitations, Implications for Future Technology, and Future Work . . . 217

A Im2Painting: Supplementary Material 225
A.1 Higher Resolution Paintings . . . 226
A.2 Ablations on Model Architecture . . . 226

A.2.1 Visual Abstraction . . . 227
A.2.2 Visual Abstraction Painting Progress . . . 229
A.2.3 Artistic Styles . . . 229
A.2.4 Precise Representations on Portraits . . . 230
A.2.5 Comparison with SOTA on Landscapes . . . 235
A.2.6 Stroke Budget Ablations . . . 237
A.3 Loss Functions . . . 237
A.3.1 WGAN-GP Loss . . . 238
A.3.2 Contrastive Loss . . . 238
A.4 Oil Painting Comparison . . . 240

B Segmentation-Based Parametric Painting: Supplementary Material 243
B.1 Optimization Method . . . 244
B.1.1 Implementation Details . . . 244
B.1.2 Style Parameters . . . 247
B.2 Comparison with SOTA . . . 248
B.3 User Study . . . 249

C Cross-modal Latent Space Alignment for Image to Avatar Translation: Supplementary Material 271
C.1 Applications . . . 271
C.2 Rendering Engine . . . 272
C.3 High Resolution Samples . . . 273
C.4 Dataset . . . 275
C.4.1 Parametric Avatar . . . 275
C.4.2 Paired Dataset . . . 276
C.4.3 Unpaired Dataset . . . 278
C.5 Model . . . 278
C.5.1 Model Architecture . . . 278
C.5.2 Training . . . 279

C.5.3 Inference and Hair Pipeline 279
C.6 Ablation Studies 280
C.6.1 Pretrained Image Encoders Ablations 280
C.6.2 Effect of Amount of Data 282
C.6.3 Loss Ablations 283
C.7 Comparison with Previous Methods 286
C.7.1 Inference Runtime 287
C.7.2 User Study 288

List of Figures

1.1 Expressionist painting of a Parisian tapestry of buildings from an aerial perspective. Generated by the method presented in Chapter 5. 31

2.1 Examples of early paintings. (a) Upper Paleolithic polychrome cave painting of Magdalenian bison at the Cave of Altamira, 34,000 BC, Spain. (b) The bull-leaping fresco, 1450 BC, Stucco panel with scene in relief. Unknown author. (c) Hades abducting Persephone, mural painting in the royal tomb at Vergina, Macedonia, Creece, Unknown author . 44

2.2 Computational Art in the early days: (a) '(Des)Ordres', plotter drawing 1974, Vera Molnar. (b) 'Hommage à Paul Klee 13/9/65 Nr.2', 1965, Frieder Nake. (c) 'Untitled Computer Assisted Drawing', 1975, Paul Brown 48

2.3 AI-generated art: (a) 'Memories of Passersby I', 2018, Mario Klingemann. (b) 'Myriad (Tulips)', 2018, Anna Ridler. (c) 'Electric Fan', 2017, Tom White. (d) 'Neural Zoo', 2018, Sofia Crespo 50

2.4 "Prehistoric paleolithic cave painting, depicting hunting and spiritual rituals, found in Africa." Image generated by using Diffusion models. . 52

2.5 Architecture difference between a parametric model (left) and a pixel-based model (right). The former produces a sequence of brushstroke parameters that are rendered, in order, onto the canvas using a differentiable rendering engine G. Pixel-based models generate the matrix of pixels at once. 53

2.6 Examples of Non-photorealistic renderings (NPR): (a) An NPR used for 3D modeling in engineering and architecture design. (b) An NPR of an existing 2D landscape image using a learning-based algorithm. (c) An NPR of an abstracted chameleon using a line drawing using an optimization algorithm. 56
2.7 Painterly renderings with method [29] (a), [28] (b) and [21] (c). Original images are shown as insets. 59
2.8 Stroke modeled as a quadratic Bézier curve 70
2.9 Diagram of an SBR algorithm with four main components: controller or agent, renderer, a canvas and a stroke model. 71

3.1 Painting Stylization. Our method optimizes brush stroke arrangements from an input image (a), (e) to create various abstraction levels and styles. Left: The best optimization settings results in a realistic style (b). Adjusting constraints and reducing stroke count alters abstraction levels (c-d). Right: Different styles are achieved by adding loss terms and constraints: (f) uses short strokes for structure emphasis, (g) applies more strokes in the foreground for detail, (h) utilizes thick, short strokes for a coarse style, and (i) renders an oil-painting style at 512x512 resolution. Except for (d) with 100 strokes, all paintings use 300 strokes. . . 87
3.2 Economical painting styles motivate our work. These landscape paintings—oil paintings by Henri Matisse (Canal du Midi, 1899) (leftmost), Clarence Gagnon (Crépuscule D'hiver, 1915) (center-left) and two from unkonwn digital painters—illustrate how an artist can convey a scene with relatively few brush strokes. 89
3.3 Painting progress of different baselines. Our method places the first 300 strokes more efficiently than previous methods, and captures facial features with fewer strokes. We see this as a measure of economy, which enables stylistic variations. 91

3.4 Model overview. We train an end-to-end painting network without stroke supervision. (Left) The framework takes an image as input and produces a stylized painting. There are two main components, an encoder function that extracts visual features from an input image, and a projection layer that maps such features into a sequence of brushstroke parameters. Finally, a differentiable renderer sequentially renders the brushstroke parameters onto a canvas. An energy function $\mathcal{L}$ is a combination of loss functions that, in conjunction with the network type and optimization strategy control the style of the output. (Right) Im2Painting Decoders: We propose two decoders, D_{FC} for visual abstractions, consisting on 2 fully connected layers followed by non-linearities (a) and D_{LSTM} which adds an LSTM architecture for precise reconstructions and artistic styles (b). 93

3.5 512x512 300-stroke economical paintings. These examples illustrate results with higher resolution, and show how our method can provide nuanced, interpretable control over styles through varying losses and other interpretable factors. (b) Shows a basic style by smooth visual abstraction, (c) adds constraints on thickness and length that reduces the number of small strokes, (d) adds random noise to strokes resulting in a scratchy style, (e) abstraction in oil-painting style. 95

3.6 Accurate Reconstructions and Visual Abstractions. (Left) Our network is able to paint accurate reconstructions with a limited budget of only 300 strokes. This is achieved using decoder D_{LSTM}. (Right) We can increase visual abstraction by decreasing stroke budgets and using coarser loss functions. (a) Input image. (b) 300 strokes using $\mathcal{L}_1$ on all intermediate canvases. (c) 100 strokes using $\mathcal{L}_1$ on all intermediate canvases. (d) 300 strokes evaluating $\mathcal{L}_1$ only on the final frame. We use D_{FC} for all abstractions. Readers are encouraged to zoom in closely for better inspection. 98

3.7 Painting Styles (left) and SOTA Comparison (right). Painting Styles (left): (a) input image, (b) style by constraint on stroke length. A small penalty encourages the network to approximate the image with short strokes. The network generally chooses a larger stroke thickness to balance the lack of long strokes. (c) Style by stroke width penalty. The network learns to quickly approximate the upper part of the image with very few strokes that receive a penalty, creating a detailed foreground and coarse background effect. (d) A combination of constraints on width and length leads to a style of thick and short rounded shapes in a coarse but smooth style. (e) Scratchy effect by approximating rapid motions with Gaussian noise. We use D_{LSTM} for all styles. SOTA Comparison (Right): (a) Original, (b) our results using D_{LSTM}. (c) and (d) Learning-to-Paint [12] limited to 300 strokes and full model, respectively, (e) Paint Transformers [15] limited to 300 strokes. 101

3.8 Precise reconstructions comparison with previous methods. All methods use 300 strokes. Our method with a fine-tuning step (a) is able to reconstruct the input image better. Without fine-tuning step (b) our method still outputs smoother paintings than previous work. Paint Transformers [15] and SNP [30] output very large strokes and they suffer with reduced stroke budgets. 105

4.1 Coarse to Fine Detail Approach. Progression of detail levels using different stroke and width settings: (a) 25 strokes, no clipping; (b) 36 strokes, 10% width clipping; (c) 81 strokes, 5% width clipping; (d) 324 strokes, 5% width clipping. The original image is shown as an inset. . . 111

4.2 Neural renderer architecture provided by [4] 114

4.3 Stroke optimization Pipeline. Black arrows correspond to forward propagation, red arrows correspond to the backpropagation of the gradients. 116

4.4 Difference of quality of paintings using the neural differentiable renderer and the non-differentiable renderer. After optimization, stroke parameteres can be rendered at higher resolutions using non-differentiable renderers. 118
4.5 Impact of random and grid stroke initialization in the final painting. Top row corresponds to a grid stroke initialization, and bottom row corresponds to random stroke initialization. Columns correspond to clipping the maximum stroke width the model is allowed to paint with: (a) no constraint, (b) maximum width 0.2*canvas size, (c) 0.1*canvas size and (d) 0.04*canvas size. (e) Input and optimization details. All paintings use 324 strokes, and L1 as pixel loss function. We let the optimization run for 800 iterations. Right: L1 loss graph over 800 iterations. 120
4.6 Comparison of the painting progress across 2 parameters: grid and random initialization, represented as G and R, respectively, and maximum stroke width, 1 and 0.05 (100% and 5% of the canvas width). The optimization process is shown every 10 iterations on each column for the first 100 iterations. The final painting is shown in last column. 121
4.7 Difference in the impact of loss functions. We show the difference in the outcome when using L1, perceptual loss and CLIP loss. A weight of 0.01 seems very low for either perceptual or CLIP to make an impact. However, a weight of 0.1 makes the perceptual loss define some fine details than L1 loss alone cannot (see definition in the toes). CLIP loss introduces some noise in the painting. 122
4.8 Comparison between two stroke initialization methods. Left block of images is initialized with a grid of strokes. Right block of images is initialized with random stroke initialization. A random initialization increases the complexity of the optimization landscape. 123

4.9 Coarse to fine approach, 512x512 painting. (a) Coarse detail level with 25 strokes per patch and no stroke width clipping. (b) Medium detail level with 36 strokes per patch and width clipping of 10% of patch size. (c) Fine detail level with 81 strokes per patch and width clipping of 5% of patch size. 124
4.10 Analysis of Computational Costs in Patch-Based Painting. (a) Demonstrates the non-linear computational cost associated with the painting-by-patches approach using a 100-stroke budget per 128x128 patch over 400 optimization steps. Doubling the image size does not result in a linear cost increase. (b) Illustrates the near-linear computational cost incurred with increasing stroke budget. 127
4.11 Canvas composition by patches. (a) Patches are organized into a structured grid without overlaps. (b) Example of an organization of patches in a structured grid with overlaps of 20 pixels. (c) Initialization of strokes before being optimized: strokes are evenly distributed across the x and y axis of the patch (upper part). Strokes are randomly sampled from a uniform distribution (bottom part). 129
4.12 Comparison of paintings with uniform and variable stroke counts per patch. (a) Original image with a 20-pixel overlap. (b) Uniform stroke painting: 400 strokes at the coarsest level (25 per patch), 576 at the medium level (36 per patch), and 1296 at the finest level (81 per patch), totaling 2272 strokes. (c) Variable stroke painting: starts similarly at the coarsest level, but uses fewer strokes at the medium (363 strokes) and finest levels (1023 strokes), totaling 1786 strokes. 130
4.13 Different stylistic outputs given by different settings (a) Painting using only 800 strokes and L_1. (b) 2 passes using only thin strokes (c) Realistic painting using more than 2000 strokes (d) Expressionist painting using a combination of L_1 and CLIP loss 132

5.1 Our method provides finer control over the canvas via the use of a segmentation network and dynamic attention maps to guide the painting process. 136
5.2 Comparison of our method against previous work on parametric painting algorithms. Panels (a) through (c) demonstrate our algorithm's capability to render scenes in various styles: abstract with simplified forms and abstract with a heavy emphasis on thick loose strokes, texture, and a lack of clear form (a), painterly impressionistic style with dynamic strokes and emphasis on effect (b), and realistic style capturing both micro and macro structures with high fidelity (c). These are juxtaposed against existing methods: Stylized Neural Painting (SNP) [40] (e), Paint Transformer (PT) [23] (f), and Learning to Paint (L2P) [15] (g), which exhibit limitations such as imprecise edge depiction and a lack of stylistic diversity. SNP and PT use rigid, blocky strokes that are unable to capture fine details, edges and shapes. L2P captures details better, although yields blurry results with visible seams at large resolutions. The input image is displayed in panel (d) for reference. This comparison underscores our method's superior ability to balance detail retention with stylistic expression. Zoom-in is encouraged to see the nuances of each work. 138
5.3 Overall Painting Framework. Our framework models the painting process as a set of sequential painting passes. (a) Pipeline of each painting pass k, which is composed by four parts: attention patches, cropping, optimization, and pasting. (b) Semantic layers and patch strategies. (c) Stroke optimization module with patch-loss. 141
5.4 Coarse-to-fine approach for a painterly portrait of a woman. (a) Initial coarse layer lays foundational strokes for a later refinement (b), guided by dynamic attention maps (insets). 143

5.5 Our approach offers enhanced precision over the canvas's semantic regions, directing specific emphases in the artwork: (a) accentuates the foliage, capturing its essence, while (b) concentrates on highlighting the architectural details of the building. 146
5.6 Different painterly styles given the same input. (a) Painterly style achieved by the use of our segmentation pipeline and selective dynamic attention maps. (b) Realistic style achieved by using uniform dynamic maps. . . . 147
5.7 Demonstration of control over the painting. (a) The painting focal point falls in the buildings while leaving the vegetation abstracted. (b) The motorcycle is emphasized by adding more detail than the background, which is coarsely painted. 148
5.8 1/2. Comparison of four parametric painting techniques—L2P, SNP, PT, and our approach (shown in next figure)—across three different scenarios. Our method's outcomes show improved detail rendition and variability. Note L2P's realistic yet blurry style, SNP's broad painterly effect which struggles to capture fine details (tree branches or windows framing), and PT's inconsistent detail and brushstroke scale. Zoom-in recommended for full appreciation of textural nuances. 151
5.9 2/2. Our realistic approach finely details both high and low-frequency elements, accurately rendering tree branches and window frames. Painterly styles with a different degree of abstraction. This style generates organic human-like paintings, biasing the painting towards different structures. Zoom-in recommended for full appreciation of textural nuances. 152
5.10 Stroke distribution comparison of Figure 5.8 middle column. SNP (a) and PT (c) exhibit a disparse clustering, while L2P (b) paints evenly in a scene-agnostic way. Our method (d) combines the selective focus of SNP with the clarity of PT, indicating a refined approach to applying strokes. 154

6.1 Generated Avatars (bottom row) from their corresponding inputs (top row.) 163
6.2 Avatar Parameterization (right) and training samples (left). We encode the avatar as a vector of 629 parameters, which include point coordinates (represented as dots in the avatar above) which define spline curves, color, and line-weights of facial features. This vector is rendered by a non-differentiable vector graphics engine. 168
6.3 Model overview. (a) Unpaired training stage: we train modality-specific networks to learn rich latent spaces. (b) Paired training stage: we introduce our cross-modal alignment module to translate between different latent spaces to transfer key facial features to a parametric domain. . . . 170
6.4 Inference including hair retrieval pipeline. We use a pretrained attribute predictor $\mathcal{H}$ to pre-compute attribute vectors a of our faces from our hair bank. We use the K-nearest neighbors algorithm to retrieve the closest vector to a query vector $\hat{a}$ from a bank of a vectors. d indicates smallest distance in the KNN search. 174
6.5 Our method supports vector graphics formed by (a) cubic Bezier curves, for contours and outlines, (b) polygons, for highlights and shadows, and (c) custom composites, for eyebrows and hair, with colored strokes and fills. 175
6.6 Comparison with state-of-the-art methods for avatar generation and image-to-image translation. All the methods are trained on the same dataset with the same amount of data. Our method is able to preserve identity with higher accuracy and more consistently than previous methods. . . . 177
6.7 Model variations without proposed losses. Readers are encouraged to zoom in to observe how details affect identity preservation. (b) Uses only L_{rec} between generated and ground truth parameters. (c) Only uses weight alignment loss. (d) Uses all proposed losses. 181

6.8 Effect of paired dataset size. (b) Our model trained on 9970 paired images. (c) Our model trained on 3995 pairs. (d) Our model trained on 1000 image-avatar parameter pairs. (e) Our model trained on 500 samples. 182

7.1 Comparison of the presented methods' outputs given the same input: (a) Original image. (b) Realistic portrait generated by our method Im2Painting, presented in Chapter 3. (c,d) Coarse-to-fine realistic portrait generated by our Semantic Segmentation-Based method, presented in Chapter 5. (e) Avatar generated by our method presented in Chapter 6. 210

7.2 Comparison of two of the presented methods' outputs given the same input: (a,c) Paintings generated by our Semantic Segmentation-Based method, presented in Chapter 5. (b,d) Paintings generated by our Im2Painting method, presented in Chapter 3. 221

A.1 512x512 300-stroke economical paintings. These examples illustrate results with higher resolution, and show how our method can provide nuanced, interpretable control over styles through varying losses and other interpretable factors. (b) Shows a basic style by smooth visual abstraction. (c) Adds constraints on thickness and length that reduces the number of small strokes. (d) adds random noise to strokes resulting in a scratchy style. (e) Oil-painting style. 225

A.2 Model Ablations. Ablation on network architectures for visual abstractions for 300 strokes (a-c) and precise-style network (d). (a) Encoder uses the first 3 residual blocks of Resnet-18 and an additional convolutional layer to extract a larger feature map. Decoder is composed by 1 linear layer followed by non-linear activation. Learnable parameters: ∼31M. **(b) Abstract net used in this chapter**. Encoder uses the first 4 residual blocks of Resnet-18. Decoder has 2 linear layers followed by non-linear activations. Learnable parameters: ∼50M. (c) Encoder is the same as (b). Decoder is composed by 3 linear layers followed by non-linear activations. Learnable parameters: ∼100M. (d) Precise-Style network, used for precise reconstructions and styles. Decoder is composed by a linear layer, followed by a bidirectional LSTM, followed by a final linear layer. Learnable parameters: ∼90M. 227

A.3 Model ablations on visual abstraction. All visual abstraction paintings are made with 300 strokes. Top row: original image. Second row: AbstNet using architecture shown in Figure A.2 (a). Third row: AbstNet using architecture shown in Figure A.2 (b). Fourth row: AbstNet using architecture shown in Figure A.2 (c). Bottom row: Precise-Style-Net (Figure A.2 (d)). The goal of this method is to provide a coarse level of abstraction, and we find a relation between network size and level of abstraction. We see how second and third rows are better abstractions than fourth and last rows. There is not a substantial difference between second and third rows, and both are suitable for this abstracted style. For all visual abstractions shown in the paper, we choose to work with the network corresponding to the third row (A.2 (c)). 228

A.4 Visual abstraction painting progress of different optimization schemas. All three rows use 300 strokes; intermediate canvases are displayed every 25 strokes. Top row: applying the loss only on the last canvas. The network "wastes" the majority of the strokes, with few of them contributing much to the final painting. Middle row: applying the loss only on every 50th canvas. Bottom row: applying the loss at every canvas, as in Equation 3 in main paper. 229
A.5 Model ablations on styles. All paintings are made with 300 strokes. (a) Original. (b) Style produced by our Precise-Style-Net. (c) AbstNet with a similar number of trainable parameters than (b). (d) AbstNet with approximately half of the total number of trainable parameters than the previous two. (b) shows a clear style, defining object boundaries much better than our abstract net, which outputs blurrier paintings. 230
A.6 Model Ablations on Reconstructions. Comparison of StyleGAN-generated face reconstructions using 300 strokes across three architectures: (b) AbstNet with one linear layer (Figure A.2 (a)), (c) AbstNet with two linear layers (Figure A.2 (c)), and (d) Precise-Style-Net with decoder D_{LSTM}. (b) shows coarse strokes with less defined facial features; (c) improves on (b) but lacks in finer details; (d) offers sharper reconstruction with better detail and edge capture. 231
A.7 Comparison with State-of-the-Art-Methods. Top row: Original. Second row: Our results using D_{LSTM} Third and fourth rows: Learning-to-Paint [4] limited to 300 strokes and full model, respectively. Bottom row: PaintTransformers [5] limited to 900 strokes. 233
A.8 Painting progress of different baselines. Our method places the first 300 hundred strokes more efficiently than previous methods, and captures facial features with fewer strokes. We see this as a measure of economy, which enables stylistic variations. 234

A.9 Precise reconstructions comparison with SOTA. All methods use 300 strokes. Our method with a fine-tuning step (a) is able to reconstruct the input image better. Without fine-tuning step (b) our method still outputs smoother paintings than previous work. This is perhaps more visible in second, third, and last rows. Paint Transformers [5] and SNP [7] output very large strokes and they suffer with reduced stroke budgets. 236
A.10 Stroke budget ablations. Precise-Style-Net using (b) 300 strokes, (c) 600 strokes and (d) 1000 strokes. Adding more strokes produces a blur effect on the painting. 237
A.11 Effect of using WGAN-GP and contrastive losses. We substitute perceptual loss by WGAN-GP (c) and contrastive loss (d) following Equation (A.3). 240
A.12 Comparison of oil painting texture via thin plate spline algorithm (top) and oil painting filter in Adobe Photoshop (bottom) 241

B.1 Styles: (a) Expressionist, consisting of thick, textured, loose and low fidelity representation where subject matter is not a priority. (b) Painterly, think, shaped strokes with a loose touch, enough to balance the scene recognition and not yield a high-fidelity painting style, (c) Abstract, scattered, fewer strokes that capture the essence of the scene. 243
B.2 Stroke distribution analysis: SNP and L2P present a uniform distribution across the canvas. L2p is completely scene-agnostic, and SNP cannot distinguish between semantic areas. PT presents a severe contrast between objects and background, and struggles to differentiate between semantic areas of the image. Ours distribute strokes based on semantic areas. 251

B.3 Painting analysis based on semantic layers and dynamic attention maps (DAMs). Top row shows DAMs per semantic layer k. Second row shows the overall stroke distribution over the canvas based on biasing the painting towards different semantic areas, and the third row shows the corresponding paintings. The fourth row shows each layer k independent from the rest of the painting. The fifth row shows the Kernel Density Estimation of the strokes that correspond to only such layer k. The bottom row shows the semantic map, and the four painting passes, from coarse to fine, with a bias on the vehicle over the rest of the scene. 252
B.4 Comparison of our method (a) against L2P [2] (b) on realistic images. Note how L2P paintings have visible seams and blurry areas. 253
B.5 Comparison of our method (a) against PT [4] (b) and SNP [5] (c) on painterly images. 254
B.6 Painterly style with dynamic attention maps corresponding to the last 3 painting layers are shown at the bottom. 263
B.7 Painterly style with dynamic attention maps corresponding to the last 3 painting layers are shown at the left. Painting process is shown at the right. 264
B.8 Realistic Style (left) and painterly style (right). Dynamic attention maps for the last painting pass are shown as an inset. 265
B.9 Realistic painting process. Painted using uniform dynamic attention maps. From coarse (top left) to fine (bottom right) 266
B.10 Pairings shown in user study from section 1. IDs from top to bottom: A5, A3, A4, and A6 267
B.11 Pairings shown in user study from section 1 and 2. IDs from top to bottom: A1, A2, B1, and B5 268
B.12 Pairings shown in user study from section 2. IDs from top to bottom: B3, B6, B2 and B4 269

C.1 Applications of our system beyond images. (a) We can export our parametric vector to animation software, and (b) 3D modeling software . . . 272
C.2 Rendering composition used in our vector-graphics engine. Facial parts are composited on the canvas layer by layer. 273
C.3 Examples of avatars of children. Our model generalizes to kids, even though our paired image-parametric avatar dataset does not contain any children. 274
C.4 Model architecture of the different components of our model. 279
C.5 Avatars generated by our method using different pretrained image encoders. 281
C.6 Effect of amount of paired data. (b) Our model trained on 9970 pairs. (c) Our model trained on 3995 pairs. (d) Our model trained on 2000 pairs. (e) Our model trained on 1000 pairs. (f) Our model trained on 500 pairs 282
C.7 Model variations without proposed losses. (b) only uses reconstruction loss $\mathcal{L}_{rec}$ between generated and ground truth parameters. (c) only uses weight alignment loss $\mathcal{L}_{w}$. (d) uses reconstruction $\mathcal{L}_{rec}$ and cross modal loss $\mathcal{L}_{cm}$ but does not use weight alignment loss $\mathcal{L}_{w}$ (e) uses all except for the MSE loss term in $\mathcal{L}_{cm}$. (f) uses all except for the cosine similarity term in $\mathcal{L}_{cm}$ (g) uses all losses 284
C.8 Comparison of our model with state-of-the-art methods. All methods are trained on the same dataset, comprising 3995 image-avatar pairs. . . 287
C.9 Example of question on identity preservation in our user study 290
C.10 Example of question on avatar quality in our user study 291
C.11 Samples generated using StyleGAN-2 generated faces. 292
C.12 Samples generated using StyleGAN-2 generated faces. 293
C.13 Samples generated using StyleGAN-2 generated faces. 294
C.14 Samples generated using StyleGAN-2 generated faces. 295
C.15 Samples generated using StyleGAN-2 generated faces. 296

C.16 Samples generated using StyleGAN-2 generated faces. 297
C.17 Samples generated using StyleGAN-2 generated faces. 298

List of Algorithms

1 Naïve SOBP 119
2 Coarse-to-fine patch-based SOBP 128
3 Main Painting Process 246

List of Tables

1.1 Comparison of ML models: Image-Based vs. Parametric-Based models. 34

3.1 Quantitative results on 500 random held-out faces from the CelebA dataset. We limit each method to 300 strokes the painting algorithms from previous work. We measure L_1 loss and perceptual loss using cosine similarity between input image and painting feature maps. Our method achieves slightly better metrics in the first 300 strokes, an indication of economy. Note that our 300-stroke model is our full model. 102

3.2 Quantitative results on 100 random sample images drawn from a held-out set on landscapes. We provide metrics on a 300-stroke model (left), and a 1000-stroke model (right). For [15], total number of strokes is averaged over the held-out set. We measure L_1 and perceptual loss using cosine similarity between input image and painting feature maps. For a reduced number of strokes, our method achieves a lower perceptual loss than previous methods. *We include the optional finetuning step for information purposes; a more meaningful comparison is done without such finetuning step. 104

5.1 Comparison with previous methods on attributes like painting control, style variations, large-scale parametric painting, and high-low frequency capture. 150

5.2 Qualitative evaluation based on user preference. This table shows a 2-way comparison between our method and previous methods. 153
5.3 Quantitative results on realistic style across different image domains. All methods use the same number of strokes (4000) on a 512x512 canvas. 154
5.4 Inference efficiency for different methods, measured in seconds for a single image at 1024x1024 and 1600 strokes. 156

6.1 Quantitative evaluation on a held-out dataset. All models are trained on the same dataset containing 3955 avatars. 179
6.2 Quantitative evaluation on user preference. This table shows a 2-way comparison between our method and previous methods. 179
6.3 Quantitative evaluation on the effect of each loss function. 180

A.1 Quantitative results on same held-out set on CelebA as in Table 1 in main paper, using networks explained in A.6. Precise-Style network achieves a much lower L_1 score than AbstNet. The latter, however, achieves a slightly better perceptual score. However, as shown in Figure A.6, it produces slightly worse reconstructions. 232

B.1 Style hyperparameters by default. Users can fine-tune these parameters for further control over the desire styles. 244

C.1 Avatar parameterization by facial components. Hair, glasses and cap parameters are not learned by our main model. Instead, they are retrieved by a separate pipeline which we refer to as "hair pipeline" for brevity. . 277
C.2 Breakdown of the parameters that define the eyebrows. 277
C.3 Training Hyper-parameters 278

C.4 Inference Runtime computed in seconds. GAN-Inversion is needed in StyleGAN2-based methods, where the latent vector corresponding to the input image needs to be found. Inversion is computed over 1000 iterations. *Our forward includes the forward pass through the inference module and the time that of the vector graphics engine to render the avatar in image space. 288

Chapter 1

Introduction

Figure 1.1: Expressionist painting of a Parisian tapestry of buildings from an aerial perspective. Generated by the method presented in Chapter 5.

From the earliest cave paintings to the latest digital creations, the interplay between art and technology has been a constant throughout human history, influencing and trans-

forming artistic expression across eras. This enduring relationship sets the stage for the current era of digital innovation, where technology not only reshapes existing art forms[1] but also inspires new modes of creativity. In this era, machine learning (ML), a pivotal advancement in artificial intelligence (AI)[2] , stands at the forefront of redefining artistic creativity. It offers novel tools and perspectives for artists and technologists alike, marking a significant shift in how art is both created and perceived.

Technological innovations have not only transformed traditional art forms but also the very nature of communication and expression. For instance, the evolution from the ancient alphabet to modern digital symbols like emojis reflects a significant shift in how we convey ideas and emotions. These developments in symbolic representation, from a limited set of alphabetic characters to the expansive array of emojis, reflect broader changes in communication technologies [1, 2]. Such shifts in representation and communication methods are parallel to the transformations occurring in the art world—art is indeed a communication channel—particularly with the integration of ML and AI.

The rise of computational methods has further democratized art creation, empowering those with algorithmic thinking skills to create art through computer programs.

[1] In this thesis, the term 'art' primarily refers to 2D representational media, with a focus on faces and portraiture (avatars) and landscapes. While art encompasses a vast array of forms and practices, including abstract, expressive, interpretive, sculptural, and time-based media, the scope here is limited to the digital realm and particularly to the creation and stylization of images. This focus is due to both the domain of expertise and the current technological capabilities in machine learning.

[2] AI and ML are related concepts. ML is a subset of AI that focuses on the development of algorithms and statistical models that allow computers to learn and make predictions or decisions without being explicitly programmed. ML techniques enable computers to improve their performance on a specific task through experience and data. AI, on the contrary, is a broader concept that encompasses the development of machines capable of simulating human intelligence and behaviour, so AI is concern specifically to creating intelligent systems that can perceive and understand the environment, reason, and make decisions, and thus, learn from experience. Subfields of AI include natural language processing, computer vision, robotics, expert systems and more. I will use AI and ML interchangeably throughout the rest of this dissertation.

Software like Adobe Photoshop (pixel-based) and Illustrator (vector-based) have made it possible for anyone with a computer and the requisite technical skills to engage in artistic creation.

The advent of generative AI and advanced optimization techniques has expanded the capabilities of computational methods in the arts. Driven by this evolving landscape, this thesis delves into the intersection of machine learning and art, investigating how computational techniques can enhance, transform, and redefine artistic expression. By examining the role of machine learning in art, we aim to understand not just how technology can mimic traditional artistic methods, but also how it can foster unique forms of expression that were previously unimaginable.

After an in-depth review of existing literature on computational approaches to art, I introduce three innovative techniques that each address different aspects of artistic creation. This study heavily relies on technical models to explore the current and future potential of machine learning both in the realm of art creation and as a tool for artists. Unlike the popular pixel-based approaches commonly used in research, this work focuses on parametric (vector) representations of art for the scope of my PhD dissertation. Pixel-based methods have dominated the field due to several factors, including data availability and volume—photographs and images are far more abundantly available than vector graphics,—advances and demands in computer vision, and lower mathematical complexity than vector graphics. However, such methods are less suitable for approximating human painting techniques: besides lacking a sequential nature[3] , they are less interpretable and less controllable (humans do not generate a painting from pixels at once.) For a general comparison between pixel-based and parametric-based methods, see Table 1.1.

[3]Painting is a sequential process, that is, strokes are placed on a canvas in sequence, one by one.

	Image-Based Models (Pixel-Based)	Parametric-Based Models (Vector/Stroke-Based)
Basic Description	These models directly manipulate and interpret pixel data for different tasks.	These models represent data in the form of parameters of stroke primitives which, when rendered, become pixels.
Data Type	Deal directly with raw pixel data.	Deal with stroke-based or vector data.
Computation	High computational cost as they process individual pixels.	Lower computational cost as they process fewer data points (strokes vs. pixels).
Scalability	Scale poorly with resolution, as more pixels require more computational resources.	Scale well with resolution, as vector graphics maintain quality regardless of size.
Training Data	Require large amounts of high-resolution, optionally labeled training data.	Can be trained on smaller datasets of paired stroke/vector data, or unpaired images.
Interpretability	Typically less interpretable due to the complex relations between pixels.	More interpretable due to the structured nature of strokes and their parametric representation.
Usage	Used extensively in image processing, object detection, image synthesis, etc.	Widely used in tasks like handwriting recognition, sketch recognition, digital art, etc.
Realism	Capable of producing highly realistic results.	Depending on the complexity of the model, can also produce realistic results but often seen as more abstract or stylized.
Flexibility	High flexibility in terms of image content.	The content is somewhat limited by the parameters of the model.
Dependency	Pixel-based models consider each pixel in the image independently, sometimes leading to inconsistencies.	Strokes have dependencies, resulting in more coherent outputs.

Table 1.1: Comparison of ML models: Image-Based vs. Parametric-Based models.

1.1 Research Method

This dissertation embarks into the intricate domain of computational algorithms for artistic purposes, with a focus on parametric representations. It aims to unravel the limitations of current computational artistic algorithms and critically assess their creative potential. Central to this investigation is the question: How can advancements in parametric algorithms not only refine technical aspects of machine-generated art, such as stylization, controllability, and identity recognition, but also bridge the gap between human and computational creativity, fostering a deeper understanding and appreciation of artistry in the AI era?

The following points summarize the key areas of exploration in this PhD dissertation:

- **Computational Artistic Frameworks:** This segment offers an in-depth analysis of computational artistic frameworks, categorized into non-learning and learning-based models. Non-learning models are further divided into deterministic, heuristic, and optimization-based algorithms. The discussion traces the evolution of computational painting from the early 90s, highlighting seminal works by pioneers like Haeberli and Hertzmann, and contrasts these early methods with contemporary techniques.

- **Algorithmic Artistic Representation:** The dissertation presents new methodologies for artistic representation through both optimization and neural network methods. The optimization-focused research develops algorithms for enhanced *control* in painting creation, with the goal of generating stylistic variation as a function of control. A detailed examination of loss functions provides insights relevant to both optimization and neural methods. One of the neural methods delves into *stylization* based on loss functions, while the other one addresses *identity preservation* in avatar portraits, balancing artistic style with recognizable features.

- **Perception and Creativity of AI-Generated Art:** A critical evaluation of human

perceptions towards AI-generated art is conducted through three experiments, each addressing distinct technical aspects: stroke optimization, learning stroke patterns for painting, and avatar generation. The proposed chapters delve into the subjective realms of aesthetics and creativity, discussing current technical limitations and the varied reactions and perceptions of human viewers.

1.2 Research Motivation, Objectives and Contributions

This dissertation is rooted in an exploration of ML applications within the artistic domain. Initially inspired by the intricate language of designers and artists, my research has evolved to focus on the interplay between visual communication and ML algorithms. This journey seeks to understand the technical capabilities and limitations of AI in artistic contexts, probing critical questions: Can AI serve as a tool to augment the creative processes of artists, or does it pose a threat to their craft? How might AI catalyze the emergence of novel artistic forms, comparable to the transformative impact of photography on art?

It is important to acknowledge that these questions may invoke inherent biases from the reader, given that art is rarely framed within rigid scientific paradigms, and subjectivity remains integral to artistic evaluation. While the primary scope of this dissertation is confined to a technical analysis of algorithms, the concluding chapter offers an in-depth discourse on the intersections of AI, art, and creativity, aimed at providing a comprehensive understanding of the experimental findings.

Throughout the course of my research, several key objectives have been pursued to advance our understanding of machine learning applications in art. These objectives are as follows:

- **Comprehensive Review of Algorithms:** A thorough review of state-of-the-art algorithms in parametric representations, spanning both AI-based and optimization-based approaches.

- **Comparative Analysis:** An exploration of image-based vs. parameter-based algorithms, as summarized in Table 1.1, highlighting their unique attributes and impacts in the artistic domain.
- **Stylization through ML:** Formulation of ML algorithms focused on stylization, an area predominantly addressed by pixel-based methods, offering a fresh perspective in this field.
- **Optimization for Control:** Development of an optimization framework aimed at enhancing controllability in artistic algorithms, a cornerstone for further advancements in stylization techniques.
- **Identity Preservation:** Design of an ML algorithm for avatar identity preservation, optimized for performance even with limited datasets.
- **Analytical Discourse:** A comprehensive analysis of AI's capacities within the artistic landscape, examining the potential for AI algorithms to exhibit creativity.

In addition to these objectives, the overarching goal of this dissertation is to provide an analytical understanding of parameterized generative deep learning and optimization frameworks. This inquiry covers computational efficiency, fidelity of synthesized images, control granularity, feature interpretability, and identity preservation. A methodical comparative framework, supported by both quantitative and qualitative assessments, aims to guide practitioners in selecting optimal generative methodologies for domain-specific applications.

To this end, this dissertation will leverage a robust methodological toolkit comprising extensive computational experiments, critical literature review, and real-world case studies. The study aims to address the following research questions:

1. What are the specific advantages and limitations inherent to parameter-based approaches across diverse domains and computational tasks?

2. Under what specific conditions do parameterized generative models demonstrably outperform their pixel-based counterparts in terms of efficacy?

Through this rigorous inquiry, the dissertation aspires to provide a comprehensive survey of the current state of generative models, contributing a foundational reference for future academic and practical applications in this dynamic field.

To summarize, this research's contributions lie in providing a technical framework to analyze ML and optimization algorithms, with a focus on *stylization, control*, and *identity preservation*. *Stylization*, commonly explored in pixel-based methods, is reexamined here through the lens of economical efficiency in art—a concept referring to the optimization of time and resources in artistic creation. This perspective on stylization is a new contribution to the field. *Control* is addressed within an optimization framework, demonstrating the potential for exerting precise influence over the artistic process. The research also breaks new ground in *identity preservation*, showcasing superior performance and quality compared to prior work in this area.

These focal points—stylization, control, and identity preservation—define the scope of this dissertation, offering insights into the capabilities and limitations of AI algorithms in artistic applications.

1.3 Thesis Structure

This dissertation is structured in seven chapters, including this introduction:

Chapter 2 provides a brief thesis overview, followed by a historical analysis of art under the lens of technology. Specifically, this section provides a brief description of the evolution of art through the lens of technological advancements, and how the bigger picture shows us how art is to some degree dependent of the technological availability of the moment. Then I situate my work within the non-photorealistic rendering field, and describe the common parts that the majority of parametric algorithms for artistic representation share, regardless the nature of their methods–whether they are learning or non-learning based.

Chapter 3 introduces a learning-based method called "Im2Painting." It explores the concept of "economy" in art and demonstrates how machine learning and optimization algorithms can be structured for efficient stylization, contrasting this approach with other state-of-the-art methods

Chapter 4 introduces the topic of stroke-based optimization algorithms. Specifically, this chapter lays the foundations for a robust painting algorithm that is able to produce style as a result of gaining control over the painting process. For this, this chapter firsts builds a naïve version of a painting algorithm, and provides analysis on loss functions, stroke initialization, and efficiency to understand limitations and potential affordances. We then improve this algorithm under a hierarchical coarse-to-fine approach, and set the ground for the following chapter.

Chapter 5 builds upon the previous chapter and introduces a robust optimization-based painting algorithm inspired by human priors. This algorithm employs segmentation networks and dynamic attention maps to guide and control the painting process, akin to how humans paint. As a result, this algorithm generates a range of artistic styles, from expressionism to realism, and situates itself as the new state of the art in painting algorithms, as corroborated by quantitative and qualitative evaluations.

Chapter 6 addresses a new technical challenge in the literature: identity preservation. This chapter describes a novel machine learning method that automates the process of creating an avatar given an artistic style and a portrait photograph. Even though this chapter works with a different parametric model than the previous two chapters, the high level concept remains the same: from a real photograph to its artistic representation through an efficient optimization of its parameters.

Chapter 7 culminates with a synthesis of the key findings from the main technical chapters, discussing their implications and outlining potential future research directions in the field of AI and art. This chapter also provides a thorough discussion on the ongoing dialogue on AI and creativity, tapping into psychological theories of creativity, and examining whether algorithms can be creative or not.

Appendix A extends the work described in Chapter 3 and provides a supplemental

analysis on the algorithm and its capabilities.

Appendix B provides supplemental information about Chapter 5

Appendix C contains a deeper analysis and more technical information about Chapter 6.

References

[1] Mario Carpo. *The alphabet and the algorithm*. MIT Press, 2011.

[2] Mario Carpo. *The second digital turn: design beyond intelligence*. MIT press, 2017.

Chapter 2

Historical and Computational Foundations of Art

In this dissertation, we embark on an exploration of novel computational methods, encompassing both ML and optimization techniques, aimed at crafting artistic parametric representations. These innovative approaches enable end-users to transform real photographs into diverse artistic styles, marking a significant leap in the realm of digital artistry.

This chapter serves as a foundational component of the thesis by carefully tracing the evolution of painting through the lens of technological advancements. We delve into the historical journey of art, examining how pivotal technological innovations have continually reshaped artistic expression and given rise to new artistic forms. This exploration is not merely historical but also contextual, as we position the narrative within the transformative computational revolution that spanned the final decades of the 20th century and the onset of the 21st century.

In dissecting the intersection of art and technology, we critically analyze various computer graphic methodologies. This includes a thorough examination of early rule-based methods, which laid the groundwork for the subsequent development of complex generative machine learning models and sophisticated optimization techniques. A significant focus is placed on ML models for artistic outputs, breaking down their compo-

nents to unravel the intricate processes behind their functioning.

A key objective of this chapter is to elucidate the distinctions between machine learning and optimization algorithms within the artistic domain. By doing so, we establish a clear understanding of their unique characteristics, applications, and the guiding principles for their training and deployment. This sets the stage for an in-depth exploration in the subsequent chapters, where these technologies are applied to specific challenges in the creation of digital art.

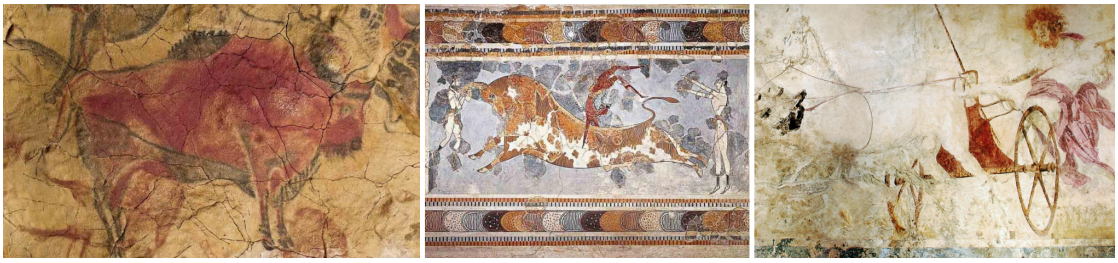
Through this comprehensive overview, we aim to provide the reader with a rich context for understanding the current technological landscape in art. By intertwining historical perspectives with modern computational techniques, this chapter forms an important link between the past, present, and future of artistic expression in the digital age.

2.1 Technological Evolution in Art

2.1.1 Prehistoric Pigment Technology

The advent of painting as a structured artistic medium can be traced back to the Neolithic era, spanning approximately from 10,200 BCE to 2,000 BCE, a period characterized by significant advances in agriculture and societal organization. During this epoch, the medium of artistic expression evolved from predominantly parietal art, such as rock paintings, to more portable substrates, including small ceramic artifacts, as well as mural depictions.

It is noteworthy to mention that the antecedent of these Neolithic art forms is generally identified as the cave paintings from the Paleolithic era. Uranium-thorium dating methods have been employed to date some of these paintings to older than 64,000 years, suggesting a Neanderthal provenance [32, 33]. However, these chronologies have been a subject of academic scrutiny. Slimak et al. [78] dispute the reliability of the oldest uranium-thorium dates and argue for a chronological gap in the artistic record, given



(a) (b) (c)

Figure 2.1: Examples of early paintings. (a) Upper Paleolithic polychrome cave painting of Magdalenian bison at the Cave of Altamira, 34,000 BC, Spain. (b) The bull-leaping fresco, 1450 BC, Stucco panel with scene in relief. Unknown author. (c) Hades abducting Persephone, mural painting in the royal tomb at Vergina, Macedonia, Creece, Unknown author

the absence of corroborating parietal art for a period of approximately 25,000 years postdating the 64,000-year mark.

The paleo-ochre, hematite, and charcoal-based pigments—technological tools at the time—utilized in these prototypical art forms exemplify the utilization of naturally-occurring mineral substances for coloration [60]. The iconography predominantly comprises depictions of fauna, flora, and inanimate objects, executed with varying degrees of mimetic accuracy. Notably, the corpus also includes complex scenes that seem to possess ritualistic or symbolic significance, such as those devised to invoke successful hunting expeditions, enhance fertility, avoid danger, or convey abstract notions and emotional states. Some of these represenations are shown in Figure 2.1.

2.1.2 Material Innovations in Ancient Art

The evolution of artistic materials and techniques is a testament to the ingenuity and creativity of ancient civilizations, profoundly influencing the development and preservation of art through the ages.

The timeline of these innovations begins in ancient Egypt, around 4000 BCE, with the invention of papyrus. Produced by flattening and drying woven reeds, papyrus offered a versatile and more enduring medium compared to its predecessors, such as clay tablets or stone. This innovation not only facilitated the recording and conservation

of art but also played a crucial role in the proliferation of textual and artistic works, marking a significant leap in the history of art preservation [35, 87].

Around 2500 BCE, another significant advancement in Egyptian artistry was the creation of Egyptian blue pigment. This vibrant hue was synthesized by combining limestone, malachite, and quartz and then firing the mixture at high temperatures. The resulting pigment added a new dimension of richness and depth to artistic expressions, demonstrating the ancients' mastery of material technology [7].

Centuries later, the focus shifts to ancient Greece, where pivotal material and representational innovations were made. By the 5th century BCE, Greek artisans had developed ova-tempera and encaustic painting techniques, the latter involving wax-based enamels. These methods not only enhanced the texture and durability of artworks but also introduced new possibilities for artistic expression through layering and translucency [9]. Concurrently, Greek artists began exploring visual perspective and volumetric shading, significantly enhancing the realism of two-dimensional representations [83, 22], although without fully grasping the mathematical laws governing these techniques [38].

The advancement of artistic materials continued into the first century CE with Ts'ai Lun's invention of paper in China. Paper, lighter and more portable than papyrus, democratized access to art and literature, reflecting broader societal transformations [56, 4]. Further contributions from China during the Tang Dynasty around 650 CE included the development of porcelain. This unique ceramic material, known for its delicate yet robust nature, opened new avenues for artistic detailing in both sculpture and decorative arts [44].

Finally, the paradigmatic shift towards mural paintings, particularly frescoes, emerged as a dominant artistic medium for about 1,500 years. Ancient Greek styles, characterized by figural representation combined with geometric designs, covered a range of themes from religious worship to natural landscapes [92]. This era's advancements in material and artistic techniques laid a foundational framework that continued to influence art for centuries.

2.1.3 Renaissance Innovations: Perspective and Oil Paint

Between the 13th and the 15th century art radically changed with new advancements. In the early 14th century, Giotto di Bondone introduced groundbreaking techniques in representational painting, incorporating depth, perspective, and temporal realism. This monumental shift in artistic concept and execution set the stage for the European Renaissance, which would take these ideas to even more sophisticated heights [31, 2]. In the 15th century, the advancement of linear perspective emerged as a seminal development in the artistic domain, notably through the contributions of Filippo Brunelleschi and the subsequent theoretical treatise by Leon Battista Alberti. Linear perspective serves as a mathematical framework for simulating three-dimensional space within the constraints of a two-dimensional plane, thereby elevating the accuracy and complexity of representational art [1]. Concurrently, the advent of oil-based paint media supplanted the extant technology of fresco painting, offering artists an expanded chromatic and tonal range. This material innovation facilitated the creation of visually richer and more intricate imagery, and, after a period of iterative refinement, artists achieved unprecedented levels of realism in their works. During this epoch, in the absence of photographic technologies, static representations of the world were solely accessible through the work of highly skilled artists, making these advancements particularly pivotal in the historical context of visual representation [27].

2.1.4 Daguerreotype

In the 19th century, the advent of photographic technologies triggered a profound reevaluation within the realm of visual art, fundamentally challenging the utility and role of traditional painting. With the introduction of the Daguerreotype process by Louis Daguerre in 1839, a photographic method capable of capturing high-resolution static images, artists confronted an existential challenge: for the first time, a mechanistic process could replicate what had been exclusive to the realm of human skill and interpretation [10]. This sentiment was encapsulated by Paul Delaroche, a notable French painter of

the early 19th century, who dramatically declared, “From today, painting is dead” [27].

While photography rapidly gained acceptance as a distinct artistic medium, its influence engendered a transformative impact on existing art forms, necessitating a radical reconceptualization of artistic purpose and technique. One might posit that this technological intrusion catalyzed the emergence of what is contemporarily identified as modern art—a genre characterized by the disengagement from mimetic representation in favor of abstraction, subjectivity, and conceptual exploration. Vincent van Gogh, a seminal figure in modern art, encapsulated this evolving paradigm when he stated, “Accurate drawing is not the essential thing to aim at, because the reflection of reality in a mirror would not be a picture at all, no more than a photograph” [27].

Thus, the advent of photography not only established itself as an independent artistic medium but also instigated a metamorphosis within traditional art forms, redefining the parameters of representation and prompting innovative aesthetic explorations. This nexus between technology and art warrants a comprehensive academic inquiry, as it offers a fertile ground for understanding how technological advancements perennially shape, challenge, and redefine artistic practice.

2.1.5 Computational Art

In the field of computer graphics, researchers and engineers first explored the mechanics of cameras and the application of physical laws to create photorealistic renderings. As they made progress and achieved mastery, they relied on straightforward inspection-based evaluations, comparing their work to photographs taken by a camera[1] . By the 1990s, the question where new frontiers lie started to be important, as research papers at that time began to emphasize a move away from hyperrealism toward stylization.

As such, during the latter half of the 20th century, the emergence of computational technologies inaugurated a transformative epoch in the landscape of artistic creation, establishing the nascent domain of digital art. This inter-disciplinary realm facilitated

[1]Measuring scientific progress in this area was relatively clear-cut using this method [85]

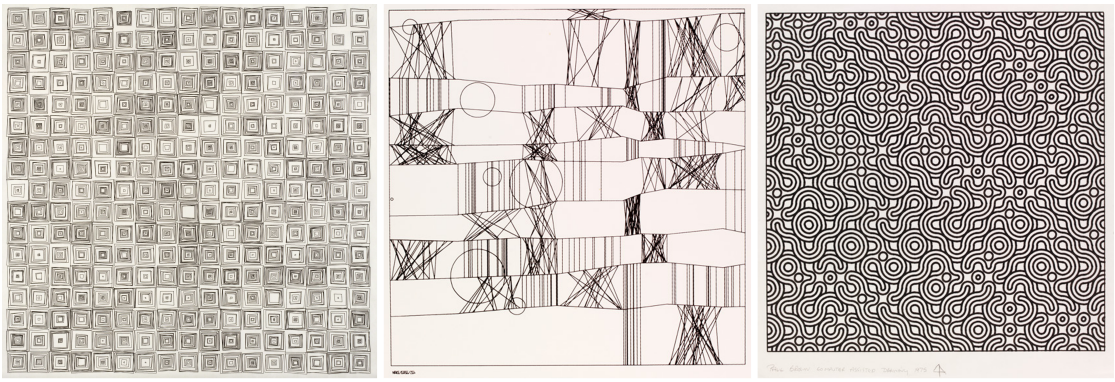


Figure 2.2: Computational Art in the early days: (a) '(Des)Ordres', plotter drawing 1974, Vera Molnar. (b) 'Hommage à Paul Klee 13/9/65 Nr.2', 1965, Frieder Nake. (c) 'Untitled Computer Assisted Drawing', 1975, Paul Brown

the confluence of artists, computer scientists, and engineers, enabling them to employ digital mechanisms as integral components of the artistic process[2] . Commencing in the late 1960s, Vera Molnar emerged as a vanguard in algorithmic artistry, employing computational methods to construct her seminal combinatorial images; notably, in 1968, she utilized a computer to produce her inaugural algorithmic drawings.

The introduction of shape grammars in the early 1970s by George Stiny and James Gips provided another significant leap in the domain of computational art, particularly impacting the fields of architecture and design. Stiny and Gips' seminal paper "Shape Grammars and the Generative Specification of Painting and Sculpture" [84] marked a pivotal moment, proposing a system of rules that could generate a wide variety of designs. These grammars were not just rule-based systems but offered a methodological framework for exploring form generation in a rigorous and systematic manner. Stiny, later in the 1980s joined by Ramesh Krishnamurti, continued to develop these concepts, exploring their applications in architectural composition and design [88, 89].

This exploration of geometric constructs and structural attributes through shape

[2]Refer to pioneering endeavors such as Ben Laposky's "electrical compositions" (1950), Gordon Pask's cybernetic MusiColor (1953), and Jean Tinguely's "Métamatics" series (1955–59), which utilized electronic, kinetic, and robotic mediums.

grammars complemented the early digital art's focus on geometric abstraction. Shape grammars allowed for the creation of complex patterns and structures, demonstrating the computer's potential as a tool for objective visual experimentation and creative design. The work of Stiny and Krishnamurti expanded the methodological arsenal available to artists and designers, reflecting the diversity of approaches in early computational art. Other technologies whose limitations impacted the output include the pen plotter, whose innate linearity dictated the aesthetics of the resultant works. Furthermore, early practitioners often eschewed representational elements to focus on the exploration of pure visual forms, conceiving the computer as an autonomous apparatus capable of facilitating objective visual experimentation.

Frieder Nake's 1965 work, "Hommage à Paul Klee 13/9/65 Nr.2," epitomizes this complexity, serving as an illustrative exemplar of early algorithmic art. Nake employed an intricate algorithm, guided by a predetermined set of instructions, to create a screenprint that drew inspiration from Paul Klee's 1929 oil painting, "Highroads and Byroads." Similarly, Paul Brown's work in the 1970s leveraged computer-generated imagery, utilizing elemental constructs that evolved in accordance with algorithmic rulesets [91], as demonstrated in Figure 2.2.

Thus, the advent of computational technologies not only expanded the methodological repertoire available to artists but also engendered a paradigmatic shift in artistic philosophy and practice, warranting a comprehensive scholarly analysis of the intricate interplay between technological innovation and artistic evolution.

2.1.6 Artificial Intelligence in Art

More recently, the advent of Artificial intelligence gave birth to a current of artists that use data-driven approaches and AI algorithms to generate new form of arts never seen before. Specifically, the technological breakthrough of Generative Adversarial Networks (GANs) in the early-to-mid 2010s [15], enabled for the first time in the AI development to generate new high-quality images that approximate the distribution of training images. Albeit GANs are known for their instability training problems [70, 47], it

(a) (b) (c) (d)

Figure 2.3: AI-generated art: (a) 'Memories of Passersby I', 2018, Mario Klingemann. (b) 'Myriad (Tulips)', 2018, Anna Ridler. (c) 'Electric Fan', 2017, Tom White. (d) 'Neural Zoo', 2018, Sofia Crespo

is a relatively easy tool to generate unseen images, even for non computer scientists. In this context, many tinkerers and artists started embracing such tools for their digital creations. For instance, Mario Klingemann, an award-winning artist, uses, between other methods, GANs to interrogate aesthetic theories. Anna Ridler creates new and unusual narratives working with collections of information of data and self generated datasets. Tom White's artwork investigates how machines see and articulate the world, generating physical abstract prints that are reliably classified by neural networks. Sofia Crespo is interested in the intersection of biological process and machine learning, see Figure 2.3.

Generative machine learning methods have emerged as powerful tools for generating novel imagery based on training datasets. Yet, these technological advancements bring forth an ethical debate surrounding artistic agency in this process. While the question of where the artist's agency lies within the utilization of GANs remains complex, a technical limitation of GANs is notably apparent: limited controllability.

In a GAN, the generator component, denoted as G, is trained to produce samples that mirror the distribution of the real training data. It accomplishes this by transforming random Gaussian noise into samples that appear as though they are part of the actual data distribution. However, due to the inherent randomness of the noise and the generator's training to accommodate any such noise, controlling the specifics of the generated

output becomes a challenge. While there are efforts in the field to offer greater control over the output [59, 37, 5], artists using GANs typically engage in a meticulous process of selection and curation from potentially hundreds or thousands of generated images.

This shift in methodology marks a significant departure from traditional artistic processes. Whereas painters, writers, or sculptors exercise complete control over their creations, the role of artists in the realm of AI art transforms significantly. In this new paradigm, artists are tasked with curating data for dataset creation, training algorithms, and then selecting and refining the algorithm-generated outputs. The moment of creation, therefore, is no longer under the full control of the artist.

These technological breakthroughs continuously reshape the established roles of artists in relation to their work, acting as catalysts for the emergence of unprecedented forms of art. They provoke a reevaluation of the creative process, opening up discussions about the evolving nature of art in the age of AI.

In recent developments, the advent of diffusion-based models[3] [30, 79, 81, 82] has challenged the longstanding dominance of GANs in the realm of image synthesis. These innovative models, combined to text-to-image guidance, train a generative network, which adeptly transitions from a Gaussian distribution to a distribution of images derived from text inputs [95]. This approach particularly excels in terms of quality and resolution. An example of this (the text prompt that generated the image is detailed in the figure's caption) is shown in Figure 2.4.

Notable entities such as Midjourney and OpenAI have unveiled their own diffusion models [67, 66], rapidly gaining popularity as tools for generating images from textual prompts. A critical aspect of these models is their training on extensive image databases sourced from the internet, often through scraping methods. This comprehensive training set, which includes the works of numerous artists, democratizes the creation of images in specific artistic styles, raising ethical concerns about the use of artists' original works

[3]Diffusion models are probabilistic generative models that work by sequentially injecting noise into data points and then learning the reverse process to generate new, unseen samples.

Figure 2.4: "Prehistoric paleolithic cave painting, depicting hunting and spiritual rituals, found in Africa." Image generated by using Diffusion models.

in training datasets.

While the generated images boast impressive quality, the integration of such a vast array of data presents challenges, particularly regarding the signal-to-noise ratio, which is pivotal for empirical validation. The potential for misuse is exemplified by the creation of deepfakes[4] , which are highly realistic fabrications or manipulations of media content. These deceptive media artifacts introduce significant complexities, potentially undermining the credibility and integrity of digital content [68].

One notable advantage of diffusion models over GANs is their inherent flexibility and higher—textual—control; they can generate a wide array of content and styles without the need for retraining or fine-tuning. However, their widespread adoption has been more pronounced among the general public than within the artist community. This trend underscores a shift in the landscape of digital art creation, influenced by the evolving capabilities of AI technologies.

While generative pixel-based models serve as valuable tools for artists, two primary

[4]Deepfakes refer to hyper-realistic digital falsification of images or videos, typically using AI techniques like deep learning, to create or manipulate media content. They can be so convincing that it becomes difficult to distinguish them from genuine content, raising serious ethical concerns in terms of misinformation and misrepresentation.

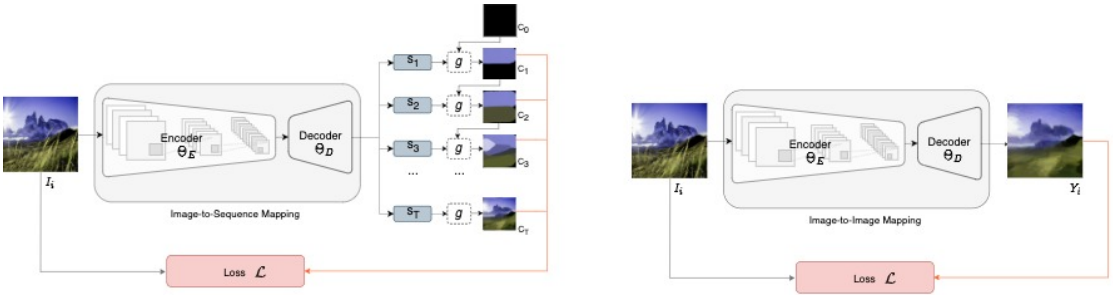


Figure 2.5: Architecture difference between a parametric model (left) and a pixel-based model (right). The former produces a sequence of brushstroke parameters that are rendered, in order, onto the canvas using a differentiable rendering engine G. Pixel-based models generate the matrix of pixels at once.

challenges persist in the debate regarding the artistic merit of their outputs. The first challenge is the level of control over the generated image, which encompasses the processes of gathering, selecting, and curating training data, besides the generating process itself. The second is the inherent nature of these models to generate images by estimating the value of each pixel simultaneously, rather than through a more natural, sequential process.

A potential solution to this second challenge lies in the use of vector-based (or parametric-based) models. Unlike traditional machine learning algorithms that predict pixel values for an entire image—a process involving over a million predictions for a typical medium-low resolution, 3-channel RGB image—vector-based models focus on learning the parameters of vectors that define groups of pixels, akin to brushstrokes. This approach mirrors the human artistic process, which is inherently sequential and based on the cumulative addition of strokes or segments while constantly referencing the evolving state of the canvas. Each brushstroke vector in these models represents a geometric primitive[5] , and the number of parameters involved is generally in the order

[5]In computer graphics and CAD systems, a geometric primitive, or simply primitive, is the simplest geometric shape that a system can handle. Modern computer graphics systems use curves or shapes (boxes, polygons, etc.) as primitives. A set of geometric primitives is based on the dimension being represented [65] such as points, lines, planes or volumes. Thus, vector data

of tens, significantly reducing the computational complexity compared to pixel-based approaches. Figure 2.5 provides a schematic comparison of these two frameworks. The efficiency of machine learning models is crucial for their trainability, scalability, and deployment. By choosing to work with geometric primitives, we align more closely with the logical and natural process of human art creation.

Our focus is not merely to develop a tool for artists or delve into the realm of pixel-based art creation. Instead, this dissertation aims to conduct a technical exploration of parametric algorithms for artistic outputs. By doing so, we seek to define the limitations and capabilities of state-of-the-art algorithms, providing insight into the artistic relevance of current technological advancements. This investigation offers a scientific perspective on the creative potential of algorithms and examines the extent to which AI can be considered "creative" or intelligent, beyond the hype surrounding it.

2.1.7 Parametric-Based Models

Within the landscape of generative models, a specialized subclass known as parameterized generative models has emerged, delineating a fundamentally divergent methodology for synthesizing images. In opposition to pixel-centric models, parameterized generative frameworks produce an ordered sequence of parameters, which are subsequently algorithmically rendered onto an initial blank canvas to instantiate the final image. These models are engineered to simulate a human painting process, iteratively applying a temporally-related series of graphical primitives to construct a composite visual representation.

One salient advantage of parameter-based models is their ability to provide a more interpretable and manipulable set of underlying features. They facilitate an explicit, granular representation of the image in terms of its constituent strokes and chromatic

structures typically represent geometric primitives using a strategic sample, organized in structures that facilitate the software to interpolate the remainder of the shape at the time of analysis or display, using the algorithms of computational geometry [90]

elements, thereby granting enhanced control over the resultant visual output. Furthermore, this parameter-driven paradigm frequently manifests superior computational efficiency by virtue of its capacity to concentrate computational resources on high-level attributes as opposed to individual pixels. This extends to the dimensionality of the parameter space that a neural network is required to predict, often resulting in a more computationally tractable problem.

In the rest of this chapter, we first provide a literature review on this topic, and then explain key concepts, methods, components, and terms crucial for understanding the subsequent chapters will be defined and explained, setting the stage for a deeper dive into the intricacies of parametric-based models in the context of art generation.

2.2 Literature Review

This section aims to introduce the field of research of this thesis, and gather a substantial background repository of works that have facilitated the progress and evolution of computer vision and computer graphics models for artistic outputs.

2.2.1 Introduction to Non-Photorealistic Rendering

The field of Non-Photorealistic Rendering (NPR) represents a significant paradigm shift from the pursuit of photorealism in computer graphics and image processing. Unlike photorealism's focus on replicating the precision of a camera lens, NPR is concerned with a diversity of representational styles that extend beyond the constraints of photographic accuracy, such as stylization and abstraction, tailoring visual communication to convey specific meanings more effectively than photorealistic images, and exploring the synergy between visual art and language. This field traces its roots to pre-photorealistic eras, drawing inspiration from artistic styles prevalent in ancient Egyptian art and European town drawings from the 16th century [85].

In terms of algorithmic approaches, NPR algorithms vary considerably based on their intended application and the style they aim to simulate. These algorithms com-

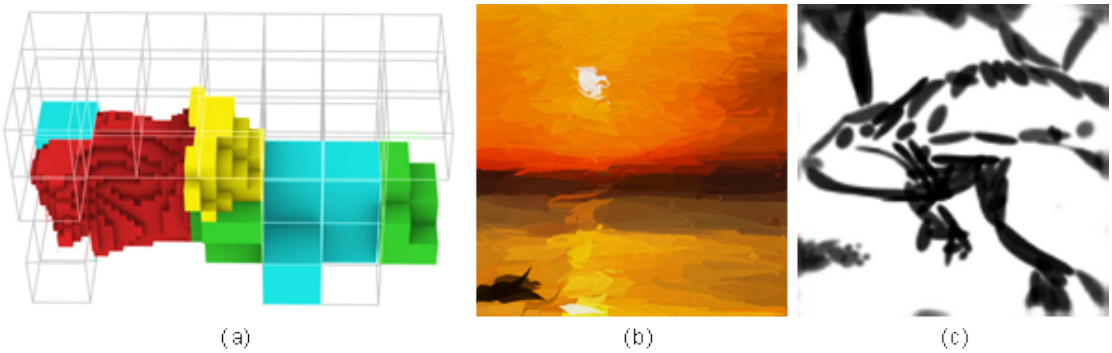


Figure 2.6: Examples of Non-photorealistic renderings (NPR): (a) An NPR used for 3D modeling in engineering and architecture design. (b) An NPR of an existing 2D landscape image using a learning-based algorithm. (c) An NPR of an abstracted chameleon using a line drawing using an optimization algorithm.

monly categorize rendition artifacts into two broad types: image artifacts and model artifacts. The choice of artifacts and their implementation in rendering algorithms are heavily influenced by the data source, which can range from simple line drawings to complex three-dimensional models [85, 93]. Image artifacts, also called Image Abstraction and Artistic Stylization (IA-AS) in NPR, typically involve stylizations that modify the appearance of two-dimensional images to resemble various artistic techniques, such as watercolor, sketching, or impressionist painting. These artifacts often focus on the manipulation of colors, textures, and stroke patterns to achieve the desired artistic effect [49]. Model artifacts, on the other hand, are employed in rendering three-dimensional models with non-photorealistic techniques. This involves using geometric primitives and lighting models that differ from those used in photorealistic rendering. The aim is to create 3D representations that evoke the aesthetics of traditional artistic media, like clay sculptures or paper models [24].

IA-AS has seen significant advancements post-1990s, notably in areas of visual communication and stroke-based rendering. IA-AS is devoted to the artistic transformation of 2D images into abstract and stylized forms, thereby influencing areas like computer graphics and multimedia extensively. Some techniques involve artistic stylization, image abstraction, line drawing, pen-and-ink illustration, pencil drawing, en-

graving, or HDR among others. A variety of filtering techniques in NPR, including but not limited to Anisotropic, Kuwahara, and Flow-based difference of Gaussian filters, have expanded NPR's scope and effectiveness [49].

Some notable contributions are done by researchers such as Haeberli et al. [21], Hertzmann [24, 28, 29, 25, 26], DeCarlo et al. [6], Winnemoller et al. [94], and Kang et al. [41], as found in esteemed publications. NPR methodologies are bifurcated into low-level approaches, involving non-linear filters and stroke placements, and global-level approaches, which include linear filters and depth map calculations. The evolution of NPR research has been inclining towards GPU-based IA-AS since 2013, prioritizing faster processing of images and videos over traditional CPU methods [49].

NPR's algorithmic approach, therefore, is not merely a technical endeavor but an artistic one, requiring a careful balance between computational methods and aesthetic judgment. The algorithms must be tailored to capture the essence of the artistic style they are emulating, whether it's the fluidity of a watercolor wash or the bold lines of a sketch, while ensuring that the rendered output effectively communicates the intended message or emotion [85].

Stroke-Based Rendering

Stroke-based rendering (SBR) is a subset of IA-AS, and by extension NPR, that focuses on painting-like outputs, creating paintings by placing discrete strokes on a canvas [24]. Traditionally, NPR focused on procedural heuristic algorithms to simulate such painting process [51].

Painterly rendering in an SBR setting, involves the strategic placement of stroke primitives on a digital canvas. This process is sequential and mirrors traditional artistic methods, where artists apply strokes in varying orders and techniques to evoke specific styles and focus the viewer's attention. The selection of stroke parameters, such as color, shape, size, and placement, plays a crucial role. By avoiding excessive detail, artists stimulate the viewer's imagination, inviting them to mentally fill in the gaps [80, 23]. This is a deliberate choice, balancing the trade-off between detail and abstraction

to engage the viewer's interpretive faculties.

In computational terms, the highest level of detail would correspond to rendering each individual pixel. However, artists often opt for a lower level of detail to achieve a higher degree of abstraction. This approach makes the artwork open to interpretation and scrutiny, a dynamic that parametric algorithms can mimic. Such methods can be adjusted to vary the level of fidelity in their output.

The sequential nature of SBR algorithms contrasts with some learning-based methods that generate art by learning a combination of pixels in a non-sequential manner, such as GANs and other style transfer-like algorithms [8, 37, 98, 48, 50][6] . However, there are overlapping elements between SBR and generative algorithms. For instance, while GANs typically produce an image (CxHxW pixels) that closely matches the distribution of training images, SBR algorithms also learn from a distribution of images. The key difference lies in their output format: SBR algorithms output in parameter space (NxK, where N is the number of strokes and K is the number of parameters defining the painting primitive), rather than pixel space.

2.2.2 Non-Learning-Based SBR Methods

Non-Learning-based methods are those computational methods that do not use ML nor train neural networks to learn from a database at their core. That is, there are no weight parameters that are being learned[7] . Within this group, we can further distinguish two main categories: the first one is based on heuristics and hand-design rules, and became popular in the 90s. The second one is based on optimization emerged and in the early

[6]For instance, the popular Pix2Pix method [37] is a conditional adversarial network for image-to-image translation. It uses paired data to train a generator for creating images that a discriminator then evaluates, learning to produce results that closely resemble target domain images from given inputs.

[7]Some non-learning-based algorithms do include already trained neural networks as part of their pipelines. However, their role is secondary, and in any case there is no data-driven approach under this category.

Figure 2.7: Painterly renderings with method [29] (a), [28] (b) and [21] (c). Original images are shown as insets.

2000s.

Procedural SBR Techniques

Procedural techniques in SBR are characterized by their user-driven and rule-based nature, often requiring manual interaction for artistic creation. The earliest painterly stylization algorithms applied hand-designed rules to produce impressionistic effects [21, 53, 29]. These methods generally point sample the image and compute control points of the brush stroke model. Generally, parameters such as stroke width are fixed in advance, and color is selected from the reference image based on the location of the first control point of the brush stroke.

One of the oldest work "Paint by Numbers" by Haeberli [21] in 1990, is an interactive paint-by-number system in which an user places strokes on a canvas by clicking with a mouse. The direction of the stroke can be fixed, or it can be guided dragging the mouse and releasing the button. In this work, Haeberli also indicates different ways to control the shape of the stroke, guided by the gradients of either a secondary image or the same reference image.

The geograftals editing system proposed by Kaplan et al. [43] is an exemplary instance, offering functionalities like geograftals generation, editing, shading, and line

drawing for complex scenes. This is complemented by the work of Salisbury et al. [72], who introduced an interactive system that empowers users to generate textures through a mouse interface, thus simplifying the creative process. Similarly, Hertzmann [29] developed a technique using the reference image's gradients to guide cubic B-spline for curved brush strokes to produce images with a hand-painted aesthetic, effectively mitigating the anti-aliasing effect. This algorithm implements a coarse-to-fine method by scanning the image with steps sizes, which correspond to stroke widths, and by extension, the amount of detail to paint at each level. Figure 2.7 (b) uses this type of algorithm[8] .

Salesin et al. [71] further proposed an orientable texture system for pen-and-ink illustration of grayscale images. Though limited to grayscale imagery, this system marks an interactive leap in texture transfer and brush stroke application. In the realm of caricature, Gooch et al. [14] focused on facial illustrations using an image-wrapping technique that accentuates facial features for applications in entertainment and education.

Jens Grubert et al. [18] explored the potential of WIMP interface-based interactive SBR, enhancing the editing facilities during the rendering process. Santella et al. [73] introduced an innovative approach using eye-tracking data to create abstracted painterly renderings, leveraging human perceptual data. Kang et al. [42] focused on interactive sketch generation using B-spline and live wire contour paradigms, allowing increased detailing in user interactions. O'Donovan and Hertzmann [64] proposed the AniPaint system, notable for its use of a WYSIWYG interface in animating individual video frames.

[8] I re-implemented this algorithm in Python thanks to discussions with Aaron Hertzmann and his original code in Java. Code available in github.com/manuelladron/painterPython.

Optimization Methods

Optimization-based SBR techniques rely more on algorithmic and mathematical models for automation and enhancement in rendering, and there is an iteration procedure to search for an "optimal" solution[9] . Here, the rendering procedure has to be differentiable to allow the backpropagation of the gradients of the error of a loss function to the parameters that are being optimized. In general, we formulate this problem as:

$$\mathcal{L} : \mathbf{x}^* = \text{argmin}_{\hat{\mathbf{x}}} \mathcal{L}(\hat{I}, I) \tag{2.1}$$

[9]The term "optimal" in optimization algorithms depends on several factors, including the specific characteristics of the optimization problem and the algorithm itself. Some key considerations are:

- Types of Optimality:
  - Global Optima: In some cases, it is possible to find a solution that is provably the best possible, known as a global optimum. This is often possible for convex problems, where any local minimum is also a global minimum.
  - Local Optima: In non-convex problems, local optima can exist, which are points where the solution is better than all other solutions in a nearby region, but not necessarily the best possible solution over the entire domain.
- Complexity and Landscape:
  - Simple Landscapes: Problems like convex optimization problems have a "simple" landscape with a single global minimum. For such problems, achieving the global optimum is indeed feasible.
  - Complex Landscapes: For many real-world problems, especially those encountered in machine learning, the landscape is high-dimensional and non-convex, with many local minima and saddle points. Here, it's often not feasible to find a global minimum, so algorithms aim for a "good enough" local minimum.

Where $\mathcal{L}$ is a loss function that measures the similarity between a canvas $\hat{I}$ and a reference image I, and $\hat{\mathbf{x}}$ are the brushstroke parameters to be optimized. We use Stochastic Gradient Descent(SDG)[10] to update the stroke parameters as follows:

$$\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} - \mu \frac{\partial \mathcal{L}(\hat{I}, I)}{\partial \hat{\mathbf{x}}} \tag{2.2}$$

where μ is a regularization parameter that adjusts the weight of the gradient.

Aaron Hertzmann's early work in optimization-based painting utilized a method known as "Paint by Relaxation" [28]. This method involves precise stroke placements, but it relied on somewhat cumbersome optimization heuristics (see Figure 2.7). The

[10]Stochastic Gradient Descent is an optimization algorithm that is commonly used in machine learning to update and fine-tune the model's parameters. The goal is to minimize a function (often called the "loss function") that represents the difference between the model's predictions and the actual data. Here's a brief overview:

Stochastic Gradient Descent (SGD): This is one of the simplest, yet effective, optimization algorithms. Unlike classic gradient descent, which uses the entire dataset to compute the gradient at each step, SGD randomly selects a subset of the data at each step. This makes SGD faster but a bit more erratic, as the smaller subset may not fully represent the data's overall shape.

An improved algorithm is called Adam (Adaptive Moment Estimation): Adam is an extension of SGD that incorporates concepts from other optimization algorithms for more efficient and robust learning. It maintains separate learning rates for each parameter and adjusts them during training, making it well-suited for problems with noisy or sparse data.

Both algorithms are critical tools in the training of machine learning models, each with its own set of advantages and drawbacks.

In machine learning, the learning rate is a crucial hyperparameter that influences how quickly or slowly a model learns from the data. Technically, it determines the size of the steps the model takes during the optimization process — specifically, when updating the weights in the network through algorithms like gradient descent. A higher learning rate means the model updates its weights more significantly with each iteration, allowing it to learn faster but risking overshooting the optimal solution. Conversely, a lower learning rate ensures more gradual updates and a more meticulous approach to finding the best solution, but it may lead to slower convergence and can get stuck in local minima. Balancing the learning rate is key to training effective models.

technique primarily focused on creating a painterly effect by placing brush strokes in a way that approximates the underlying image. The optimization process iteratively adjusted the strokes' positions, colors, and orientations to minimize the difference between the rendered image and the source image. Despite its effectiveness in creating visually appealing results, the method required considerable computational effort and fine-tuning.

Expectation-Maximization (EM)-like packing algorithms have been used for non-overlapping stroke primitives [24, 69], such as stipples [75] and tile arrangements [45]. They function by iteratively improving the placement of strokes to optimize coverage and distribution without overlap. The EM-like approach typically starts with an initial random placement of strokes and then refines their positions based on the underlying image features and the desired artistic effect. For instance, [75] is an approach for weighted Voronoi stippling. This method uses an adaptation of Lloyd's algorithm, a type of EM algorithm, for distributing stipples across an image. The stipples are placed in a way that their density corresponds to the darkness of the image, creating a stippled representation that closely resembles the source. This method is particularly effective for creating images that resemble traditional stipple art. Kim et al. [45] focus on arranging various shaped tiles to recreate an image. This approach can be seen as a type of mosaic art creation, where the optimization process involves fitting tiles of different shapes and colors in a way that collectively they resemble the source image.

A variety of related methods can be used to stylize 3D models, e.g., see [17, 3]. Further advancing the field, in the work by Tamás Szirányi et al. [86], the focus was on optimizing paintbrush rendering using dynamic Markov Chain Monte Carlo (MCMC) methods. This approach involves using probabilistic models to determine the placement and characteristics of brush strokes. The dynamic aspect of this method allows the system to adapt the stroke placement iteratively, based on the evolving state of the rendered image. MCMC methods are particularly suited for complex optimization problems where the solution space is vast and not easily navigable with simpler algorithms.

Semmo et al. [76] developed an oil paint filtering system for image abstraction and

stylization, incorporating color correction and non-linear filtering to preserve global color features and dominant structures. In this work, optimization is used to quantize the chrominance and luminance channels.

A few recent methods have explored direct stroke parameter optimization. DiffVG [52] is a method focusing on optimizing vector graphics through gradient-based optimization. The key innovation of DiffVG is its ability to optimize the parameters of vector graphics (like paths and shapes) in a differentiable manner, allowing for gradient descent techniques to be applied directly to vector graphic elements. This approach is particularly useful for generating high-quality, scalable vector graphic images. By optimizing the stroke arrangement, DiffVG can create detailed and precise illustrations with a high degree of control over the final output. Stylized Neural Painters (SNP) [99] directly optimize the stroke arrangement with gradient-based optimization. SNP represented the state-of-the-art of stroke optimization until 2023. Unlike methods that use neural networks for determining stroke placement, SNP starts with a pseudo-random initialization of a set number of strokes. These strokes are then optimized iteratively, allowing for significant control over the style and texture of the final image. SNP can generate a wide variety of styles by adjusting stroke parameterization and textures. The method is capable of producing images with thousands of strokes, ranging from impressionistic effects to highly detailed images using numerous small strokes.

CLIPDraw [11] uses a text-based prompt instead of a target image for generating drawings. This method leverages the CLIP (Contrastive Language–Image Pretraining) model to interpret the text prompt and guide the drawing process. The unique aspect of CLIPDraw is its ability to create images that are closely aligned with the semantic content of the text, demonstrating a novel integration of language understanding and image generation.

Painterly stylization is a high-dimensional optimization problem, with challenging local minima, and each of these of methods may struggle to efficiently achieve precise results with a small number of strokes. State-of-the-art (SOTA) techniques employ strategies that alleviate the difficulties of optimizing strokes, such as sequentially work-

ing on small overlapping patches instead of finding the stroke parameters that produce the whole painting. Moreover, direct optimization can be slow for each output image.

2.2.3 Neural-Based SBR Methods

In neural-based approaches to stroke-based rendering, a network is tailored to synthesize a series of brushstrokes in response to an input image. These methods involve the optimization of the neural network's weights, which is guided by gradient feedback derived from the discrepancy between the network's output and a reference photograph. Such a discrepancy is quantified by a loss function, often utilizing perceptual losses [96] instead of conventional pixel-wise losses like L_1 or L_2[11] . Most of the work found in the literature does not use stroke-based supervision. This is primarily due to the complexity and cost associated with creating datasets that pair images with corresponding sets of stroke parameters.

Typically, neural methodologies integrate either a pretrained neural renderer or a non-neural differentiable renderer, establishing an end-to-end differentiable pipeline. This setup allows for the effective application of advanced loss functions, such as those based on perceptual similarity. A key advantage of deep-learning-based approaches lies in their ability to generalize to novel images, unseen during the training phase. This is in stark contrast to stroke optimization methods, which necessitate the determination of a new set of parameters for each distinct painting. This process can be time-consuming, often taking from several seconds to minutes per image. In comparison, deep-learning methods can readily adapt to new images, particularly if they share characteristics or distributions similar to those in the training set, without the need for additional training or fine-tuning.

[11] For a further description of losses, refer to Section 2.3.2

Reinforcement Learning

Within neural methods, Reinforcement Learning (RL) algorithms have been instrumental in training painting agents, as evidenced by several notable works. These agents, as detailed in papers such as by Ganin et al. [12], Mellor et al. [58], and Huang et al. [36], generate sequences of strokes through interactions with a critic network. The RL framework provides a learning signal in the form of rewards, often utilizing adversarial learning concepts [16] to train the critic network.

Among these methods, certain approaches, like those by Huang et al. [36] and Singh et al. [77], concentrate on achieving precise reconstructions of reference images. Huang et al.'s method, although capable of accurate image reconstruction, relies on the generation of thousands of minute strokes, resulting in a style that, while precise, is somewhat limited in artistic diversity. However, Schaldenbrand and Oh [74] have demonstrated variations of this style, particularly for applications involving robotic arms.

Another intriguing approach is LpaintB [39], which combines RL with self-supervised learning. This method tends to produce coarser but still recognizably similar versions of the input images. On a similar note, Singh and Zheng's work [77] employs a semantic guidance pipeline to distinguish between foreground and background, slightly refining the quality of image reconstructions. However, like Huang et al.'s method, it also tends towards precise depiction rather than artistic stylization.

A subset of RL methods have shown promise in creating more artistic abstractions. One such notable method is SPIRAL, introduced by Ganin et al. [12]. SPIRAL employs adversarially-trained actor-critic algorithms, mastering both unconditional and conditional generation and reconstruction tasks. Initially demonstrated on datasets like MNIST and OMNIGLOT, SPIRAL showed proficiency in these simpler domains but encountered challenges with more complex datasets such as CelebA faces, leading to less precise, somewhat blurry results.

Building upon SPIRAL, SPIRAL++ [58] presented significant enhancements. This improved version achieved a remarkable spectrum of abstracted image styles, pushing the boundaries of what is possible in artistic abstraction using RL. SPIRAL++ leveraged

advancements in loss function optimization, shifting from the WGAN-GP [19] used in SPIRAL to a standard GAN loss with spectral normalization [61]. This shift enabled SPIRAL++ to create abstractions and stylized representations, particularly in the challenging CelebA dataset. Despite these advancements, SPIRAL++ still faced limitations in terms of user control. The method provides limited interpretability and control over the style and level of precision in the reconstructions, posing a challenge for applications requiring precise or customizable artistic outputs.

Deep Learning

Deep learning has significantly contributed to advancements in stroke-based rendering, with several methods demonstrating varying capabilities in terms of style, precision, and generalization. The next paragraphs describe different methods within a deep learning approach.

CNN and Transformer Methods Paint Transformer, proposed by Liu et al. [54], exemplifies a novel approach by modeling the painting process as a stroke prediction problem. Utilizing a combination of Convolutional Neural Networks (CNNs) and Transformers, this method operates without reliance on pre-existing datasets, showcasing remarkable generalization capabilities. Both Stylized Neural Painters (SNP) and Paint Transformer tend towards a texturized painting appearance, somewhat sacrificing the precision of depictions. These methods, however, do not address stylistic control beyond the applied textures and pixel-based stylization (after the optimization process is done), nor do they explore abstractions within the limitations of their optimization approaches.

Recurrent Neural Network (RNN)-Based Methods In contrast to CNN and Transformer-based models, RNN-based methods have been employed to learn painting or sketching styles. One such instance is *Sketch-RNN* by Ha et al. [20], which utilizes a sequence-to-sequence variational autoencoder [46]. This model employs a bidirectional RNN as

an encoder and an autoregressive RNN as a decoder, effectively handling simple line drawings but not extending to RGB images. Similarly, StrokeNet [97] combines a CNN encoder with an RNN decoder, also focusing on line drawings and sketches, yet falling short in generalizing to color images. Mo et al. [62] further develop this concept, introducing a dynamic window for vector line drawings, using a CNN encoder and RNN decoder framework.

Image Translation and Style Variation Other methods in image translation, such as those proposed by Isola et al. [37] and Zhu et al. [98], train with painting images to achieve similar outputs. However, these techniques fundamentally differ as they operate in pixel space, in contrast to the stroke space approach adopted in other methods.

Alternative Approaches Diverging from RL and RNN-based methods, Nakano [63] employs pretrained classification CNNs on ImageNet to optimize a set of strokes. Using a differentiable renderer, this approach seeks to activate specific neurons in the network, painting images that maximize this activation.

Trade-offs and Thesis Experiment The body of work in this domain suggests an inherent trade-off between accurate depiction and stylistic variation. Methods achieving visual abstraction often fail to attain high precision, and vice versa. This thesis will demonstrate how, with an appropriate combination of loss functions and optimization strategies, it is possible to achieve precise reconstructions with fewer strokes while also allowing controllable levels of artistic styles. This is achieved without stroke supervision or reliance on external stylistic images, as commonly seen in style transfer algorithms [13, 48, 98, 55]. Furthermore, the thesis will show that gaining control over this complex painting process does not necessitate the use of highly complex models or intricate training procedures like RL.

2.3 Foundations for Parametric-Based Models

This section provides the theoretical foundations of parametric-based models, important for understanding the experiments and methodologies discussed in this thesis. Each experiment presented herein is grounded in these foundational concepts, ensuring a coherent and unified approach across different applications.

2.3.1 The SBR Model

Stroke-Based Rendering (SBR) algorithms, central to this thesis, comprise several key components:

- **Dataset or Target Image**: In data-driven approaches, such as those employing deep learning, a dataset is essential for training algorithms to generalize to new paintings within a given domain. Conversely, optimization algorithms typically require only a single target image per execution, as they operate independently for each image intended for stylization.

- **Stroke Primitive Parameterization Model**: This mathematical model defines the characteristics of brushstrokes. It encompasses various primitives, including lines, n-Bézier curves, n-splines, and geometric shapes like circles, triangles, or squares. Figure 2.8 illustrates the control points of a quadratic Bézier curve. Often, a combination of primitives is employed to render diverse stroke shapes. Chapters 3, 4, and 5 will utilize a quadratic Bézier curve as the stroke model, since the shape of these curves can be parameterized using just six coordinates: the x,y coordinates of the start, middle, and end control points. Some relevant previous works in the literature employ either quadratic Bézier curves or straight curves, which are parameterized by a center point and width and length parameters. I have found that the quadratic Bézier parameterization is sufficient for capturing the details of a painting. It offers more flexibility than straight curves, and its nature is more akin to human strokes.

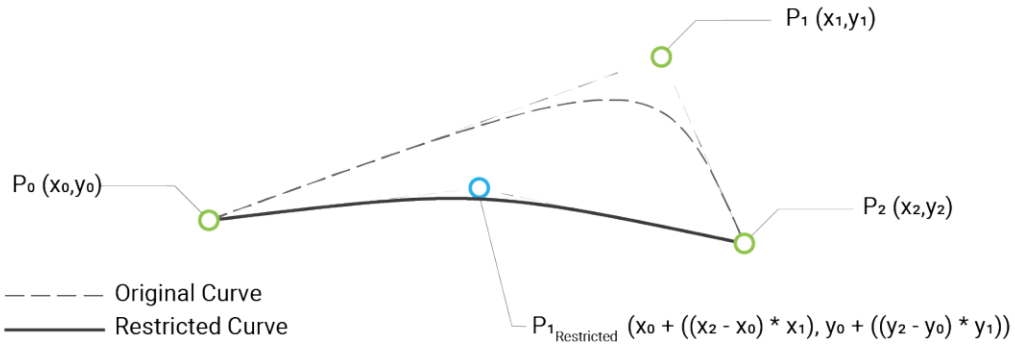


Figure 2.8: Stroke modeled as a quadratic Bézier curve

- **Canvas Model**: The canvas is conceptualized as a matrix where strokes are sequentially rendered, updating its state with each new stroke. Typically initialized with a uniform background color, the canvas registers the evolution of the painting process.

- **Rendering Module**: Rendering converts vector information into 2D images. Traditional rendering frameworks use rasterization, which is generally non-differentiable. This non-differentiability poses challenges for gradient-based methods. Implementing the renderer as a neural network, however, facilitates differentiability, enabling gradient backpropagation and integration within a deep learning framework.

- **Controller Function**: This function orchestrates the translation from the input photograph to the output painting. Ranging from deterministic to stochastic, the controller defines the algorithm's approach. It may be modeled as a neural network or a set of hand-crafted equations, among other possibilities.

These elements are the common denominators in SBR algorithms. A more detailed exploration of the painting functions follows, as they are paramount in the SBR process. These functions can be broadly categorized into heuristic-based, optimization-based,

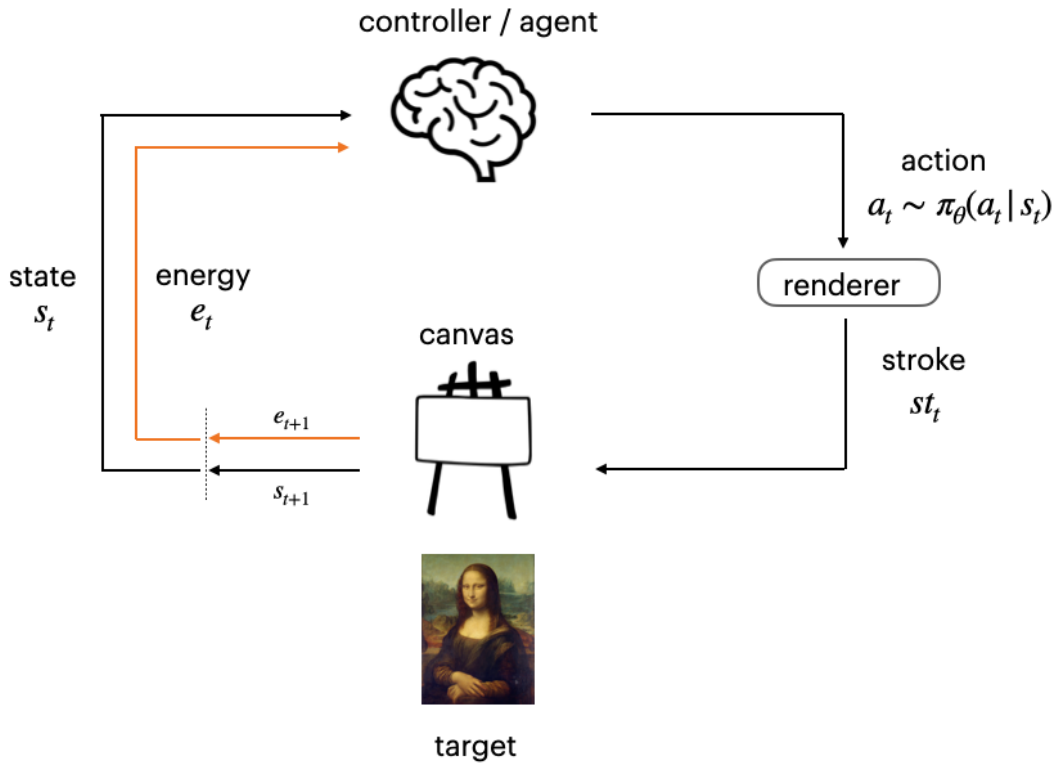


Figure 2.9: Diagram of an SBR algorithm with four main components: controller or agent, renderer, a canvas and a stroke model.

and learning-based (AI) methods. Figure 2.9 provides a high-level overview of these components. Typically, a controller or agent (often a neural network) outputs an action (predicts stroke parameters) based on the current state of the canvas s_t at time t. The renderer then transforms these parameters into a raster image, which is subsequently compared to a target image. An evaluation function assesses the similarity between the generated canvas and the target, providing feedback to the agent for improved future iterations. However, creating a stylized painting involves more than mere replication of the target image. The ensuing sections delve into the nuances of achieving stylization—a complex and technically demanding process.

2.3.2 Overview of Loss Functions for Gradient Methods

Loss functions are essential in methods that use gradients to update the optimization parameters. Such optimization parameters can be pixels in an image, stroke parameters, or a neural network's weights—as it is the case in deep learning settings. A model tries to minimize such loss (or objective) functions during training. Generally, a combination of different loss functions is more effective than using just one loss function, although this depends on the nature of the problem. Formally, we define an overall loss function $\mathcal{L}$ as a combination of different loss functions, as follows:

$$\mathcal{L} := \beta_1 \mathcal{L}_1 + \beta_2 \mathcal{L}_2 + \cdots + \beta_n \mathcal{L}_n + \ldots \tag{2.3}$$

where β_n controls the weight or importance of each specific loss, balancing the overall total loss $\mathcal{L}$.

Pixel loss. The most common loss in image synthesis is pixel loss, which measures the distance between pixels, commonly in RGB space. The pixel loss is commonly represented by L_1, also known as Least Absolute Deviation and formally, $L_1 = |\hat{x} - x|$. The L_2 function, also known as Least Square Errors or Mean Squared Error (MSE), and formulated as $L_2 = (\hat{x} - x)^2$, is very popular in machine learning, especially in regression models. Both are used as functions to measure image similarity, but it has been empirically proven that L_2 tends to produce blurrier results, often caused by the failure of the assumption than the noise is Gaussian distributed, which works poorly with multimodal distributions [57].

Perceptual Losses. Traditional loss functions such as Mean Squared Error (MSE) or Cross-Entropy are primarily designed to operate directly on the output space of a model. However, in tasks where preserving the texture, content, and overall aesthetic of an image is crucial, such as in image super-resolution, style transfer, or generative modeling, these loss functions fall short. This gap is where perceptual losses [40, 96] come into play, aiming to align the assessment of image quality more closely with human visual

perception.

Perceptual losses leverage the semantic information captured by deep neural networks, typically those trained on extensive image classification tasks. The underlying principle involves extracting features from various convolutional layers of a pre-trained image classification network, such as VGG-16 or ResNet. These extracted features are then compared using norms like L_2, L_1, or cosine similarity, focusing on capturing the high-level content and texture information from the images.

To illustrate, let's consider an application in image super-resolution. In this scenario, we have a generator function G that takes a low-resolution image x and produces a high-resolution counterpart $G(x)$. Our target is a high-resolution reference image y. The perceptual loss, denoted as $\mathcal{L}_{perceptual}$, is formulated as follows:

$$\mathcal{L}_{perceptual}(G(x), y) = ||\phi(G(x)) - \phi(y)||_2^2 \tag{2.4}$$

In this equation, $\phi(\cdot)$ represents a feature extraction function obtained from a pre-trained neural network. This function outputs feature maps from specified layers within the network. The term $||\cdot||_2^2$ signifies the squared L_2 norm. By using this formulation, perceptual loss measures the difference in high-level features (like textures and contours) between the super-resolved image and the target high-resolution image, providing a loss metric that is more attuned to human visual perception compared to traditional pixel-wise losses.

Content and Style Losses. In advanced implementations of perceptual loss, particularly in neural style transfer, the concept is often divided into two distinct components: content loss and style loss.

- **Content Loss**: This component ensures that the high-level features (such as objects and their arrangements) of the generated image resemble those of the content image. It can be mathematically expressed as:

$$\mathcal{L}_{content} = ||\phi(G(x))_{content} - \phi(y)_{content}||_2^2 \tag{2.5}$$

Here, $\phi(\cdot)_{content}$ extracts content-related features from the respective images.

- **Style Loss**: This loss focuses on replicating the stylistic elements (like textures and color patterns) from the style image to the generated image. It is typically computed by comparing the Gram matrices of the feature maps extracted from different layers of the network:

$$\mathcal{L}_{style} = ||G(\phi(G(x))_{style}) - G(\phi(z)_{style})||_F^2 \tag{2.6}$$

  In this equation, $G(\cdot)$ denotes the Gram matrix calculation [34], and $||\cdot||_F^2$ represents the squared Frobenius norm. z is the style reference image.

The perceptual losses, comprising both content and style components, offer several advantages. Primarily, they model a form of human-like perception, producing results that are visually more appealing compared to traditional pixel-wise loss functions like MSE. Additionally, leveraging transfer learning by using layers from pretrained models as feature extractors capitalizes on the sophisticated feature hierarchies learned by these networks, often leading to faster training and improved generalization.

However, these perceptual losses also have limitations. They tend to be more computationally intensive than pixel-wise losses due to the additional forward pass through a pretrained network. Furthermore, the efficacy of perceptual losses can be contingent on the nature of the pretrained network, as these networks are typically trained for specific tasks and on specific datasets. Therefore, their effectiveness might vary when applied to different types of tasks or data domains.

2.4 Conclusion

This chapter started with an exploration of art's evolution through a technological lens, highlighting the dynamic and ever-evolving nature of artistic expression, particularly in the context of modern computational methods and the advent of AI. This historical

perspective sets the stage for understanding the significance and potential of ML models in art creation, which is the central focus of this dissertation.

We delved into the technical aspects of parametric-based models, detailing their critical components such as loss functions and various algorithmic approaches. This foundation is essential for comprehending the subsequent chapters, where we apply these concepts in practical experiments.

The following three chapters are dedicated to addressing three specific challenges in the realm of computational art that remain unresolved: stylization, controllability, and identity preservation. Chapter 3 explores the application of deep learning in capturing and representing artistic style, showcasing the versatility and power of neural networks in artistic contexts. Chapter 4 and 5 investigate how stylization can be further improve by designing a more controllable pipeline within an optimization framework, offering new insights into this crucial component of artistic creation. Finally, Chapter 6 focuses on the challenge of identity preservation in art. Unlike the previous experiments, this chapter introduces different types of primitives and approaches to artistic representation, broadening the scope of our exploration in deep learning-based art generation.

Together, these chapters contribute significantly to the fields of computational art and AI, each addressing a distinct yet interconnected aspect of the artistic creation process.

References

[1] Artlex. *Art Dictionary*. URL: https://%20www.artlex.com/ArtLex/Li.html.

[2] Kenneth Bartlett. *The civilization of the Italian Renaissance: A sourcebook*. University of Toronto Press, 2011.

[3] Pierre Bénard and Aaron Hertzmann. "Line Drawings from 3D Models". In: *Foundations and Trends in Computer Grapics and Vision* 11.1-2 (2019). DOI: 10.1561/0600000075.

[4] Paul A Bennett et al. *Books and Printing; a Treasury for Typophiles*. Good Press, 2021.

[5] Xi Chen et al. "Infogan: Interpretable representation learning by information maximizing generative adversarial nets". In: *Advances in neural information processing systems* 29 (2016).

[6] Doug DeCarlo and Anthony Santella. "Stylization and abstraction of photographs". In: *ACM transactions on graphics (TOG)* 21.3 (2002), pp. 769–776.

[7] Nicholas Eastaugh et al. *Pigment compendium: a dictionary of historical pigments*. Routledge, 2007.

[8] Ahmed Elgammal et al. "CAN: Creative Adversarial Networks Generating "Art" by Learning About Styles and Deviating from Style Norms". In: *ICCC*. 2017.

[9] Encaustic Art Institute. *What is Encaustic?* https://eainm.com/what-is-encaustic/. Accessed: 2023-09-05.

[10] Barnes Foundation. *From Today Painting is Dead: Early Photography in Britain and France*. URL: https://www.barnesfoundation.org/whats-on/early-photography.

[11] Kevin Frans, LB Soros, and Olaf Witkowski. “CLIPDraw: Exploring Text-to-Drawing Synthesis through Language-Image Encoders”. In: *arXiv preprint arXiv:2106.14843* (2021).

[12] Yaroslav Ganin et al. “Synthesizing programs for images using reinforced adversarial learning”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1666–1675.

[13] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. “Image Style Transfer Using Convolutional Neural Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[14] Bruce Gooch, Erik Reinhard, and Amy Gooch. “Human facial illustrations: Creation and psychophysical evaluation”. In: *ACM Transactions on Graphics (TOG)* 23.1 (2004), pp. 27–44.

[15] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).

[16] Ian J. Goodfellow et al. “Generative Adversarial Nets”. In: *Proc. NIPS*. 2014.

[17] Stéphane Grabli et al. “Programmable rendering of line drawing from 3d scenes”. In: *ACM Transactions on Graphics (TOG)* 29.2 (2010), pp. 1–20.

[18] Jens Grubert, Sheelagh Carpendale, and Tobias Isenberg. “Interactive stroke-based NPR using hand postures on large displays”. In: *EPRINTS-BOOK-TITLE*. University of Groningen, Johann Bernoulli Institute for Mathematics and ..., 2008.

[19] Ishaan Gulrajani et al. “Improved training of wasserstein gans”. In: *arXiv preprint arXiv:1704.00028* (2017).

[20] David Ha and Douglas Eck. “A neural representation of sketch drawings”. In: *arXiv preprint arXiv:1704.03477* (2017).

[21] Paul Haeberli. “Paint By Numbers: Abstract Image Representations”. In: *Proc. SIGGRAPH*. 1990.

[22] George M. A. Hanfmann. “Hellenistic Art”. In: *Dumbarton Oaks Papers* 17 (1963), pp. 77–94. ISSN: 00707546. URL: http://www.jstor.org/stable/1291191 (visited on 11/30/2023).

[23] Siddharth Hegde, Christos Gatzidis, and Feng Tian. “Painterly rendering techniques: a state-of-the-art review of current approaches”. In: *Computer Animation and Virtual Worlds* 24.1 (2013), pp. 43–64.

[24] Aaron Hertzmann. “A Survey of Stroke-Based Rendering”. In: *IEEE Computer Graphics & Applications* 23.4 (2003).

[25] Aaron Hertzmann. “A survey of stroke-based rendering”. In: *IEEE Computer Graphics and Applications* 23.4 (2003), pp. 70–81.

[26] Aaron Hertzmann. “Algorithms for Rendering in Artistic Styles”. PhD thesis. New York University, 2001.

[27] Aaron Hertzmann. “Can Computers Create Art?” In: *Arts* 7.2 (2018). ISSN: 2076-0752. DOI: 10.3390/arts7020018. URL: https://www.mdpi.com/2076-0752/7/2/18.

[28] Aaron Hertzmann. “Paint By Relaxation”. In: *Proc. CGI*. 2001.

[29] Aaron Hertzmann. “Painterly Rendering with Curved Brush Strokes of Multiple Sizes”. In: *Proc. SIGGRAPH*. 1998.

[30] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[31] Susie Hodge. “Art in detail: 100 masterpieces”. In: *(No Title)* (2016).

[32] Dirk L Hoffmann et al. “Methods for U-series dating of CaCO3 crusts associated with Palaeolithic cave art and application to Iberian sites”. In: *Quaternary Geochronology* 36 (2016), pp. 104–119.

[33] Dirk L Hoffmann et al. “Response to Comment on “U-Th dating of carbonate crusts reveals Neandertal origin of Iberian cave art””. In: *Science* 362.6411 (2018), eaau1736.

[34] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[35] Keith Houston. *The book: a cover-to-cover exploration of the most powerful object of our time*. WW Norton & Company, 2016.

[36] Zhewei Huang, Wen Heng, and Shuchang Zhou. “Learning to Paint With Model-based Deep Reinforcement Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[37] Phillip Isola et al. “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[38] Jonathan Janson. *Essential Veermer 3.0*. 2002. URL: http://www.essentialvermeer.com/technique/perspective/history.html.

[39] Biao Jia et al. “LPaintB: Learning to Paint from Self-Supervision”. In: *Proc. Pacific Graphics*. 2019.

[40] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. “Perceptual losses for real-time style transfer and super-resolution”. In: *European conference on computer vision*. Springer. 2016, pp. 694–711.

[41] Henry Kang, Seungyong Lee, and Charles K Chui. “Flow-based image abstraction”. In: *IEEE transactions on visualization and computer graphics* 15.1 (2008), pp. 62–76.

[42] Hyung W Kang et al. “Interactive sketch generation”. In: *The Visual Computer* 21 (2005), pp. 821–830.

[43] Matthew Kaplan, Bruce Gooch, and Elaine Cohen. “Interactive artistic rendering”. In: *Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*. 2000, pp. 67–74.

[44] Chen Kelun. *Chinese porcelain: Art, elegance, and appreciation*. Long River Press, 2004.

[45] Junhwan Kim and Fabio Pellacini. “Jigsaw image mosaics”. In: *ACM Transactions on Graphics* 21.3 (2002), pp. 657–664.

[46] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[47] Naveen Kodali et al. “On convergence and stability of gans”. In: *arXiv preprint arXiv:1705.07215* (2017).

[48] Nicholas Kolkin, Jason Salavon, and Gregory Shakhnarovich. “Style transfer by relaxed optimal transport and self-similarity”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10051–10060.

[49] MP Pavan Kumar et al. “A comprehensive survey on non-photorealistic rendering and benchmark developments for image abstraction and stylization”. In: *Iran Journal of Computer Science* 2 (2019), pp. 131–165.

[50] Gihyun Kwon and Jong Chul Ye. “Clipstyler: Image style transfer with a single text condition”. In: *arXiv preprint arXiv:2112.00374* (2021).

[51] Jan Eric Kyprianidis et al. “State of the ”Art”: A Taxonomy of Artistic Stylization Techniques for Images and Video”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.5 (2013), pp. 866–885. DOI: 10.1109/TVCG.2012.160.

[52] Tzu-Mao Li et al. “Differentiable vector graphics rasterization for editing and learning”. In: *ACM Trans. Graph.* 39.6 (2020).

[53] Peter Litwinowicz. “Processing Images and Video for an Impressionist Effect”. In: *Proc. SIGGRAPH*. 1997.

[54] Songhua Liu et al. “Paint Transformer: Feed Forward Neural Painting With Stroke Prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 6598–6607.

[55] Fujun Luan et al. “Deep Photo Style Transfer”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[56] George Mann et al. “Print: a manual for librarians and students, describing in detail the history, methods, and applications of printing and paper making”. In: *(No Title)* (1952).

[57] Michael Mathieu, Camille Couprie, and Yann LeCun. *Deep multi-scale video prediction beyond mean square error*. 2016. arXiv: 1511.05440 [cs.LG].

[58] John F. J. Mellor et al. *Unsupervised Doodling and Painting with Improved SPIRAL*. arXiv:1910.01007. 2019.

[59] Mehdi Mirza and Simon Osindero. *Conditional Generative Adversarial Nets*. 2014. arXiv: 1411.1784 [cs.LG].

[60] Steven Mithen. “The prehistory of the mind”. In: *Cambridge Archaeological Journal* 7 (1997), pp. 269–269.

[61] Takeru Miyato et al. “Spectral normalization for generative adversarial networks”. In: *arXiv preprint arXiv:1802.05957* (2018).

[62] Haoran Mo et al. “General Virtual Sketching Framework for Vector Line Art”. In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2021)* 40.4 (2021), 51:1–51:14.

[63] Reiichiro Nakano. *Neural painters: A learned differentiable constraint for generating brushstroke paintings*. arXiv:1904.08410. 2019.

[64] Peter O’Donovan and Aaron Hertzmann. “Anipaint: Interactive painterly animation from video”. In: *IEEE transactions on visualization and computer graphics* 18.3 (2011), pp. 475–487.

[65] Donna J Peuquet. “A conceptual framework and comparison of spatial data models”. In: *Cartographica: The International Journal for Geographic Information and Geovisualization* 21.4 (1984), pp. 66–113.

[66] Aditya Ramesh et al. “Hierarchical text-conditional image generation with clip latents”. In: *arXiv preprint arXiv:2204.06125* 1.2 (2022), p. 3.

[67] Aditya Ramesh et al. “Zero-shot text-to-image generation”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8821–8831.

[68] Reggie Raye and K. Alexandria Bond. “Text-to-CAD: Risks and Opportunities”. In: *The Gradient* (2023).

[69] Paul Rosin and John Collomosse. *Image and Video-Based Artistic Stylisation*. Springer, 2013.

[70] Tim Salimans et al. “Improved techniques for training gans”. In: *Advances in neural information processing systems* 29 (2016).

[71] Michael P Salisbury et al. “Orientable textures for image-based pen-and-ink illustration”. In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 1997, pp. 401–406.

[72] Michaet P Salisbury et al. “Interactive pen-and-ink illustration”. In: *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*. 2023, pp. 393–400.

[73] Anthony Santella and Doug DeCarlo. “Abstracted painterly renderings using eye-tracking data”. In: *Proceedings of the 2nd International Symposium on Non-photorealistic Animation and Rendering*. 2002, 75–ff.

[74] Peter Schaldenbrand and Jean Oh. “Content Masked Loss: Human-Like Brush Stroke Planning in a Reinforcement Learning Painting Agent”. In: *Proc. AAAI*. 2021.

[75] Adrian Secord. “Weighted voronoi stippling”. In: *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*. 2002, pp. 37–43.

[76] Amir Semmo et al. “Image stylization by interactive oil paint filtering”. In: *Computers & Graphics* 55 (2016), pp. 157–171.

[77] Jaskirat Singh and Liang Zheng. “Combining Semantic Guidance and Deep Reinforcement Learning For Generating Human Level Paintings”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 16387–16396.

[78] Ludovic Slimak et al. “Comment on “U-Th dating of carbonate crusts reveals Neandertal origin of Iberian cave art””. In: *Science* 361.6408 (2018), eaau1371.

[79] Jascha Sohl-Dickstein et al. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: *International conference on machine learning*. PMLR. 2015, pp. 2256–2265.

[80] Robert L Solso. *Context, cognition and art*. 1994, pp. 101–127.

[81] Yang Song and Stefano Ermon. “Generative modeling by estimating gradients of the data distribution”. In: *Advances in neural information processing systems* 32 (2019).

[82] Yang Song et al. “Score-based generative modeling through stochastic differential equations”. In: *arXiv preprint arXiv:2011.13456* (2020).

[83] Filippo Stanco et al. “Augmented Perception of the Past-The Case of Hellenistic Syracuse.” In: *Journal of Multimedia* 7.2 (2012).

[84] George Stiny and James Gips. “Shape grammars and the generative specification of painting and sculpture.” In: *IFIP congress (2)*. Vol. 2. 3. Citeseer. 1971, pp. 125–135.

[85] Thomas Strothotte and Stefan Schlechtweg. *Non-photorealistic computer graphics: modeling, rendering, and animation*. Morgan Kaufmann, 2002.

[86] Tamás Szirányi and Zoltán Tóth. “Optimization of paintbrush rendering of images by dynamic MCMC methods”. In: *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer. 2001, pp. 201–215.

[87] Pierre Tallet. “Ayn Sukhna and Wadi el-Jarf: Two newly discovered pharaonic harbours on the Suez Gulf”. In: *British Museum Studies in Ancient Egypt and Sudan* 18 (2012), pp. 147–168.

[88] “The arithmetic of shapes”. In: *Environment and Planning B: Planning and Design* 7.4 (1980), pp. 463–484.

[89] “The construction of shapes”. In: *Environment and Planning B: Planning and Design* 8.1 (1981), pp. 5–40.

[90] “Vector Data Models”. In: *Essentials of Geographic Information Systems* (2012).

[91] Victoria and Albert Museum. *A History of Computer Art*. URL: http://www.vam.ac.uk/content/articles/a/computer-art-history/.

[92] Wikipedia. *History of Art*. URL: https://lobopopart.com.br/en/the-history-of-painting/#:~:text=Painting%5C%20emerged%5C%20in%5C%20prehistory%5C%2C%5C%20when,more%5C%20than%5C%2042%5C%2C000%5C%20years%5C%20ago.

[93] Wikipedia. *Non-photorealistic rendering*. URL: https://en.wikipedia.org/wiki/Non-photorealistic_rendering.

[94] Holger Winnemöller, Sven C Olsen, and Bruce Gooch. “Real-time video abstraction”. In: *ACM Transactions On Graphics (TOG)* 25.3 (2006), pp. 1221–1226.

[95] Ling Yang et al. “Diffusion models: A comprehensive survey of methods and applications”. In: *arXiv preprint arXiv:2209.00796* (2022).

[96] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *Proc. CVPR*. 2018.

[97] Ningyuan Zheng, Yifan Jiang, and Dingjiang Huang. “StrokeNet: A Neural Painting Environment”. In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=HJxwDiActX.

[98] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. 2017.

[99] Zhengxia Zou et al. “Stylized Neural Painting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 15689–15698.

Chapter 3

Im2Painting: Economical Painterly Stylization

MANUEL LADRON DE GUEVARA, MATTHEW FISHER, AARON HERTZMANN. UNDER REVIEW CVPR 2024

This chapter delves into a deep learning-based method for artistic image stylization, a project developed during my first internship at Adobe Research under the mentorship of Aaron Hertzmann and Matthew Fisher.

Our approach introduces an original technique in stroke optimization to achieve artistic stylization from a given input image. Unlike existing computational methods that rely on a large number of brush strokes to represent a scene, our method emulates the efficiency of a skilled artist by using a significantly reduced number of strokes. This, in art, is called *economy*: conveying a lot using a few strokes or lines. This strategy not only aims for high-fidelity image reconstruction but also opens up possibilities for rendering various artistic styles, each characterized by different degrees of abstraction.

The idea of working around the concept of economical art emerges after careful examination of the state of the art in stroke-based methods. In the realm of artistic stylization, a persistent challenge has been the effective execution of style, especially when considering the approaches of existing methodologies. Such methods often in-

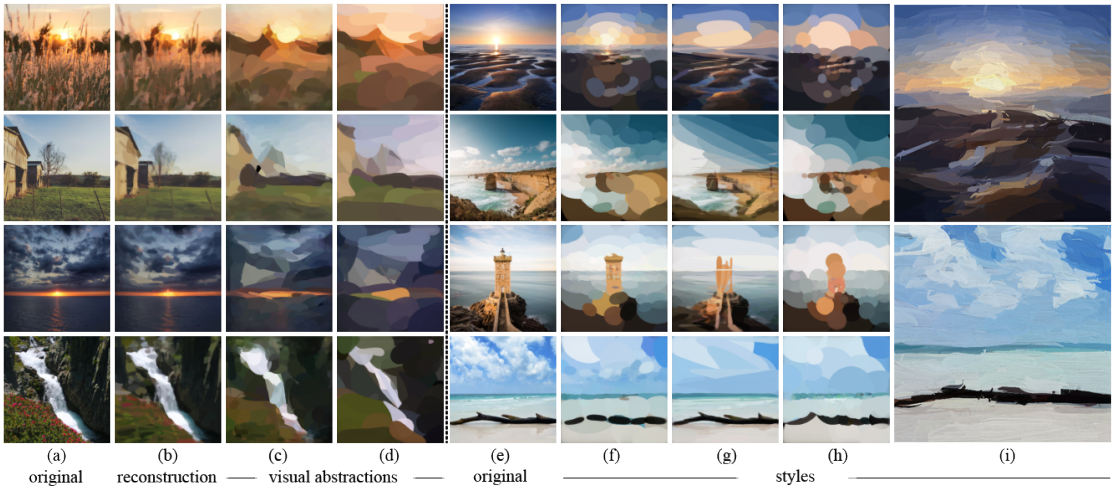


Figure 3.1: Painting Stylization. Our method optimizes brush stroke arrangements from an input image (a), (e) to create various abstraction levels and styles. Left: The best optimization settings results in a realistic style (b). Adjusting constraints and reducing stroke count alters abstraction levels (c-d). Right: Different styles are achieved by adding loss terms and constraints: (f) uses short strokes for structure emphasis, (g) applies more strokes in the foreground for detail, (h) utilizes thick, short strokes for a coarse style, and (i) renders an oil-painting style at 512x512 resolution. Except for (d) with 100 strokes, all paintings use 300 strokes.

volve the application of thousands of tiny strokes on a canvas, gradually building up to a painted representation of an input image. This technique, while effective in some respects, tends to be excessively wasteful, lacking in stroke efficiency. This approach contrasts with human artistic practices, where each stroke is deliberate and meaningful. Recognizing this, we identified an opportunity to enhance stylization through the principle of economy in art. By focusing on the efficient placement of strokes, where each one carries significant artistic weight, we aim to achieve a more effective and nuanced style.

The idea of developing an algorithm rooted in the principle of economic design is a novel approach within the field of computer graphics. This unexplored territory may partly stem from the rarity of individuals who have expertise in both computational science and art. The intersection of these two fields requires a unique blend of skills and perspectives, which is not commonly found. My hypothesis is that this gap in interdis-

ciplinary expertise has contributed to the oversight of integrating economic principles in algorithmic design within computer graphics.

A key aspect of this work is the utilization of much simpler neural network architectures compared to previous learning-based methods. This simplicity circumvents the often complex and resource-intensive process of training control models, common in reinforcement learning (RL). Instead, our approach focuses on learning a direct, non-linear mapping from an image to a set of strokes. This direct mapping technique results in more precise outcomes than previously established methods and allows for a diverse range of style variations. Notably, these variations can be efficiently trained on a single GPU, showcasing the method's efficiency.

In summary, this chapter presents an advancement in computational artistry, demonstrating that artistic stylization with minimal brush strokes and straightforward architectures is not only feasible but also capable of producing high-quality, stylistically diverse results.

3.1 Introduction

A skilled painter can convey the essence of a scene with a few brush strokes (Figure 3.2). In art, this is sometimes called *economy*: conveying a lot with just a minimal number of strokes. The tension between content and media has been hypothesized as a major source of art's appeal [19], and economy heightens this tension: something that just looks both like a compelling landscape and like a splattering of brush strokes can be more intriguing and appealing than an accurate and detailed painting of the same landscape.

Painterly stylization begins with an input photograph, and generates a set of brush strokes to create a stylized version of the input. Many existing painterly stylization algorithms create an appealing "impressionistic" appearance by placing many scattered paint strokes that roughly approximate the image, e.g., [7, 14, 11, 30, 15]. These methods are not economical; they can capture fine details only by drawing thousands of small

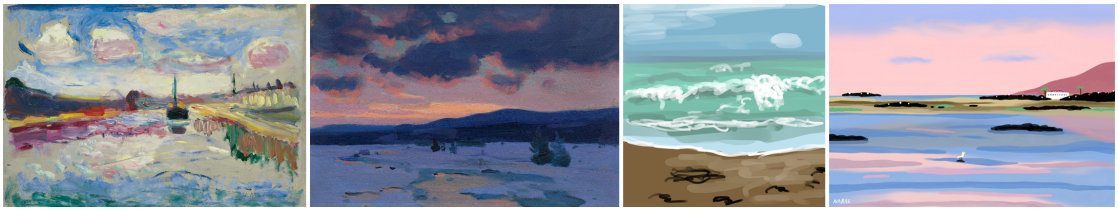


Figure 3.2: Economical painting styles motivate our work. These landscape paintings—oil paintings by Henri Matisse (Canal du Midi, 1899) (leftmost), Clarence Gagnon (Crépuscule D'hiver, 1915) (center-left) and two from unkonwn digital painters—illustrate how an artist can convey a scene with relatively few brush strokes.

strokes.

This chapter proposes a neural algorithm for *economical* painting styles. We express economy as minimizing a loss function with a small number of brush strokes. Our main technical insight is that we can use much simpler architectures than previous methods, by training a network that directly maps an image to a collection of strokes. Perhaps surprisingly, this produces higher-accuracy results as compared with agent-based methods, while being straightforward to train in a few hours. Our method produces accurate renderings with far fewer strokes than previous methods.

This work demonstrates variations of artistic abstraction of the input image, i.e., very visually similar to the input, realistic style, in one extreme (Figure 3.1 (b)), and abstracted representations in the other extreme (Figure 3.1 (c-d)).

We also show variations of artistic *style* by changing the techniques used for the artistic abstraction, such as by preferring shorter but thicker strokes (Figure 3.1 (h)) or injecting noise into the strokes (Figure 3.7 (left, bottom row)). Varying the losses and constraints produces predictable variations in style, e.g., adding more strokes produces more accurate image reconstructions but less visual abstraction.

Though our method could be applied to any class of images, we focus on landscape painting. Landscapes in art history span a range from realism to abstraction, as seen in the work of painters like J. M. W. Turner, Claude Monet, and Richard Diebenkorn, each of whom created increasingly abstract landscapes as they got older. We also show quantitative comparisons on facial portraiture.

Our contributions are summarized as follows:

- We present an approach for economical stroke-based paintings. Our method approximates stylistic versions of the input with fewer but with a more efficient placement of strokes than previous methods, where they often scatter thousand of strokes to approximate an input image (see Figure 3.3).
- Our painting method uses a new and simple architecture based on a direct nonlinear mapping from a given photograph to a set of strokes.
- We demonstrate a controllable framework to achieve different styles, resulting in visually appealing paintings.

3.2 Related Work

Foundations of Stroke-Based Rendering (SBR): The inception of SBR in the digital art world marked a paradigm shift from traditional methods to computational techniques. Early pioneers like Haeberli [7] and Litwinowicz [14] introduced procedural models that opened avenues for creative expression in digital art. These models, although groundbreaking, were limited by their lack of adaptability in stroke manipulation, leading to the exploration of more flexible methods.

Advancements in Optimization-Based SBR: The field witnessed significant evolution with the introduction of optimization-based methods. These approaches, exemplified by Hertzmann's work [10], allowed for dynamic stroke placement, offering artists greater control over the final visual output. Such methods transitioned from basic heuristic optimizations to more sophisticated algorithms, capable of replicating intricate human-like strokes [1]. A more recent approach by Zou et al. [30] optimizes a set of brushstrokes following a multi-scale strategy, achieving different styles principally by using different brushstroke parametrizations and textures, and by applying style transfer techniques [6] after the optimization is done. However, it fails to capture details of the original input photograph with high fidelity.

Figure 3.3: Painting progress of different baselines. Our method places the first 300 strokes more efficiently than previous methods, and captures facial features with fewer strokes. We see this as a measure of economy, which enables stylistic variations.

Incorporation of Deep Learning into SBR: The integration of deep learning revolutionized SBR, introducing neural networks that could simulate complex painting styles. Zheng et al.'s StrokeNet [28] uses RNNs to produce sequences of strokes, although the work can't capture detailed or complex inputs. Huang et al.'s learning-based painting methods [12] are notable examples of using RL to model painting agents to output strokes and update the state of the canvas. These advancements, however, faced challenges in efficiently handling large collections of strokes, particularly in high-resolution artworks. A more recent work by Liu et al. [15] uses a Transformer architecture [24] and models the process as a stroke prediction problem. While this method generates

painterly style paintings, it is structurally rigid and struggles to capture finer details.

Economical Painting in Art: Economical painting, characterized by minimalistic stroke usage, has its roots in traditional art forms, exemplified by artists like Henri Matisse and Kazimir Malevich, who are renowned for their ability to convey rich narratives with minimal brushwork. In the realm of SBR, this principle translates to using fewer, strategically placed strokes to achieve a desired aesthetic effect, balancing simplicity with expressiveness [3, 17]. The minimalist approach in digital painting, inspired by traditional art, offers unique advantages. It emphasizes the importance of each stroke in the overall composition, aligning with contemporary trends in digital art that prioritize efficiency and focus [13, 26]. This approach challenges artists and algorithms alike to consider the weight and impact of every brushstroke in the creation of digital artwork.

3.3 Painting Framework

Given an input image $I \in \mathbb{R}^{3xHxW}$ and a stroke budget T, our method outputs a sequence of brush strokes that, when rendered sequentially onto a blank canvas C_0, produce a painting representation C_T of I, as shown in Figure 3.4. The level of precision of the final canvas as well as the painting style depend on the loss function $\mathcal{L}$, the stroke budget, and the optimization strategy. Except where stated, we use $T = 300$ strokes as the economical stroke budget in this chapter.

Previous work in this field has had varied focuses. Some studies achieve good reconstructions but do not control style, as seen in Huang et al. [12]. Others concentrate on stylization, either through texture, as in Liu et al. [15] and Zou et al. [30], or via brushstroke parameterization and style-transfer techniques [30]. In contrast, our goal is distinct. We aim to train a network capable of producing economical paintings in a range of controllable styles. These styles range from reconstructions to visual abstractions, achieved by varying loss functions and constraints. That is, our main interest resides in the ability to generate artistic styles given by the concept of economy in art. We maintain the same stroke parameterization and we do not use external style images

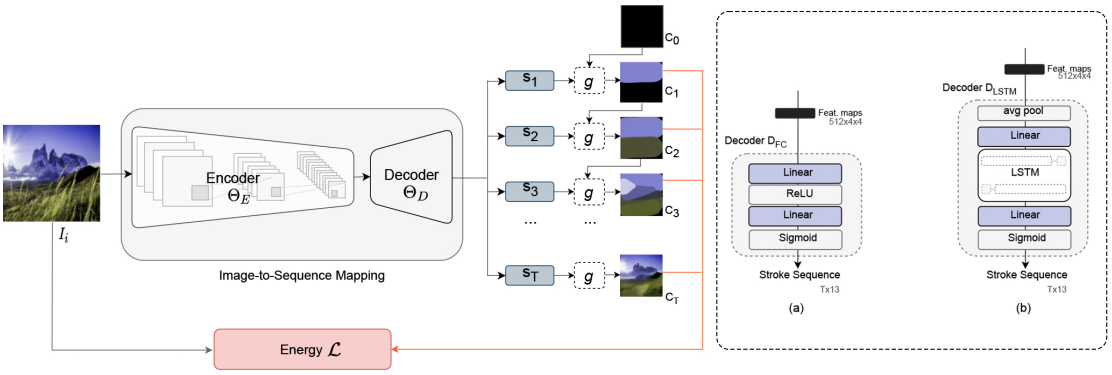


Figure 3.4: Model overview. We train an end-to-end painting network without stroke supervision. (Left) The framework takes an image as input and produces a stylized painting. There are two main components, an encoder function that extracts visual features from an input image, and a projection layer that maps such features into a sequence of brushstroke parameters. Finally, a differentiable renderer sequentially renders the brushstroke parameters onto a canvas. An energy function $\mathcal{L}$ is a combination of loss functions that, in conjunction with the network type and optimization strategy control the style of the output. (Right) Im2Painting Decoders: We propose two decoders, D_{FC} for visual abstractions, consisting on 2 fully connected layers followed by non-linearities (a) and D_{LSTM} which adds an LSTM architecture for precise reconstructions and artistic styles (b).

to achieve style variations.

3.3.1 Model Architecture

We propose a direct non-linear mapping network $f : \mathcal{I} \rightarrow \mathcal{S}^{T\times 13}$ from image to brush stroke sequence. We split the mapping into an encoder and a decoder (Figure 3.4). The encoder extracts feature maps using the four residual blocks of a Resnet-18 [8], not including the last average pooling layer, to get a vector of feature maps $X \in \mathcal{X}^{512\times 4\times 4}$ from a given image I. The decoder D transforms these feature maps into a fixed sequence of stroke parameters $s = \{s_1, s_2, ..., s_T\}$, that is, a $T \times 13$ vector. A differentiable renderer g renders the stroke parameters onto a canvas, and an energy function $\mathcal{L}$ evaluates how well the painting fits the desired artistic style.

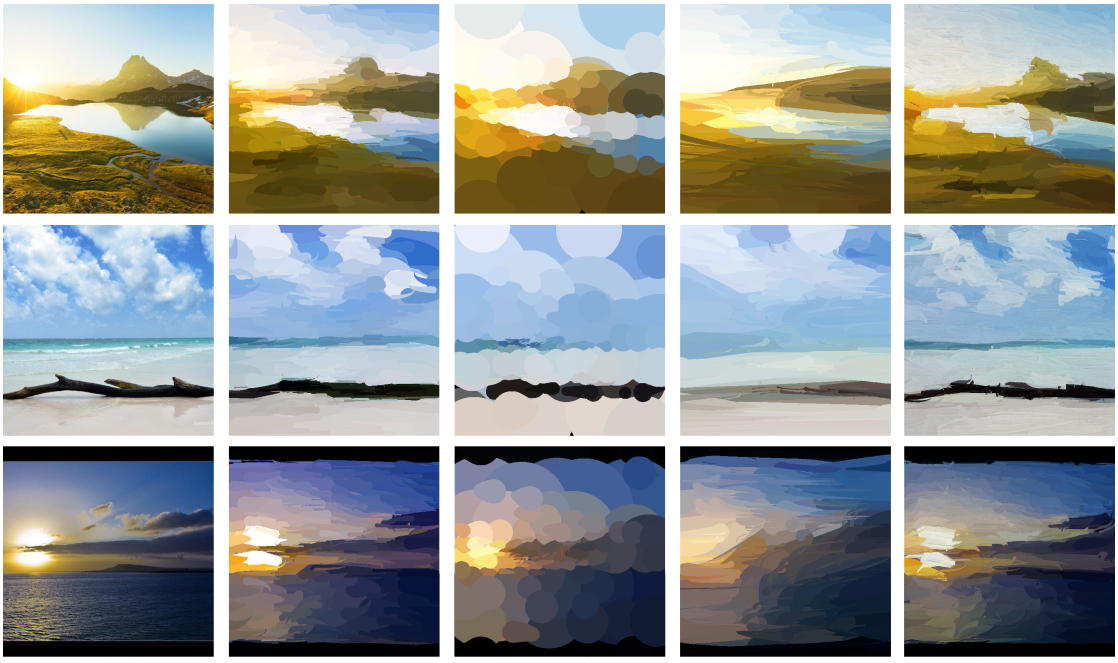
In this work, we use two different decoder architectures, D_{FC} and D_{LSTM}, as shown in Figure 3.4. The first decoder architecture, D_{FC}, is a stack of non-linear fully-connected

(FC) layers. D_{FC} resizes and transforms X into a fixed sequence of strokes using 2 FC layers, with ReLU after the first layer and a sigmoid function after the last layer. The second decoder, D_{LSTM}, uses a bidirectional LSTM[1] layer in combination with fully-connected layers. D_{LSTM} uses average pooling on X to get a vector $H \in \mathbb{R}^{512}$ before feeding it into its first FC layer. The first FC layer expands H into $W \in \mathbb{R}^{512 \times T}$, forming the sequence of vectors that are the input to the LSTM layer. A second FC followed by a sigmoid outputs the sequence of brushstroke parameters. We find that D_{FC} works better for visual abstractions, whereas D_{LSTM} is more effective at producing precise reconstructions and artistic styles (see Appendix A for a thorough comparison, more technical details, and visualizations about the difference between the two decoders.)

3.3.2 Stroke Parameterization and Differentiable Renderer

We use the stroke parameter representation and differentiable renderer provided by [12]. Each stroke is parameterized by a 13-D tuple that encodes start, middle and end points of a quadratic Bézier curve, radii and transparency at start and end points, and RGB color of the stroke. We use a neural renderer g that has been previously trained to approximate a non-differentiable renderer. g consists of four fully connected layers and six convolutional layers, and takes in the sequence of actions, one by one, and sequentially updates the initial canvas C_0. The rendered painting C_T is then passed to a loss (or energy) function $\mathcal{L}$, which generally is a combination of different losses that

[1] An LSTM, or Long Short-Term Memory, is a type of artificial intelligence algorithm used in the field of ML, particularly in the processing of sequential data like language or time series. Imagine it as a special kind of 'brain cell' (neuron) in a computer's neural network that has a remarkable ability to remember and understand sequences of information for an extended period. Unlike standard neurons that process data in a single step, LSTMs have internal mechanisms, often likened to gates, that can maintain, update, or forget information based on its relevance. This makes them exceptionally useful in tasks like language translation, speech recognition, or predicting the next word in a sentence, where understanding the context and order of past information is crucial.



(a) Original (b) Abstraction - smooth (c) Style - thickness and length (d) Style - rapid motions (e) Style - oil-painting

Figure 3.5: 512x512 300-stroke economical paintings. These examples illustrate results with higher resolution, and show how our method can provide nuanced, interpretable control over styles through varying losses and other interpretable factors. (b) Shows a basic style by smooth visual abstraction, (c) adds constraints on thickness and length that reduces the number of small strokes, (d) adds random noise to strokes resulting in a scratchy style, (e) abstraction in oil-painting style.

contribute to generating the style of the painting.

A common fact when working with these stroke models is that they do not optimize nor train with textures. Such textures e.g., an oil painting texture, are added at inference time.

While previous methods use an $N \times N$ canvas subdivision [12, 15, 30] to paint small strokes on N^2 patches in parallel, our network operates at the canvas level at all times.

3.3.3 Loss Functions and Optimization

Training our model with different loss functions, weights, and optimization schemes produces different styles. Given an input image I, we define the objective as a combi-

nation of different loss functions:

$$\mathcal{L}_{\text{painting}} = \lambda_1 \mathcal{L}_{\text{perc}} + \lambda_2 \mathcal{L}_{\text{guidance}} + \lambda_3 \mathcal{L}_{\text{style}} \tag{3.1}$$

where the first term is a perceptual loss comparing the output painting C_T to the input image, the second term is a pixel loss applied to intermediate canvases, the third term is optional style losses on strokes, and λ_n are values that weigh the contribution of each loss function. We explain the first two terms below, and describe the optional style losses in the next section.

Perceptual loss. We use a standard form of perceptual loss [25]. Such losses produce more appealing results than pixelwise losses like L_2, which lead to blurry paintings. Specifically, let $V_{ij} = \{V_{ij}^1, ..., V_{ij}^k\}$ and $W_{ij} = \{W_{ij}^1, ..., W_{ij}^k\}$ be a set of k feature vectors extracted from image I and canvas C_T, respectively, we use cosine similarity as follows:

$$\mathcal{L}_{\text{perc}} = -\cos\theta = \frac{1}{K} \sum_{k}^{K} \sum_{ij} V_{ij}^k W_{ij}^k \,/\, \left\|V_{ij}^k\right\| \left\|W_{ij}^k\right\| \tag{3.2}$$

where i, j index the spatial dimensions of the feature maps V and W, and K the extracted layers from VGG16 trained on ImageNet [21]. Specifically, we use layers 1, 3, 6, 8, 11, 13, 15, 22 and 29[2] . Perceptual loss generally captures high-frequency parts of

[2]The VGG16 network, trained on the ImageNet dataset, is widely used for perceptual loss due to its effectiveness in capturing a wide range of image features. The selected layers (1, 3, 6, 8, 11, 13, 15, 22, and 29) represent different levels of abstraction in image processing. Early layers tend to capture basic features like edges and textures, while deeper layers represent more complex patterns and objects. By using these specific layers from a pre-trained VGG16 network, we can measure perceptual similarities in a way that aligns more closely with human visual perception. This approach helps in emphasizing the content and style in the images, leading to results that are more visually appealing and realistic from a human perspective, as compared to traditional pixel-wise loss functions.

the images, which are not usually represented by pixel losses.

Guidance losses. Previous methods typically apply losses to the model's output. However, propagating the loss error gradient to earlier strokes leads to slow convergence. RL methods partially bypass this with critic functions[3] , but training the critic is itself challenging, leading to very long training times.

Instead, we simply apply a pixelwise loss to every intermediate stage of the canvas to help "guide" the optimization. Let C_t be the rendered painting after the t-th stroke is added, so that C_0 is the initial canvas. Then,

$$\mathcal{L}_{\text{guidance}} = \sum_{t=1}^{T} \mathcal{L}_{\text{pixel}}(I, C_t) \tag{3.3}$$

where $\mathcal{L}_{\text{pixel}}$ computes the pixelwise L_1 distance between the input image and C_t.

3.4 Economical Painterly Styles

In this section, we show a range of artistic styles, from accurate reconstructions or realistic-looking paintings, to variations of visual abstraction given by loosening constraints of our precise network and varying optimization guidance, stroke budget, shape constraints or rapid drawing motions. We show that a limited stroke budget of 300 strokes is enough to achieve all our styles, and we can increase visual abstraction more by further limiting the stroke budget.

[3] In the context of reinforcement learning, critic functions are part of an architecture known as Actor-Critic models. The critic evaluates the actions taken by the actor by estimating the value function, which represents the expected cumulative reward. This feedback helps in training the actor to make better decisions. Training the critic itself is challenging, as it requires learning to accurately predict the future rewards based on the current state and actions.

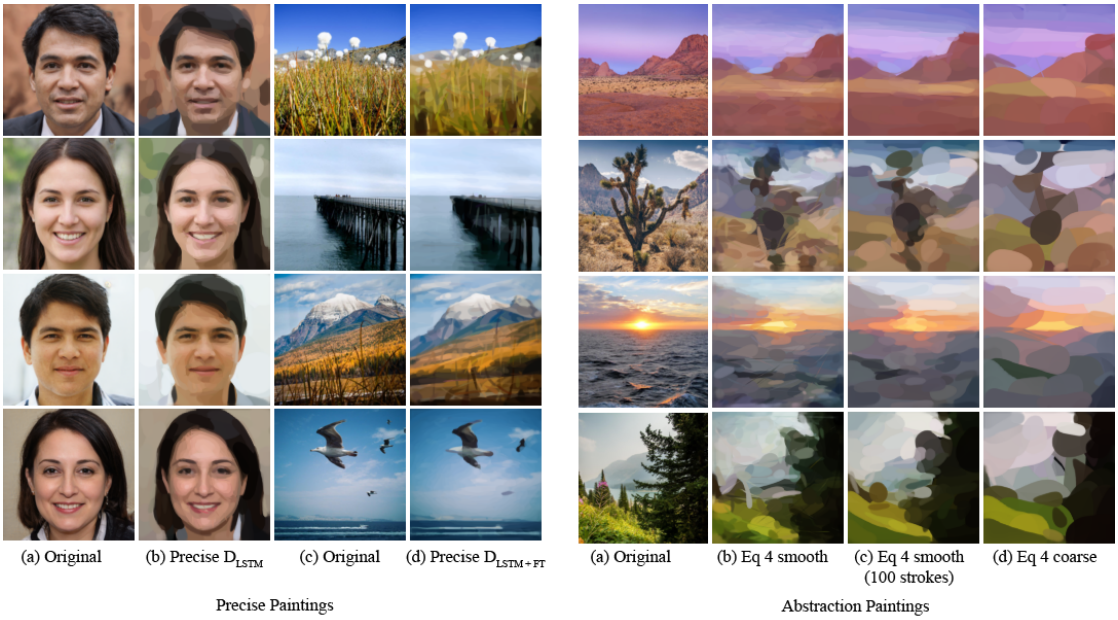


Figure 3.6: Accurate Reconstructions and Visual Abstractions. (Left) Our network is able to paint accurate reconstructions with a limited budget of only 300 strokes. This is achieved using decoder D_{LSTM}. (Right) We can increase visual abstraction by decreasing stroke budgets and using coarser loss functions. (a) Input image. (b) 300 strokes using $\mathcal{L}_1$ on all intermediate canvases. (c) 100 strokes using $\mathcal{L}_1$ on all intermediate canvases. (d) 300 strokes evaluating $\mathcal{L}_1$ only on the final frame. We use D_{FC} for all abstractions. Readers are encouraged to zoom in closely for better inspection.

Training details. Different optimization approaches require different settings. For precise depiction painting, we train on 8 Nvidia Tesla V100 16-GB GPU with batch size of 168, and takes approximately 24h to train. For other optimization methods, we train on a single GPU with a batch size of 24, converging after 5h on average. We use Adam optimizer with a learning rate of 0.0002 and betas 0.5 and 0.99. We use 100000 landscapes training images gathered from [5, 22, 29] and 200000 CelebA [16] training images for precise depiction.

Accurate Reconstruction We first describe a basic style that shows the ability of our model to achieve precise reconstructions using an economical budget. Even though we find this realistic style less artistically interesting, we use it as a testbed to assess

the validation of our method, because reconstruction is a measure of economy. To achieve this precise or realistic style, we set the loss function to $\mathcal{L}_{\text{realistic}} = \lambda_1 \mathcal{L}_{\text{perc}} + \lambda_2 \mathcal{L}_{\text{guidance}}$, where $\lambda_2 = 1$ and we set λ_1 to 0.1 on landscapes and to 0.001 on face portraits. We do not use any style-specific loss term, $\mathcal{L}_{style}$, and we use decoder D_{LSTM}. We show paintings on this style in Figure 3.1 (b), Figure 3.6 (left) (b, d), Figure 3.3 (top), Figure 3.7 (right, b), and ablations in the supplemental. Landscapes require a higher λ due to a higher variance of the distribution of images. To achieve nearly-perfect reconstructions on landscapes, we add a fine-tuning step consisting on a 100-step optimization of the stroke parameters output by the network. Note that such step is only necessary for high-accurate reconstructions on landscapes, but not necessary for face images, nor any other visual style (Figure 3.6 (left) (b) does not use fine-tune step, (d) uses fine-tune step. Refer to Appendix A for more details).

Visual Abstraction. We can achieve stylistic variations and artistic abstraction by reducing optimality, while preserving the semantics of the input image. This is done by varying the optimization process, i.e, backpropagating gradients at different timesteps. We find that using only L_1 loss reduces visual fidelity, as does applying the loss only on the final canvas C_T rather than on all canvases as in Equation 3.3. The only loss we use for these experiments is $\mathcal{L}_{\text{abst}} = \lambda_2 \mathcal{L}_{\text{guidance}}$, and our guidance loss is defined as follows:

$$\mathcal{L}_{\text{guidance}} = \begin{cases} \sum_{t=1}^{T} ||I - C_t||_1 & \text{smooth} \\ \sum_{t=1}^{T/k} ||I - C_k t||_1 & \text{medium} \\ ||I - C_T||_1 & \text{coarse} \end{cases} \tag{3.4}$$

Where smooth, medium, and coarse define the degree of abstraction within an economical style. Likewise, we can further control the level of abstraction by varying the number of strokes used; fewer strokes produce coarser abstractions. Examples of these ranges of abstraction are shown in Figure 3.6 (right). This figure, arranged left to right in increasing abstraction, shows how economical paintings are achieved using different constraints. We can see the effect of the number of strokes and the optimization schema

in the last two columns. In (c), paintings are generated by using just 100 strokes and optimizing using a smooth guidance. However, the last column (d) uses 300 strokes within a coarse schema, which results in a more abstracted painting than the previous paintings, even using three times more strokes. The coarse loss is computed only in the final frame, which causes strokes in early stages of the process to not contribute much to the final painting (refer to Appendix A for more details). For these experiments we use D_{FC}. We also show the results of applying an oil painting texture on smooth abstraction using the thin plate spline algorithm [27, 2] in Figure 3.1 (i), and Figure 3.5 (b, e). For more examples, refer to Appendix A.

Stroke Styles. In the above styles, the network is free to paint with any stroke size. In this section, we add optional style losses to penalize stroke parameters that lie outside desired ranges.

Specifically, let S_p be some property of stroke shape. We set a shape threshold T_s to penalize properties that are too large.

$$\mathcal{L}_{\text{shape}} = \begin{cases} 0 & \text{if } S_p < T_s \\ ||S_p - T_s||_1 & \text{if } S_p \geq T_s \end{cases} \tag{3.5}$$

The properties we restrict are (1) the arc length of the control polygon $S_l = ||\mathbf{p}_1 - \mathbf{p}_2|| + ||\mathbf{p}_2 - \mathbf{p}_3||$, where $\mathbf{p}_i = [x_i, y_i]^T$ are the three control point locations, and (2) the center stroke radius $S_w = (r_1 + r_2)/2$, where r_1 and r_2 are the start and end radii. Theses losses are summed over all canvases, i.e., $\mathcal{L}_{\text{style}} = \sum_t \mathcal{L}_{\text{shape}}(I, C_t)$ in this case.

Figure 3.7 (left) shows artistic styles using these losses. (b) Shows a style using an arc length threshold $T_s = 0.3$. Strokes longer than a third of the canvas are penalized, and so the network compensates with thicker strokes. This results in a smooth image approximation. As shown in (c), penalizing thick strokes ($T_s = 0.05$) leads to a style with few large strokes in low-detail regions, instead placing much more detail in the foreground. Due to the difficulty in approximating the input image with 300 thin strokes,

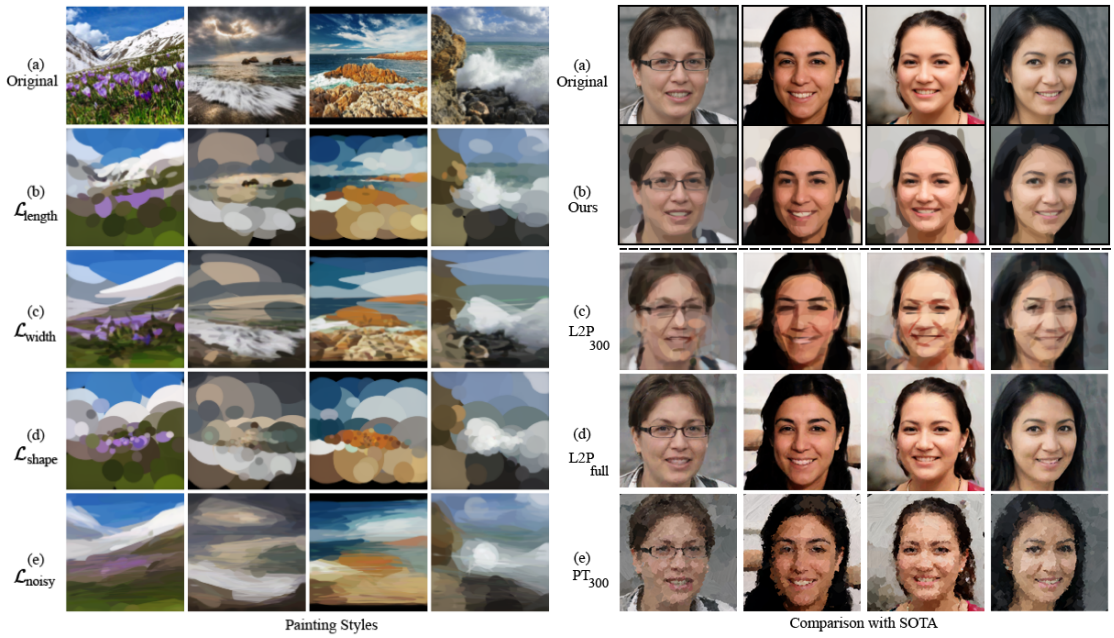


Figure 3.7: Painting Styles (left) and SOTA Comparison (right). Painting Styles (left): (a) input image, (b) style by constraint on stroke length. A small penalty encourages the network to approximate the image with short strokes. The network generally chooses a larger stroke thickness to balance the lack of long strokes. (c) Style by stroke width penalty. The network learns to quickly approximate the upper part of the image with very few strokes that receive a penalty, creating a detailed foreground and coarse background effect. (d) A combination of constraints on width and length leads to a style of thick and short rounded shapes in a coarse but smooth style. (e) Scratchy effect by approximating rapid motions with Gaussian noise. We use D_{LSTM} for all styles. SOTA Comparison (Right): (a) Original, (b) our results using D_{LSTM}. (c) and (d) Learning-to-Paint [12] limited to 300 strokes and full model, respectively, (e) Paint Transformers [15] limited to 300 strokes.

the network assumes a larger cost in a small number of strokes on low-frequency areas of the input image by choosing thicker strokes, resulting on an interesting contrasting style of thick and thin strokes. (d) Combines both these constraints. For these experiments, we use D_{LSTM} and we set λ_1 to 0.01, λ_2 to 1 and λ_3 to 0.001.

Noisy Motions. With enough drawing skills and time, a human can accurately paint a precise painting of a target image. However, on a limited time budget, painting becomes less precise and more like a rough sketch. These rapid motions create their own styles

Method	Model-300S			Full Model		
	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	N. Strokes	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	N. Strokes
L2P (RL) [12]	0.037	0.729	300	**0.022**	**0.890**	1000
Paint Transformers [15]	0.089	0.490	351	0.057	0.609	3212
SNP (Optimization) [30]	0.043	0.631	300	0.030	0.625	1000
Ours$_{D_{LSTM}}$	**0.035**	**0.747**	300	0.035	0.747	300
Ours$_{D_{FC}}$	0.046	0.735	300	0.046	0.735	300

Table 3.1: Quantitative results on 500 random held-out faces from the CelebA dataset. We limit each method to 300 strokes the painting algorithms from previous work. We measure L_1 loss and perceptual loss using cosine similarity between input image and painting feature maps. Our method achieves slightly better metrics in the first 300 strokes, an indication of economy. Note that our 300-stroke model is our full model.

in writing [4] and painting.

We approximate the effect of motion dynamics given by a limited time per individual stroke by adding Gaussian noise into the actions. Let $\mathbf{p}_{1:3}$ be a vector of brushstroke coordinates produced by the painter network at run-time. We add noise to the control points as $\hat{\mathbf{p}}_{1:3} = \mathbf{p}_{1:3} + z$ where $z \sim \mathcal{N}(0, \beta^2\mathbf{I})$. The minimization objective becomes

$$\mathcal{L}_{\text{noisy}} = \lambda \mathcal{L}_{\text{perc}}(I, C_T) + \sum_{t=1}^{T} \mathbb{E}_{z_t \sim p_z} \left[\mathcal{L}_{\text{guidance}}(I, C_t) \right] \tag{3.6}$$

where C_t is affected by the noise value as described above. In the context of stochastic optimization, we can approximate the expectation with a single random sample, and, further improve optimization by fixing the random seed [23, 18]. We use D_{LSTM} and set the same lambdas as for the precise network, and we use $\beta = 0.05$ and $\lambda = 0.01$. We do not apply noise at test time.

Figure 3.7 (left, e), Figure 3.5 (d) show examples of this style. Because the network learns that movements are noisy, the painter overrides any detail and approximates the main image composition with scratchy strokes.

3.4.1 Comparison with State-of-the-Art Methods

We compare our method with three previous works: an optimization based method, Stylized Neural Painting (SNP) by [30], an RL method, Learning to Paint (L2P) by [12], and a stroke prediction method based on transformers named Paint Transformer (PT) by [15]. Simply comparing reconstruction error alone would provide a poor metric for painting algorithms. In this case, the best scores would be achieved by an algorithm that produces thousands of tiny strokes, reproducing the input image almost exactly, but not looking much like a painting. Instead we focus on reconstruction error, not as a measure of image quality but a measure of economy—if the method is able to reconstruct the input using a few strokes, it is successful at achieving an economical painting, as all strokes have been carefully placed. How well can an algorithm reconstruct an image with only 300 brush strokes?

Quantitative Comparison We first provide a comparison on portraiture with existing methods on our precision network, trained on CelebA and reported in Table 3.1. In the first two columns all models use around 300 strokes. SNP is an optimization method and to ensure a fair comparison, we limit the number of strokes in the optimization procedure. Paint Transformer casts the painting process as a stroke prediction task, and generates a set of 8 random background strokes in a 32x32 canvas, and a second set of 8 foreground strokes. The network is tasked to correctly guess the foreground stroke parameters from the background set of strokes. Since it trains on a random number of strokes, the training process is not related to the final number of strokes used at inference. To ensure a fair comparison, we adjust the algorithm so that the total number of strokes is approximately 300, and we remove all textures. We also retrain L2P with 300 strokes to ensure fair comparison. Our high precision network outperforms previous methods when using 300 strokes, as seen in Figure 3.7 (right). Our D_{LSTM} achieves better reconstructions than D_{FC} (see Appendix A for more details).

Table 3.2 shows quantitative results on landscapes using 300 strokes (left), and 1000 or more strokes (right). Metrics are calculated using 100 samples from a held-out set.

Method	Model-300S			Full Model		
	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	N. Strokes	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	N. Strokes
l2p (RL) [12]	**0.041**	0.651	300	**0.040**	**0.814**	1000
Paint Transformers [15]	0.092	0.513	351	0.068	0.678	3212
SNP (Optimization) [30]	0.055	0.561	300	0.046	0.561	1000
Ours$_{D_{LSTM}}$	0.048	**0.658**	300	0.045	0.667	1000
Ours$_{D_{LSTM}+FT}$*	0.023	0.868	300	0.020	0.899	1000

Table 3.2: Quantitative results on 100 random sample images drawn from a held-out set on landscapes. We provide metrics on a 300-stroke model (left), and a 1000-stroke model (right). For [15], total number of strokes is averaged over the held-out set. We measure L_1 and perceptual loss using cosine similarity between input image and painting feature maps. For a reduced number of strokes, our method achieves a lower perceptual loss than previous methods. *We include the optional finetuning step for information purposes; a more meaningful comparison is done without such finetuning step.

We use D_{LSTM} decoder. Our method achieves better scores in perceptual loss than previous methods, but does not obtain better metrics than L2P on pixel loss. We add our method with fine-tuning step for information purposes.

Qualitative Comparison. The lack of previous methods that produce visual abstractions and styles without the use of external style images or brush textures, challenges an assessment on styles. We instead compare our precision network with existing methods as a measure of economy. Figure 3.3 shows the painting progress at 50, 100, 150, 200 and 300 strokes. We see that our method needs fewer strokes to paint facial details than previous methods. L2P [12] achieves similar reconstructions on faces with hundreds of more strokes. Figure 3.8 shows landscapes paintings. Qualitatively, our method generates less artifacts than L2P (even though L2P performs slightly better on L_1 and slightly worse on perceptual loss) and shows better reconstructions than Paint Transformer and SNP. More details are found in Appendix A.

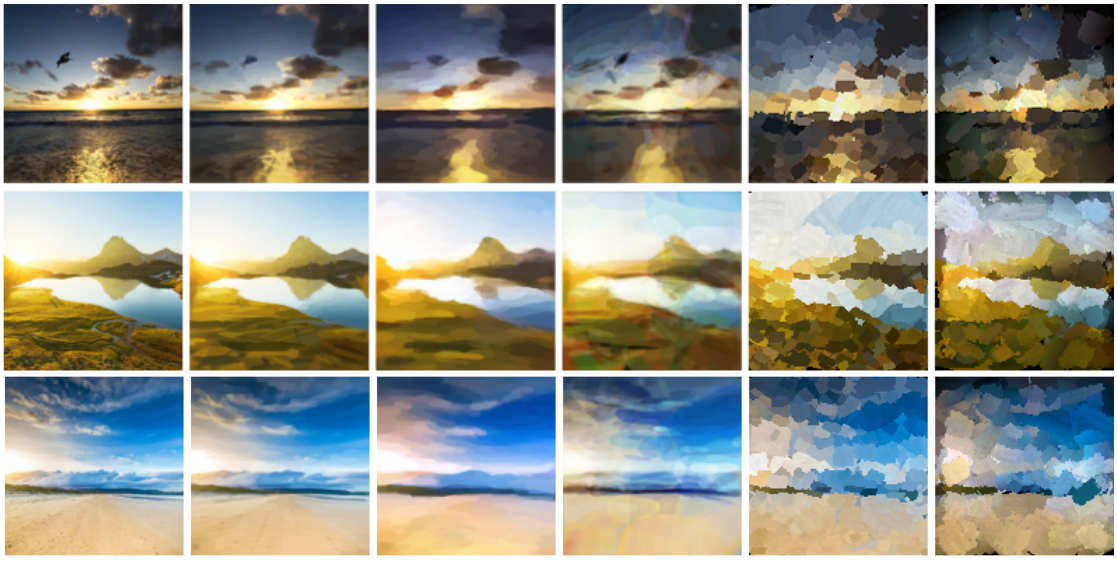


Figure 3.8: Precise reconstructions comparison with previous methods. All methods use 300 strokes. Our method with a fine-tuning step (a) is able to reconstruct the input image better. Without fine-tuning step (b) our method still outputs smoother paintings than previous work. Paint Transformers [15] and SNP [30] output very large strokes and they suffer with reduced stroke budgets.

3.5 Conclusion and Discussion

This chapter presents an approach for economical-style paintings. We argue that *economy* is a foundational approach for many painting styles and a step towards developing general-purpose painting techniques. This is because, regardless of the desired style, placing strokes efficiently leads to a more controlled and intentional painting. We see that our method, in general, places strokes more efficiently than previous methods (Figure 3.3), which, in contrast, spray thousands of strokes to reproduce the input image, which leads to a less desirable approach to work on styles.

The main goal of the work presented in this chapter is to achieve high-quality stylization in distinct styles using very few strokes, a common feature of many artistic visual styles. We show that a much simpler architecture may be used than in previous methods, which will enable faster research progress in this space. Like many works in this

literature [9, 20], our goal is not to advance an artificial benchmark but to produce novel, useful artistic visual styles. We provided a simpler model and showed a more efficient optimization than previous methods. For instance, a robotic arm could paint much faster using our method, as it uses way fewer strokes. Our trained network lays the groundwork for a more straightforward optimization process, enabling further refinement of stroke parameters to explore additional styles more efficiently.

This chapter demonstrates that it is possible to control the existing trade-off between accurate reconstruction and abstraction. In contrast to previous methods that are able to produce either accurate representations or abstracted styles, we provide a flexible optimization framework that allows for adjustments in artistic styles without using external style images. From highly precise paintings to very abstract styles, we demonstrate that combining pixel, perceptual and style-specific losses in an energy function, and optimizing under a greedy or a one-off scheme is enough to capture a portion of the spectrum of visual abstraction. Moreover, our method proves a better optimization than previous methods, achieving higher precision paintings in approximately a third of the strokes.

Limitations. Our algorithm still suffers from local minima problems, and the optimization for the high-precision network converges slowly, resulting in larger training time. Our algorithm maintains a tight relation between output size and number of strokes. We choose to work with a tight stroke budget of 300 strokes to produce economical paintings, which we demonstrate enough for good reconstructions of size 128x128. Since the algorithm outputs stroke parameters, at test time we can substitute the pre-trained neural renderer by a non-differentiable renderer and make the painting larger, as seen in Figure 3.5. Even though our best results on precision on CelebA do not need further processing, in order to achieve accurate reconstructions on landscape images we need to do a fine-tuning operation. This is partially because the variance in the distribution of landscapes images is much higher than the CelebA dataset. This step is not needed for any other style.

Another limitation of the work presented in this chapter is that there is a lack of

control over the painting process. That is, we cannot control the amount of strokes that are dedicated to each semantic area of the original picture. Moreover, artistic styles are still limited and results still have a computational feeling.

Discussion Our work scratches the surface for economical painterly stylization. Previous work focuses either on good reconstructions of an input, or on generating some level of abstraction via texture maps or relying on external style images. For instance, Learning to Paint [12] is able to approximate the input with high accuracy with more than a thousand strokes, but does not output any stylistic variation. Paint Transformers [15] and Stylized Neural Painters (SNP) [30] output paintings in a very similar style. They use texture maps in their strokes and SNP achieves style by changing the stroke primitive or doing style transfer. Our work, however, focuses on the generation of variations and styles (given by the concept of economy in art) using different optimization procedures and loss functions, but keeping the same stroke primitive (quadratic Bézier curve). The concept of economy in art opens up new stylistic possibilities, and we design our networks under this scenario. Even though our method generates good reconstructions, we find it less interesting than the explored artistic styles and visual abstractions. Future research can build upon this concept, leveraging the technical feasibility of moving from simple input reconstructions to more abstract and conceptual representations with a reduced number of strokes. For example, setting a lower stroke limit opens the door to depicting landscapes in a more abstract or conceptual manner, as a limited stroke count naturally results in less detailed, more stylized representations.

In the next two chapters, we work on some of the limitations presented in this chapter. We will see how constructing mechanisms to provide control over the painting process, and adopting some human priors such as working memory and painting techniques not only improves the quality of the results considerably, but also yields a greater stylistic variation.

References

[1] Aseem Agarwala et al. “Interactive digital photomontage”. In: *ACM SIGGRAPH 2004 Papers*. 2004, pp. 294–302.

[2] Ian Barrodale et al. “Warping digital images using thin plate splines”. In: *Pattern Recognit.* 26 (1993), pp. 375–376.

[3] Itamar Berger et al. “Style and abstraction in portrait sketching”. In: *ACM Transactions on Graphics (TOG)*. Vol. 32. 4. 2013, pp. 1–12.

[4] D. Berio, S. Calinon, and F. Fol Leymarie. “Learning dynamic graffiti strokes with a compliant robot”. In: *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*. Daejeon, Korea, Oct. 2016.

[5] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. “Cartoongan: Generative adversarial networks for photo cartoonization”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 9465–9474.

[6] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. “Image Style Transfer Using Convolutional Neural Networks”. In: *CVPR*. 2016.

[7] Paul Haeberli. “Paint By Numbers: Abstract Image Representations”. In: *Proc. SIGGRAPH*. 1990.

[8] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[9] Aaron Hertzmann. “A Survey of Stroke-Based Rendering”. In: *IEEE Computer Graphics & Applications* 23.4 (2003).

[10] Aaron Hertzmann. “A survey of stroke-based rendering”. In: *IEEE Computer Graphics and Applications* 23.4 (2003), pp. 70–81.

[11] Aaron Hertzmann. “Painterly Rendering with Curved Brush Strokes of Multiple Sizes”. In: *Proc. SIGGRAPH*. 1998.

[12] Zhewei Huang, Wen Heng, and Shuchang Zhou. “Learning to Paint With Model-based Deep Reinforcement Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[13] Jan Eric Kyprianidis et al. “State of the” art”: A taxonomy of artistic stylization techniques for images and video”. In: *IEEE transactions on visualization and computer graphics* 19.5 (2012), pp. 866–885.

[14] Peter Litwinowicz. “Processing Images and Video for an Impressionist Effect”. In: *Proc. SIGGRAPH*. 1997.

[15] Songhua Liu et al. “Paint Transformer: Feed Forward Neural Painting With Stroke Prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 6598–6607.

[16] Ziwei Liu et al. “Deep Learning Face Attributes in the Wild”. In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dec. 2015.

[17] Jingwan Lu et al. “Decobrush: Drawing structured decorative patterns by example”. In: *ACM Transactions on Graphics (TOG)* 33.4 (2014), pp. 1–9.

[18] Andrew Y Ng and Michael I Jordan. “PEGASUS: A policy search method for large MDPs and POMDPs”. In: *UAI*. 2000.

[19] Robert Pepperell. “Artworks as dichotomous objects: implications for the scientific study of aesthetic experience”. In: *Frontiers in Human Neuroscience* 9 (2015). ISSN: 1662-5161. DOI: 10.3389/fnhum.2015.00295. URL: https://www.frontiersin.org/article/10.3389/fnhum.2015.00295.

[20] Paul Rosin and John Collomosse. *Image and Video-Based Artistic Stylisation*. Springer, 2013.

[21] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[22] Ivan Skorokhodov, Grigorii Sotnikov, and Mohamed Elhoseiny. “Aligning latent and image spaces to connect the unconnectable”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 14144–14153.

[23] James C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley, 2003.

[24] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[25] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *Proc. CVPR*. 2018.

[26] Changshen Zhao. “A survey on image style transfer approaches using deep learning”. In: *Journal of Physics: Conference Series*. Vol. 1453. 1. IOP Publishing. 2020, p. 012129.

[27] Mingtian Zhao and Song-Chun Zhu. “Portrait painting using active templates”. In: *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering*. 2011, pp. 117–124.

[28] Ningyuan Zheng, Yifan Jiang, and Dingjiang Huang. “StrokeNet: A Neural Painting Environment”. In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=HJxwDiActX.

[29] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. 2017.

[30] Zhengxia Zou et al. “Stylized Neural Painting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 15689–15698.

Chapter 4

Stroke Optimization Methodology

(a) (b) (c) (d)

Figure 4.1: Coarse to Fine Detail Approach. Progression of detail levels using different stroke and width settings: (a) 25 strokes, no clipping; (b) 36 strokes, 10% width clipping; (c) 81 strokes, 5% width clipping; (d) 324 strokes, 5% width clipping. The original image is shown as an inset.

The preceding chapter explored stylization under the concept of economy in art. We saw how our neural methodology places strokes more efficiently, necessitating fewer strokes than previous methods to recreate a painting version of an input image. This, in fact, lays the ground for further stylization, as we saw some stylistic variations by simply varying the parameters of our training method. Nevertheless, our neural method presented some limitations in stylization and control.

In the current chapter, we shift our focus specifically towards optimization-based methodologies, starting with a technical explication of these strategies. Initially, a foundational algorithm will be dissected to furnish an elemental understanding of the core principles. Subsequently, we will transition this preliminary exposition to introduce an

innovative, high-fidelity method based on semantic and attention-guided paradigms, inspired by human painting strategies and visual working memory, presented in the next chapter.

While certain sub-components of the algorithms delineated in this chapter employ deep learning techniques—most notably in the rendering engine—the central stroke formation and positioning are computed through non-neural algorithmic procedures. For the sake of terminological conciseness, we refer to these stroke optimization-based painting methodologies as SOBP (Stroke Optimization-Based Painting).

4.1 Introduction

Direct optimization on stroke parameters presents an alternative to hand-design algorithms. The primary objective of stroke optimization techniques lies in the identification of an optimal set of stroke parameters that minimize a specified objective function. It will become evident that locating this optimal configuration poses substantial challenges, particularly when attempting to simultaneously optimize a comprehensive set of strokes. To ameliorate the complexities inherent in this optimization landscape, I have employed an array of strategies designed to facilitate convergence towards locally or globally almost "optimal" solutions.

Optimization in the context of stroke parameters is done with reference to a singular target image, necessitating an individual optimization procedure for each new image we want to stylize. This stands in contrast to neural network-based methodologies, which have the capacity to generalize across multiple images via batch processing. It is important to note that optimization-based techniques are inherently non-generalizable, given that they do not engage in data-driven learning, that is, we need to run the optimization procedure for every image we want to stylize.

4.2 SOBP Methodology

4.2.1 Loss Functions for Parameter Optimization

An essential component of any optimization algorithm is the loss function, employed to backpropagate its gradients to the parameters under optimization. This was formally introduced in Section 2.3, and we saw its application in the previous chapter. While in neural models we optimize the neural network's weights, in stroke optimization settings we optimize the stroke parameters directly. In the following experiments, three distinct types of loss functions—and their combinations—are used. The first is pixel loss, expressed by the L_1 distance between pixels of a reference image and the generated canvas. This loss function generally suffices for approximating the similarity between the reference image and the generated output.

Nonetheless, pixel loss alone may prove insufficient for high-fidelity or photorealistic approximations of the target image. To address this limitation, we introduce a second type of loss, termed perceptual loss. Let $V_{ij} = \{V_{ij}^1, ..., V_{ij}^k\}$ and $W_{ij} = \{W_{ij}^1, ..., W_{ij}^k\}$ be a set of k feature vectors extracted with a pre-trained classifier from an image I and a canvas C_T, respectively. We define perceptual loss as the cosine similarity between these feature vectors, formalized as follows:

$$\mathcal{L}_{perc} := \cos\theta = \frac{1}{K}\sum_{k}^{K}\sum_{ij}\frac{V_{ij}^k W_{ij}^k}{\left\|V_{ij}^k\right\|\left\|W_{ij}^k\right\|} \tag{4.1}$$

where i, j index the spatial dimensions of the feature maps V and W, and K the extracted layers from VGG16 trained on ImageNet [12]. After empirical tests, we use layers 1, 3, 6, 8, 11, 13, 15, 22 and 29. Minimization of this perceptual loss has been observed to effectively approximate the target image, capturing high-frequency details crucial for high-precision painting and thereby providing sufficient signal for the accurate representation of edges and finer details.

A third loss variant evaluated in this study utilizes the Contrastive Language-Image Pretraining (CLIP) model [8]. Trained on an extensive dataset of 400 million images

and their corresponding textual descriptions, CLIP has demonstrated a broad range of capabilities including classification, captioning, translation, and generation. Both the reference image and the generated canvas are encoded via CLIP's image encoder to obtain semantically meaningful representations. The cosine similarity between these representations is then minimized. This CLIP-based perceptual loss has recently been substantiated as effective for the generation of line drawings [1], and has been utilized in various language-driven models for image generation and painting [9, 2, 11, 6, 10].

4.2.2 Differentiable Rendering Engine

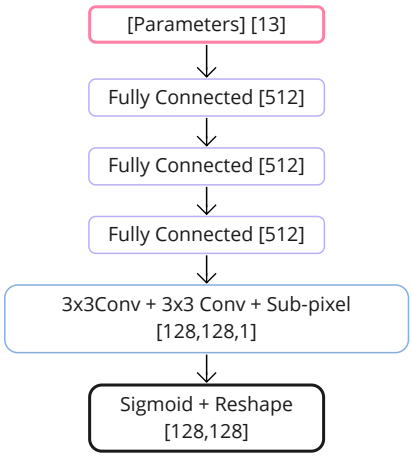


Figure 4.2: Neural renderer architecture provided by [4]

For gradient-based methodologies to function effectively, the backpropagation of the gradients from the objective function through the parameters under consideration necessitates the employment of differentiable and continuous functions throughout the computational graph. A disruption in this differentiability would consequently inhibit the optimization signal from propagating back to the parameters in question.

In the domain of computer graphics, the operation of converting vectorial informa-

tion into pixelated images is termed as rasterization. This operation is characteristically non-differentiable, posing a significant challenge for gradient-based optimization methods. However, the advent of deep neural networks has facilitated the development of differentiable approximations to traditional rasterization techniques. Recent research has pioneered differentiable renderers devoid of deep learning architectures [7], and some works, such as [2], have incorporated these engines into their frameworks.

In the experimental evaluations presented in this and following chapter, unless otherwise noted, we utilize a differentiable neural rendering engine as described in [4]. This engine is designed to convert parameters of quadratic Bézier curves (a diagram of this curve parameterization was shown in Chapter 2 in Figure 2.8) into an image grid of dimensions 128×128. The network architecture employed for this purpose is illustrated in Figure 4.2.

Technical Details. The differentiable neural renderer is trained over 500,000 optimization steps using the Adam optimization algorithm [5], initiated with a learning rate of 0.0001.[1] This learning rate is subsequently decayed at intervals of 200,000 iterations and is held constant beyond the 400,000th iteration.[2] During training, a batch size of

[1]The term "learning rate" refers to a numerical constant that determines the size of the steps taken during the optimization process in machine learning. Imagine you're in a hilly landscape in the dark, and you want to get to the lowest point. A large learning rate is akin to taking big leaps, which could either get you to the lowest point quickly or cause you to overshoot and miss it. A small learning rate is like taking tiny, cautious steps. It might take you longer to reach the lowest point, but you're less likely to miss it. The learning rate helps the algorithm balance between speed (how fast it learns) and accuracy (how well it learns).

[2]The term "400,000 iterations" refers to the number of forward and backward passes the deep learning model makes through the process of learning from the training data and adjusting its internal parameters from the error signal between its predictions and the actual outcomes. Each iteration is essentially a single round of this learning process. Think of it as a student reading through a set of questions and answers 400,000 times, each time getting slightly better at answering them. After a fixed number of iterations, the learning rate may be reduced to fine-tune the model's performance.

64 is adopted. At each optimization step, stroke parameters are sampled from a uniform distribution and rendered via OpenCV to generate what is considered the "ground truth" reference image. The discrepancy between this reference image and the network-generated image is quantified through the L_2 loss function.

Owing to the employment of this specific rendering engine, all resultant images from the conducted experiments intrinsically adhere to a stroke image size of 128×128, influencing the characteristics of the generated output to some extent.

4.2.3 Overview of Optimization Framework

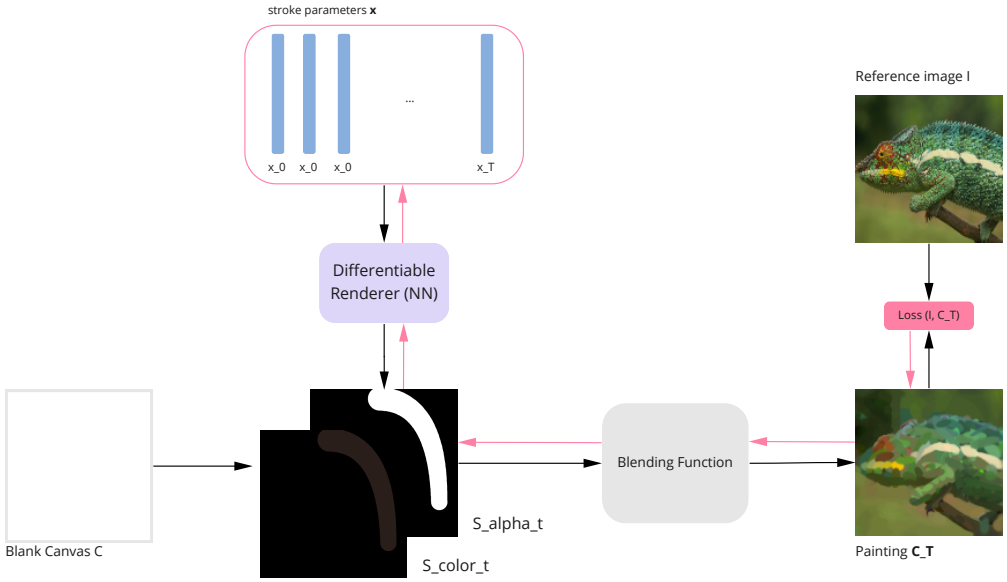


Figure 4.3: Stroke optimization Pipeline. Black arrows correspond to forward propagation, red arrows correspond to the backpropagation of the gradients.

In this section, I present a collection of optimization methods in increasing complexity. This facilitates an incremental exploration of the algorithms' inherent characteristics and functional capabilities. Common to all methodologies is a foundational structure composed by a canvas, differentiable renderer G, stroke model, and optimization and

objective function (an overview diagram of the framework can be seen in Figure 4.3).

These methods start with an arbitrary set of stroke parameters denoted as $\hat{\mathbf{x}}$, aiming to minimize the deviation from the optimal parameter set $\mathbf{x}^*$. For simplicity, the predefined number of strokes with which the algorithm initiates is referred to as the "stroke budget," or simply "budget." It is important to note that the final stroke budget may dynamically vary from the initial set, as will be elaborated upon in the context of specific algorithms that adaptively modify the stroke count based on the quality of the generated artwork at a given iteration.

The optimization algorithm employed across all methodologies is a specialized variant of gradient descent, delineated in Equation 2.2, known as the Adam optimizer [5]. To maintain experimental consistency, each methodology utilizes the same stroke model parameterized by a quadratic Bézier curve with 13 variables. Each stroke is formally represented as $\boldsymbol{s_t} = (x_0, y_0, x_1, y_1, x_2, y_2, r_0, r_2, t_0, t_1, R, G, B)$, where (x, y) are the Cartesian coordinates of the curve's control points, (r) denotes the radii at the endpoints, and (t) indicates the transparency levels. Typically, the transparency variables are set to unity, rendering all strokes opaque.

In the rendering pipeline, given a stroke budget of T, the parameters are either randomly or semi-randomly initialized in the configuration space of dimensions $T \times 13$. Subsequently, these parameters are fed into the differentiable rendering function G, which produces an array of T rasterized alpha strokes with dimensions 128×128. The color channels are obtained by element-wise multiplication of the alpha stroke with the respective RGB values. Each rasterized stroke is sequentially composited onto the canvas in accordance with the blending equation:

$$\text{canvas}_t = \text{canvas}_{t-1} \odot \text{alpha}_t + \text{stroke}_t \tag{4.2}$$

Here, t represents the temporal index within the stroke budget T, alpha_t refers to the alpha channel generated by the differentiable renderer G, stroke_t to the colored stroke, and canvas_{t-1} denotes the canvas state prior to the incorporation of the new stroke. The symbol $\odot$ in the above equation denotes the Hadamard Product, signifying the element-

Figure 4.4: Difference of quality of paintings using the neural differentiable renderer and the non-differentiable renderer. After optimization, stroke parameteres can be rendered at higher resolutions using non-differentiable renderers.

wise multiplication of the canvas at time $t-1$ with the alpha channel of the stroke at time t to composite the new stroke onto the canvas based on its transparency.

The initial canvas background is configured to be black across all experimental setups. The loss function $\mathcal{L}$ is computed between the reference image I and the finalized artwork C_T, followed by the backpropagation of the gradients for parameter optimization. This cycle iterates until the optimization algorithm converges to a stable state.

4.2.4 Preliminary Approach: Naïve Implementation

This section presents an initial approach to the stroke-based optimization painting (SOBP) problem, referred to as the "naïve" algorithm. We discuss the algorithm's specifications, stroke initialization methods, and the challenges encountered.

Algorithmic Constraints and Renderer Differences

The naïve SOBP algorithm, as outlined in Algorithm 1, operates on a 128×128 pixel canvas, a size constraint stemming from the differentiable renderer's resolution limitations. Training the renderer for higher resolutions is feasible but inefficient, thus the choice of a 128×128 pixel canvas. In theory, this should not be a critical constraint, as the optimized strokes can be rendered at any resolution size by employing a non-differentiable renderer after the optimization procedure is executed. However, the output strokes generated by the differentiable and non-differentiable renderers exhibit noticeable discrepancies in their edge sharpness and definition (Figure 4.4). Even when trained to convergence, the non-differentiable renderer falls short in visual fidelity. This is because the differentiable renderer outputs strokes with smoother gradients, e.g., edges are not sharp. Therefore, we confine the naïve implementation to the resolution provided by the differentiable renderer.

Algorithm 1 Naïve SOBP

Inputs:
ref image I, diff. renderer $\mathcal{G}$, budget T, loss $\mathcal{L}$,
optimizer ADAM, learning rate β
Initialize:
$\hat{\mathbf{x}} \in \mathcal{R}^{T \times 13} \sim U(0, 1)$
$C_0 \leftarrow 0_{[W \times H]}$
while $\hat{\mathbf{x}}$ has not converged **do**
  $\alpha_{T \times [W \times H]}, S_{T \times [W \times H]} \leftarrow \mathcal{G}(\hat{\mathbf{x}})$ ▷ Alpha mask and colored stroke
  for t in [T] **do**
    $C_t \leftarrow C_{t-1} * \alpha_t + S_t$ ▷ Blending function
  end for
  $g_{\hat{\mathbf{x}}} \leftarrow \nabla_{\hat{\mathbf{x}}} \mathcal{L}(C_T, I)$
  $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + \beta \cdot \text{ADAM}(\hat{\mathbf{x}}, g_{\hat{\mathbf{x}}})$
end while

Impact of Stroke Initialization

Stroke initialization plays a crucial role in algorithm performance. In its simplest form, we employ random initialization with parameters uniformly distributed in $[0, 1]$. A more

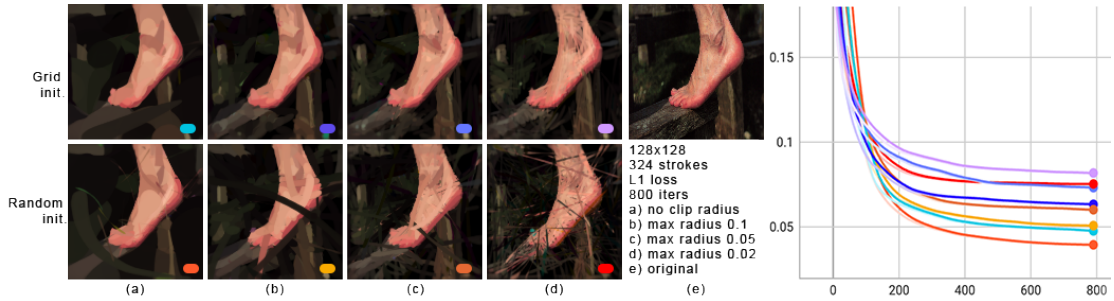


Figure 4.5: Impact of random and grid stroke initialization in the final painting. Top row corresponds to a grid stroke initialization, and bottom row corresponds to random stroke initialization. Columns correspond to clipping the maximum stroke width the model is allowed to paint with: (a) no constraint, (b) maximum width 0.2*canvas size, (c) 0.1*canvas size and (d) 0.04*canvas size. (e) Input and optimization details. All paintings use 324 strokes, and L1 as pixel loss function. We let the optimization run for 800 iterations. Right: L1 loss graph over 800 iterations.

advanced technique involves grid-based initialization, anchoring strokes to a grid and modulating stroke lengths and positions via Gaussian distribution. To illustrate, Figure 4.5 presents a series of examples showcasing paintings rendered at 128×128 pixels with different stroke initializations and stroke widths. These instances demonstrate that grid-based initialization, combined with Gaussian-controlled stroke lengths, substantially simplifies the optimization process. Interestingly, while random initialization may result in lower L_1 losses, grid-based initialization typically yields visually superior outputs and simplifies optimization.

Optimization Dynamics and Challenges

As depicted in Figure 4.6, the naïve algorithm encounters difficulties optimizing longer strokes, although color optimization is more manageable. This challenge underscores the complexity of stroke length adjustment in SOBP.

Influence of Loss Functions

The selection of loss function significantly affects the final output quality. Our experimental analysis explores the utility of different loss functions, as summarized in Figure 4.7. While L_1 loss is generally effective for basic image reconstruction, it fails to

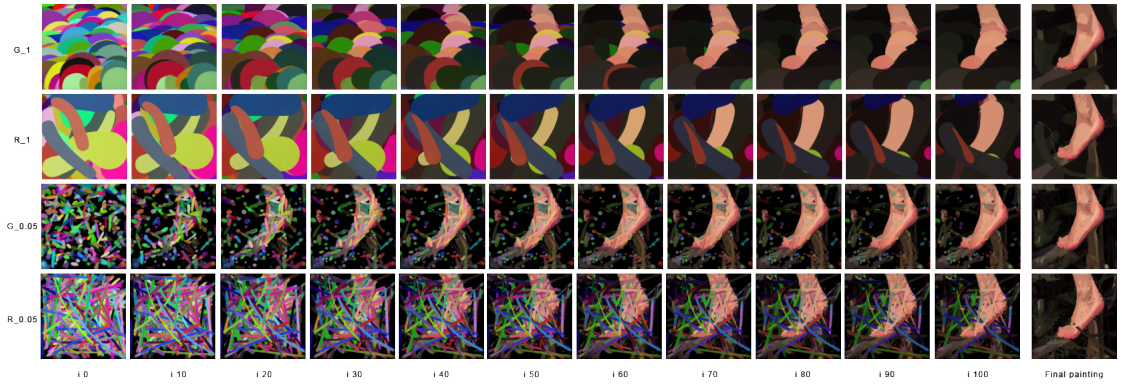


Figure 4.6: Comparison of the painting progress across 2 parameters: grid and random initialization, represented as G and R, respectively, and maximum stroke width, 1 and 0.05 (100% and 5% of the canvas width). The optimization process is shown every 10 iterations on each column for the first 100 iterations. The final painting is shown in last column.

capture finer details. Incorporating perceptual or CLIP loss enhances detail reconstruction, with perceptual loss particularly effective when well-calibrated.

Figure 4.8 shows a comparison of results using different losses between strokes initialized in grid (left) and strokes initialized randomly (right). The latter approach struggles across different losses, in contrast to grid initialization, which presents a smoother optimization landscape.

In essence, the naive algorithm establishes a foundational understanding of the intricacies in SOBP. It underscores the importance of strategic stroke initialization and careful loss function selection to attain desirable outcomes in artistic stylization.

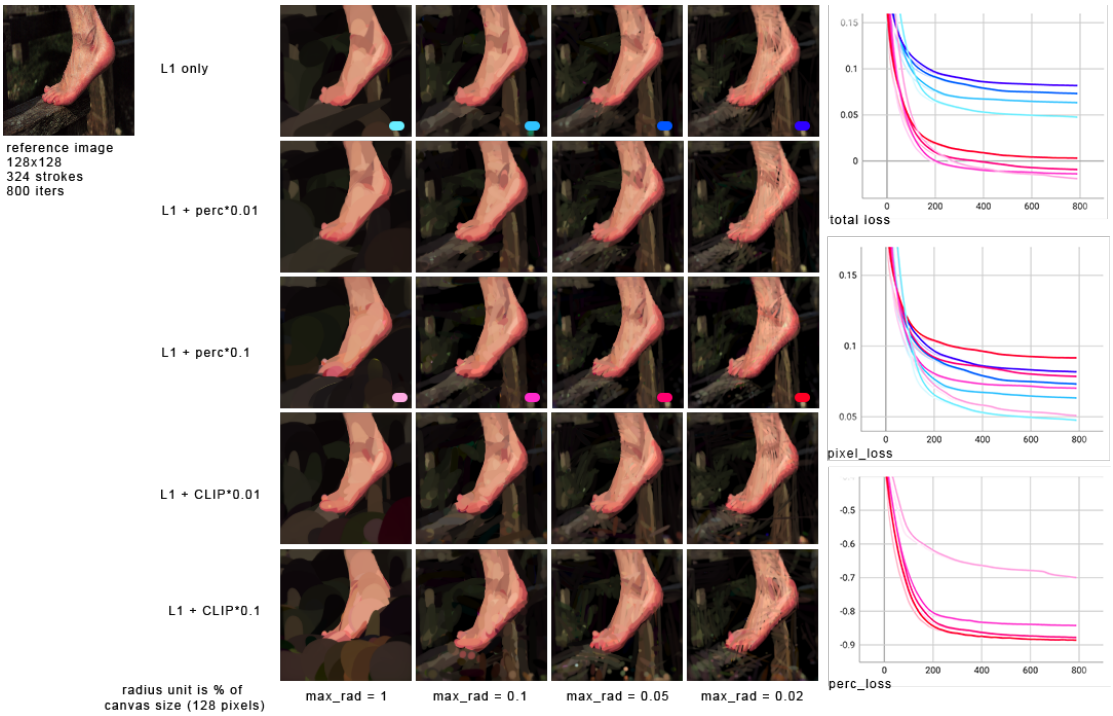


Figure 4.7: Difference in the impact of loss functions. We show the difference in the outcome when using L1, perceptual loss and CLIP loss. A weight of 0.01 seems very low for either perceptual or CLIP to make an impact. However, a weight of 0.1 makes the perceptual loss define some fine details than L1 loss alone cannot (see definition in the toes). CLIP loss introduces some noise in the painting.

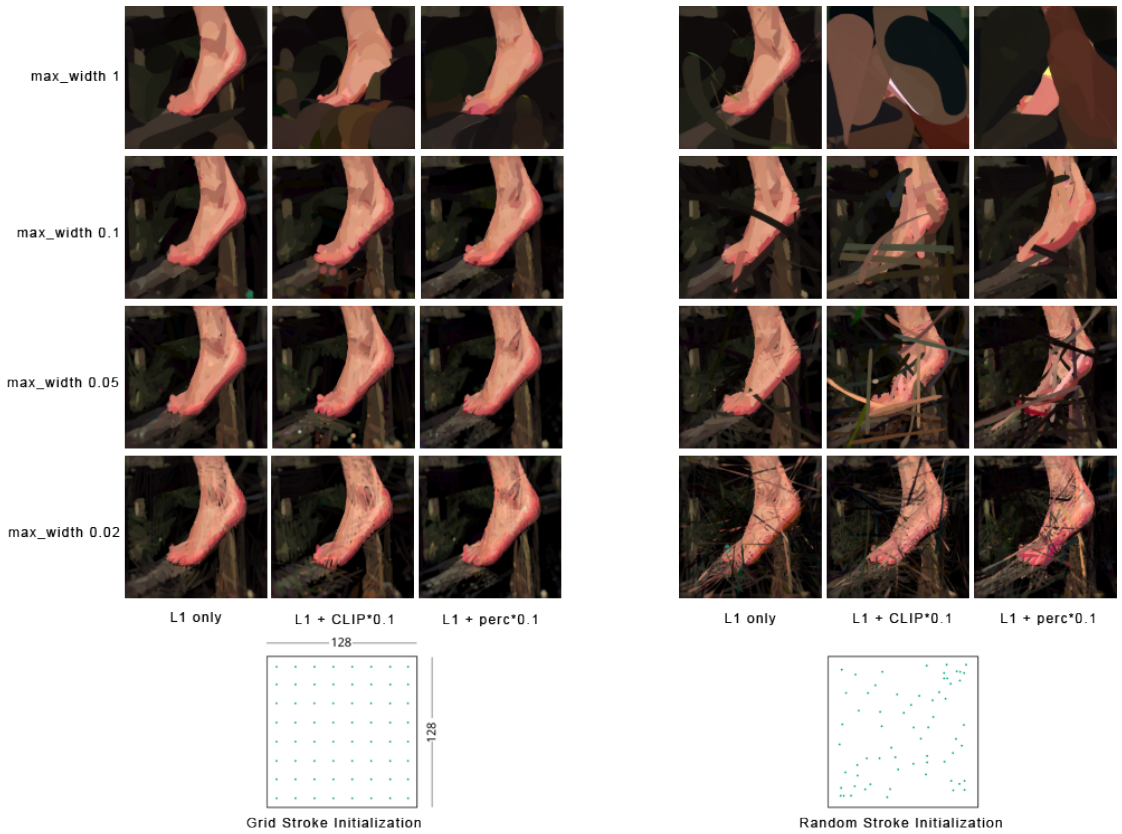


Figure 4.8: Comparison between two stroke initialization methods. Left block of images is initialized with a grid of strokes. Right block of images is initialized with random stroke initialization. A random initialization increases the complexity of the optimization landscape.

4.2.5 Hierarchical Coarse-to-Fine Algorithm

This section outlines a more advanced algorithmic framework developed to overcome the limitations of the naïve algorithm, particularly regarding resolution constraints and stroke placement efficiency. Drawing inspiration from the hierarchical approach in Hertzmann's pioneering work [3], our method implements a multi-level strategy, varying in detail granularity and stroke width, to facilitate more precise and efficient optimization.

Conceptual Framework and Design Principles

Our algorithm operates through a series of refinement "levels,"[3] each characterized by a distinct level of detail granularity and a corresponding set of strokes. This hierarchy begins with a coarse representation using broader strokes and progressively advances to finer layers with a denser array of narrower strokes. This structure adheres to the principle that broad strokes capture global features effectively, while finer strokes are adept at rendering detailed nuances.

Figure 4.9: Coarse to fine approach, 512x512 painting. (a) Coarse detail level with 25 strokes per patch and no stroke width clipping. (b) Medium detail level with 36 strokes per patch and width clipping of 10% of patch size. (c) Fine detail level with 81 strokes per patch and width clipping of 5% of patch size.

[3] Also called "painting passes." We use these two terms interchangeably

Optimization Strategy Across Levels

In this multi-level approach, each refinement phase inherits and builds upon the strokes optimized in the previous level. Starting with the coarsest layer, the algorithm methodically optimizes stroke parameters until reaching a convergence point. It then ascends to the next level, adding more detailed strokes to the existing composition. This layered optimization ensures a systematic enhancement of details, maintaining computational efficiency throughout the process. By segmenting the painting into distinct levels or passes, the optimization process becomes smoother and more manageable. At each level, the algorithm focuses on a narrower set of stroke parameters, all aimed at capturing the same level of detail. This approach eliminates the need for all strokes in a painting to simultaneously address varying degrees of detail, from broad backgrounds to intricate, high-frequency elements. Consequently, this layered method not only streamlines the optimization but also ensures a consistent quality of detail across different areas of the artwork.

This stratified approach to optimization not only mirrors computational efficiency but also reflects traditional painterly practices. Just as an artist may first block out the large forms with broad strokes before attending to the finer details in subsequent layers, the algorithm similarly progresses from the general to the specific. This method of working in passes, a technique often used in classical painting, allows for an incremental refinement of the artwork. Each layer adds complexity and depth, enabling the artist—or in this case, the algorithm—to develop a rich, textured image that gradually evolves into a finished piece. The parallels between this multi-level optimization and time-honored artistic techniques underscore the synergy between technology and traditional artistry, offering a digital counterpart to the nuanced process of creating a painting.

Empirical Demonstrations

Figure 4.1 and Figure 4.9 provide illustrative examples of the algorithm's coarse-to-fine progression. These figures highlight the method's effectiveness in generating refined

artistic representations, showcasing how strokes are strategically employed across various levels to incrementally enhance image detail and overall aesthetic quality.

The Cost of Higher Resolution

As we saw in the naïve approach, the resolution constraints linked to the differentiable renderer significantly limit the achievable level of detail and, consequently, the broader applicability of the algorithm. To address this, I have developed an adaptive, patch-based optimization strategy, specifically tailored to manage the complexities associated with generating higher-resolution outputs.

Patch-Based Optimization Scheme This approach circumvents resolution constraints by dividing the reference image into multiple 128×128 patches. These patches are then optimized concurrently in a batch process. After optimization, we employ a blending function to seamlessly merge these patches, resulting in a cohesive high-resolution canvas. This patch-based strategy has proven to be highly efficient for resolutions scaling up to 1024×1024[4] . The patch-based approach is somewhat common in the literature, as it enables paintings to achieve a larger scale. For instance, Learning to Paint [4], a reinforcement learning method for stroke-based painting, can achieve larger scale paintings by dividing the original image into equal 128×128 patches, and painting each patch individually. However, this method produces some visible seams. Another optimization method in the literature, Stylized Neural Painting [13], employs a different technique based on a multi-scale process. Albeit different from a patch-based approach, both share the same goal: to break down the optimization process into more manageable stages to soften the optimization landscape.

Empirical Analysis of Computational Cost Figure 4.10 presents an empirical comparison of the computational costs, examining the impact of increasing both the number

[4]Experiments have been tested using a single Nvidia RTX A6000 GPU.

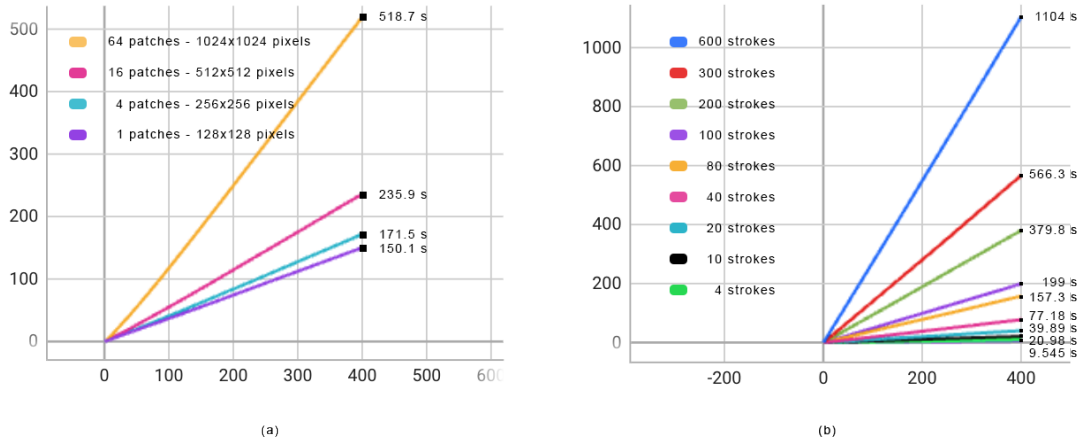


Figure 4.10: Analysis of Computational Costs in Patch-Based Painting. (a) Demonstrates the non-linear computational cost associated with the painting-by-patches approach using a 100-stroke budget per 128x128 patch over 400 optimization steps. Doubling the image size does not result in a linear cost increase. (b) Illustrates the near-linear computational cost incurred with increasing stroke budget.

of patches (left) and the stroke budget (right). The analysis shows that a fourfold increase in resolution results in a relatively moderate 1.56 times increase in computational demand.

Bottlenecks in Stroke-Based Rendering Algorithms A key insight from our empirical studies is that the primary computational bottleneck in Stroke-Based Rendering (SBR) lies in the rendering of strokes. We observe that computational complexity scales almost linearly with the number of strokes, attributable to the sequential nature of stroke rendering. In this process, each stroke is rendered individually, thereby incrementing computational time proportionally. This finding underscores the need for innovative approaches in SBR algorithms to enhance efficiency and facilitate higher resolution outputs.

Painting by Patches using a Coarse-to-Fine Strategy

This subsection explains our advanced patch-based painting strategy, which implements a hierarchical coarse-to-fine approach to render detailed and nuanced depictions of the

input image I. We describe the strategy's mechanics and theoretical underpinnings, referencing Algorithm 2 for detailed algorithmic steps.

Algorithm 2 Coarse-to-fine patch-based SOBP

Inputs:
ref image I, diff. renderer $\mathcal{G}$, number of patches N, list of budgets $[T]$,
list of max stroke widths $[R]$, composite Z, overlap v, loss $\mathcal{L}$,
optimizer ADAM, learning rate β
$H, W = N * 128 - (v * (N - 1))$
Initialize:
$C_{\text{general}} \leftarrow 0^{[N^2 \times W \times H]}$
for r in [R] **do**
  Initialize:
  $\hat{\mathbf{x}} \in \mathcal{R}^{N^2 \times T \times 13} \sim U(0, 1)$
  $C_{r_0} \leftarrow 0^{[W \times H]}$
  while $\hat{\mathbf{x}}$ has not converged **do**
    $\alpha_{T \times [W \times H]}, S_{T \times [W \times H]} \leftarrow \mathcal{G}(\hat{\mathbf{x}})$ ▷ Alpha mask and colored stroke
    for t in [T] **do**
      $C_{r_t} \leftarrow C_{r(t-1)} * \alpha_t + S_t$ ▷ Blending function
    end for
    $g_{\hat{\mathbf{x}}} \leftarrow \nabla_{\hat{\mathbf{x}}} \mathcal{L}(C_{r_t}, I)$
    $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + \beta \cdot \text{ADAM}(\hat{\mathbf{x}}, g_{\hat{\mathbf{x}}})$
  end while
  $C_{\text{general}} \leftarrow Z(C_{r_T})$ ▷ Composite function
end for

Partitioning and Resizing of Input Image The algorithm divides the input image I into $N_h \times N_w$ equally sized patches, each measuring 128×128 pixels. The resultant dimensions of I are denoted as $I_h = N_h * 128 - (v * (N_h - 1))$ and $I_w = N_w * 128 - (v * (N_w - 1))$, where v is the extent of overlap between adjacent patches, as illustrated in Figure 4.11. The overlap v serves as a tuning parameter to optimize the visual seamlessness of the final painting, contingent upon the blending function employed.

Hierarchical Stroke Allocation and Optimization Following Hertzmann's methodology, each detail level is assigned a specific stroke budget T and maximum stroke

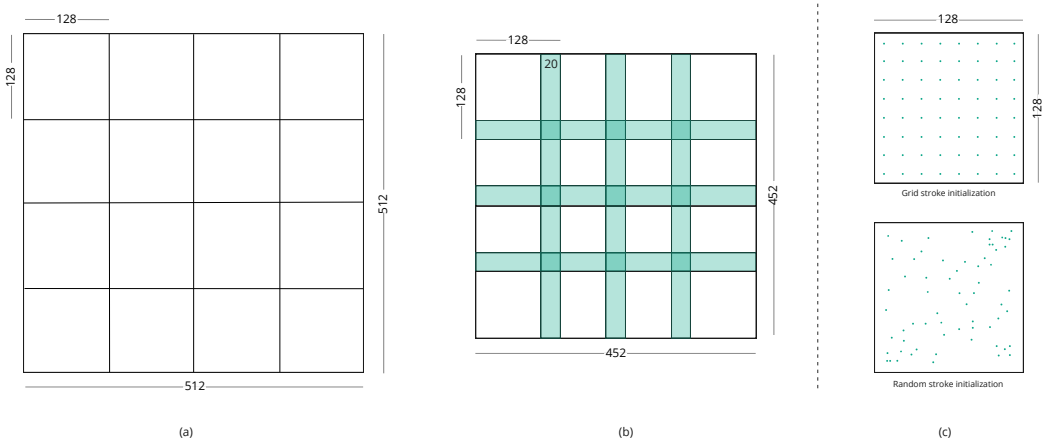


Figure 4.11: Canvas composition by patches. (a) Patches are organized into a structured grid without overlaps. (b) Example of an organization of patches in a structured grid with overlaps of 20 pixels. (c) Initialization of strokes before being optimized: strokes are evenly distributed across the x and y axis of the patch (upper part). Strokes are randomly sampled from a uniform distribution (bottom part).

width. These parameters are adjusted at each layer to yield a realistic painting. Stroke parameters corresponding to each level are initialized, and the patches are subsequently rendered and optimized in batches. It is important to note that the loss function is computed on a per-patch basis rather than globally, e.g., measuring the entire input image against the entire painting. This process is more efficient as it takes the advantage of the power of parallel computation of GPUs.

Dynamic Stroke Budgeting Across Patches To enhance complexity and realism, the algorithm dynamically allocates strokes across patches. The variable allocation, requiring batch computation, involves padding sequences to form a rectangular matrix. After rendering the coarsest layer, the algorithm assesses loss differentials for each patch compared to the reference image. Patches with loss below a certain threshold have their stroke budget reduced, promoting efficient computational resource allocation by focusing on detail-needy areas.

Evaluative Observations Our empirical evaluation aimed to assess the impact of dynamic stroke budgeting on the painting's naturalistic quality. However, the findings indicate no significant difference in visual quality between uniform and variable stroke budgeting (Figure 4.12). This observation suggests that while dynamic budgeting adds a layer of complexity, its influence on the final painting's perceptual quality may be limited. We will see in the next chapter how we build upon this finding to effectively convey a more naturalistic appeal.

Figure 4.12: Comparison of paintings with uniform and variable stroke counts per patch. (a) Original image with a 20-pixel overlap. (b) Uniform stroke painting: 400 strokes at the coarsest level (25 per patch), 576 at the medium level (36 per patch), and 1296 at the finest level (81 per patch), totaling 2272 strokes. (c) Variable stroke painting: starts similarly at the coarsest level, but uses fewer strokes at the medium (363 strokes) and finest levels (1023 strokes), totaling 1786 strokes.

Exploring Variability in Painting Styles

Our algorithm is versatile enough to generate paintings with different style variation at high resolutions. This flexibility is achieved by varying several key parameters—such as the number of strokes, stroke widths, and choice of loss functions—within a consistent stroke model framework. Visual examples of these styles can be seen in Figure 4.13.

Exemplified Styles We present four distinct painting styles, each illustrating the algorithm's range:

- **Abstract Representation (a):** With a reduced stroke count, this style abstracts the input image's essence, focusing on basic shapes and colors to convey meaning.
- **Minimalist Approach (b):** This style further constrains stroke width and employs a two-pass painting method as opposed to the conventional three-pass approach, yielding a pared-down yet effective rendition.
- **Realistic Rendering (c):** Utilizes a hybrid loss function comprising both L_1 and perceptual loss to produce a painting that closely mimics the appearance of the original image.
- **Expressionist Interpretation (d):** Employing a weighted combination of L_1 and CLIP loss, this style lends itself to heightened emotional or symbolic interpretation, as it slightly departs from mere reconstruction due to the impact of the CLIP loss.

4.3 Conclusion

In this chapter, we have established the foundation for an optimization-based approach to stroke-based painting. The chapter began with a naïve version of an optimization algorithm, serving as a testbed to analyze the influence of critical parameters such as stroke count, loss functions, and stroke parameter initialization. Through a series of experiments, we observed the impact of different loss functions and, crucially, the visual differences between random and grid-based stroke initialization on the painting process and the final artwork. While insightful, the naïve approach highlighted certain limitations, notably in resolution capabilities.

To overcome these challenges and drawing inspiration from Hertzmann's pioneering work, we introduced a hierarchical coarse-to-fine algorithm. This approach effectively manages painting at larger scales and smooths the optimization landscape through a layered, coarse-to-fine process. Experimenting with various combinations of optimization strategies, parameters, and loss functions, we observed the algorithm's capacity to

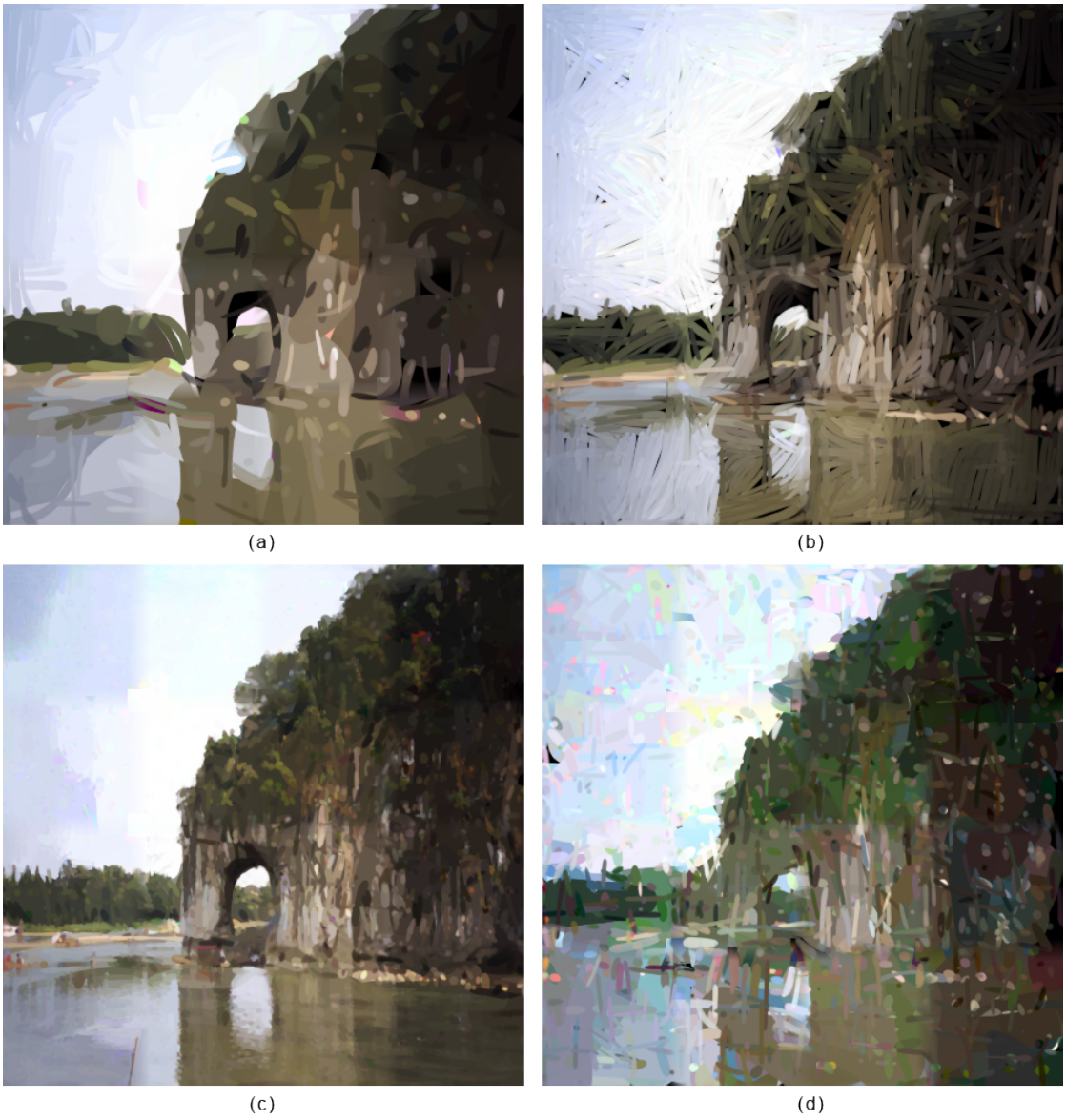


Figure 4.13: Different stylistic outputs given by different settings (a) Painting using only 800 strokes and L_1. (b) 2 passes using only thin strokes (c) Realistic painting using more than 2000 strokes (d) Expressionist painting using a combination of L_1 and CLIP loss

produce subtly different styles. However, the primary focus of this chapter was not on stylization per se, but on providing an empirical analysis of the method's effectiveness and its potential for enhancing controllability and further stylization.

Limitations and Discussion Despite its versatility, the current algorithm exhibits constraints in terms of the variability and depth of stylization. To achieve more compelling and diverse styles, a greater emphasis on controllability is necessary, as the algorithm lacks mechanisms to precisely control stroke placement, an aspect crucial for artistic expression. The presented algorithm uniformly divides the patches across the entirety of the input image, providing no specific control on where to lay the strokes.

The next chapter builds upon this groundwork, concentrating on controllability and stylization. We aim to introduce and explore innovative techniques, inspired by human artistic processes, to significantly expand the algorithm's stylistic repertoire. This endeavor will not only enhance the diversity of styles achievable but also improve the overall artistic quality of the generated paintings, pushing the boundaries of computational creativity in stroke-based rendering.

References

[1] Caroline Chan, Fredo Durand, and Phillip Isola. “Learning to generate line drawings that convey geometry and semantics”. In: *arXiv preprint arXiv:2203.12691* (2022).

[2] Kevin Frans, LB Soros, and Olaf Witkowski. “CLIPDraw: Exploring Text-to-Drawing Synthesis through Language-Image Encoders”. In: *arXiv preprint arXiv:2106.14843* (2021).

[3] Aaron Hertzmann. “Painterly Rendering with Curved Brush Strokes of Multiple Sizes”. In: *Proc. SIGGRAPH*. 1998.

[4] Zhewei Huang, Wen Heng, and Shuchang Zhou. “Learning to Paint With Model-based Deep Reinforcement Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[5] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[6] Gihyun Kwon and Jong Chul Ye. “Clipstyler: Image style transfer with a single text condition”. In: *arXiv preprint arXiv:2112.00374* (2021).

[7] Tzu-Mao Li et al. “Differentiable vector graphics rasterization for editing and learning”. In: *ACM Trans. Graph.* 39.6 (2020).

[8] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8748–8763.

[9] Aditya Ramesh et al. “Zero-shot text-to-image generation”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8821–8831.

[10] Aditya Sanghi et al. “Clip-forge: Towards zero-shot text-to-shape generation”. In: *arXiv preprint arXiv:2110.02624* (2021).

[11] Peter Schaldenbrand, Zhixuan Liu, and Jean Oh. “StyleCLIPDraw: Coupling Content and Style in Text-to-Drawing Synthesis”. In: *arXiv preprint arXiv:2111.03133* (2021).

[12] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[13] Zhengxia Zou et al. “Stylized Neural Painting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 15689–15698.

Chapter 5

Segmentation-Based Parametric Painting

MANUEL LADRON DE GUEVARA, MATTHEW FISHER, AARON HERTZMANN. UNDER REVIEW CVPR 2024

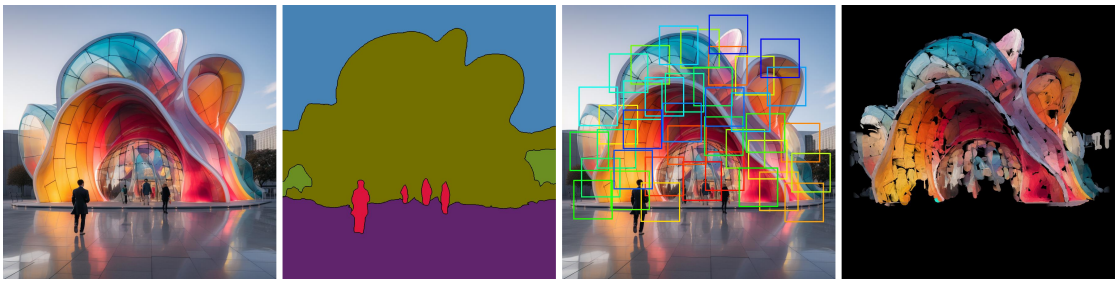


Figure 5.1: Our method provides finer control over the canvas via the use of a segmentation network and dynamic attention maps to guide the painting process.

This chapter introduces a novel image-to-painting method that facilitates the creation of large-scale, high-fidelity paintings with human-like quality and stylistic variation. To process large images and gain control over the painting process, we introduce a segmentation-based painting process and a dynamic attention map approach inspired by human painting strategies, allowing optimization of brush strokes to proceed in batches over different image regions, thereby capturing both large-scale structure and fine de-

tails, while also allowing stylistic control over detail. Our optimized batch processing and patch-based loss framework enable efficient handling of large canvases, ensuring our painted outputs are both aesthetically compelling and functionally superior as compared to previous methods, as confirmed by rigorous evaluations. Furthermore, when comparing with previous optimization methods [40], our method is significantly faster (4x) for the same number of strokes (1600) and resolution size (1024x1024).

Code available at: https://github.com/manuelladron/semantic_based_painting.git

Project Website: Semantic-based-painting

5.1 Introduction

Painting is the world's oldest technology for creating visual imagery, and a continuing source of fascination and artistic expression. Representational painting is a complex process where an artist creates a picture of a realistic scene from a series of brush strokes. Artists use a variety of strategies to determine where and how to paint, and different strategies produce different painting styles. For example, a painter might work in a coarse-to-fine manner, beginning with general shapes and then refining details, or in a semantic-region-based manner, painting areas such as sky, mountains, and buildings sequentially; the latter strategy may produce a more segmented style than the former, with different styles for different objects.

These strategies, in part, are a function of human working memory and foveal vision: unlike optimizers, artists cannot simultaneously work on every part of a picture in parallel. They work piece-by-piece, focusing on one region or aspect of a painting at a time [4, 7, 28, 36, 35, 3], before moving on to the next.

As we have previously discussed, the computational *stroke-based painting* problem is: given an input photograph, generate a painting to represent the photograph, parameterized by a set of brush strokes, which themselves are parameterized by their colors, control curves and thicknesses, with optional texturing [10, 11]. These methods allow styles to be defined directly in terms of brush stroke properties, with further additional

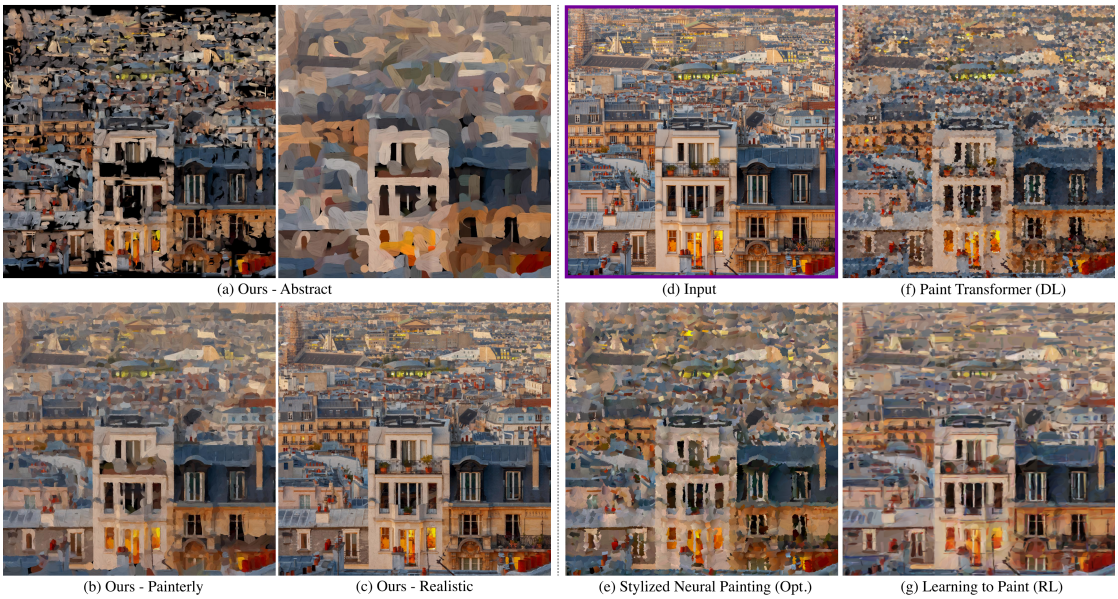


Figure 5.2: Comparison of our method against previous work on parametric painting algorithms. Panels (a) through (c) demonstrate our algorithm's capability to render scenes in various styles: abstract with simplified forms and abstract with a heavy emphasis on thick loose strokes, texture, and a lack of clear form (a), painterly impressionistic style with dynamic strokes and emphasis on effect (b), and realistic style capturing both micro and macro structures with high fidelity (c). These are juxtaposed against existing methods: Stylized Neural Painting (SNP) [40] (e), Paint Transformer (PT) [23] (f), and Learning to Paint (L2P) [15] (g), which exhibit limitations such as imprecise edge depiction and a lack of stylistic diversity. SNP and PT use rigid, blocky strokes that are unable to capture fine details, edges and shapes. L2P captures details better, although yields blurry results with visible seams at large resolutions. The input image is displayed in panel (d) for reference. This comparison underscores our method's superior ability to balance detail retention with stylistic expression. Zoom-in is encouraged to see the nuances of each work.

user control possible on the strokes themselves, e.g., [27]. Current methods treat this as an optimization problem: using a limited set of brush strokes, approximate the input image as closely as possible [5, 12, 25, 15, 40, 23, 30]. By varying the level of approximation, a user should be able to select from more- or less- abstract styles, and selecting different placement strategies should produce qualitatively different styles. Moreover, robotic painting systems require explicitly-defined brush strokes, e.g., [30, 21, 8, 1], which are not available with pixel-based stylization techniques.

Despite recent progress, existing methods for this problem suffer from a limited

ability to perform this optimization well for large images and limited user control. State-of-the-art methods either jointly optimize a large collection of strokes, or train a network to generate them. These approaches are limited by the problem of generating strokes uniformly for an input image, which becomes impractical as the input size increases. Hence, these methods do not generate high-fidelity results for even moderate-sized input images, thereby limiting the level of abstraction and control [15, 23, 40, 34]. Moreover, they lack mechanisms to mimic varied artistic strategies, such as applying different styles for different scene elements, and coarse-to-fine painting.

Inspired by these observations, we propose a new approach to generating stroke-based paintings from photographs. We first structure the entire painting process in layers by isolating different semantic parts of the canvas. Then dynamic attention maps guide the painting process to refine areas that need more detail, and differentiable strokes are optimized sequentially within each region. This sequential optimization process allows us to generate strokes over a large image canvas efficiently, in contrast to methods that produce pictures as a single joint optimization. It also approximates human painting strategies that operate sequentially on different image regions, rather than jointly optimizing all strokes at once. Several objective function parameters are provided to control the style, including limiting the number of strokes and level of detail for each semantic region, providing a continuum from realism to abstraction. Quantitative and qualitative evaluations show that our method achieves a greater range of high-fidelity reconstructions to abstract imagery, while also producing high-quality results at large image sizes. Figure 5.2 shows a comparison of the styles generated by our method against previous work.

5.2 Related Work

Early stroke-based-rendering (SBR) approaches varied from procedural, rule-based systems without optimization [10, 22], including layering strokes in a coarse-to-fine fashion. Procedural methods effectively produce captivating paintings, but their rigidity in

stroke placement is a notable limitation. In principle, optimization-based SBR overcomes this constraint, offering greater adaptability in stroke positioning and more accurately encoding artist choices [12]. Such methods have evolved from heuristic-based optimization to more advanced techniques such as EM-like packing algorithms for arranging non-overlapping stroke primitives [11, 29], with applications extending to 3D model stylization [24, 32]. Recent SBR methods employ deep networks trained to simulate a painting agent through reconstruction losses, involving pixel and perceptual losses [17, 38], facilitated by differentiable rendering engines [20]. However, the difficulty of optimizing large collections of strokes over an entire image remains challenging.

Reinforcement Learning (RL) has become prominent in SBR for simulating decision-making in painting agents [37, 5, 25, 15, 16, 34, 31], with diverse applications ranging from detailed reconstructions to robotic painting. SPIRAL [5] and its enhanced version SPIRAL++ [25] are noteworthy for their adversarial training algorithms, despite producing less defined images. Nonetheless, both methods struggle with interpretability, style control, and accurate input reconstructions. Some RL methods focus on accurate depiction [15, 34]. Huang *et al.*'s method [15] needs thousands of tiny strokes to reconstruct an image, although becomes blurry for high resolution images, and it is stylistically limited.

Deep learning methods like Paint Transformer [23] utilize a CNN-Transformer architecture to predict stroke sequences, showing promise in generalization despite pattern repetition in results. RNN-based models have also been explored for distinct line drawing and sketching [9, 19, 39, 26], with our method adopting similar attention mechanisms [26].

Additionally, recent works have leveraged direct differentiable optimization for stroke placement [20, 40], with style variations often induced by style-transfer techniques in pixel space [6, 17].

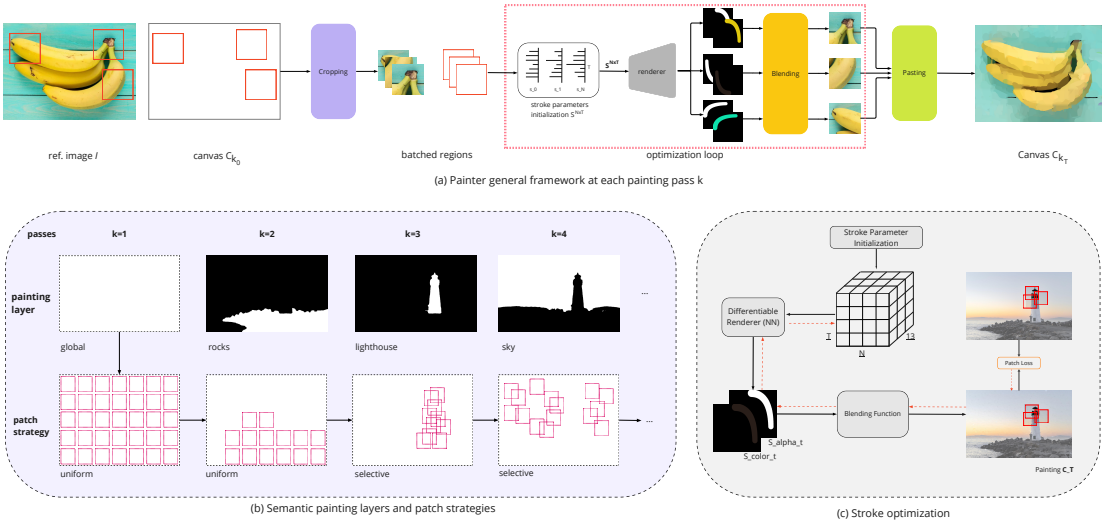


Figure 5.3: Overall Painting Framework. Our framework models the painting process as a set of sequential painting passes. (a) Pipeline of each painting pass k, which is composed by four parts: attention patches, cropping, optimization, and pasting. (b) Semantic layers and patch strategies. (c) Stroke optimization module with patch-loss.

5.3 Method

Our goal is to efficiently optimize a collection of stroke parameters that define a large-scale painting from a given image, capturing micro and macro structures with high fidelity, and providing control for stylistic variation. Our system takes as input an image X, and style parameters, encapsulated by four style presets: "realistic," "painterly," "abstraction," and "expressionist." Then, our objective is to optimize a parameter vector $\bar{Y}$ of strokes, which, when rendered on a canvas C, will approximate the given image X as a stylized painting. Our scalable solution accommodates varying sizes and aspect ratios through batch-patch optimization, cropping the input and canvas into a set of 128×128 patches, which are reassembled after the optimization process to match the original image scale.

Inspired by human painting techniques, and to provide more control over the painting process, we first use a segmentation network that decomposes the painting into semantic areas, and then employ dynamic attention maps to determine where to paint. An

overview of our framework is shown in Figure 5.3. For information on the parametrization of styles and implementation details, refer to the supplemental material presented in Appendix B.

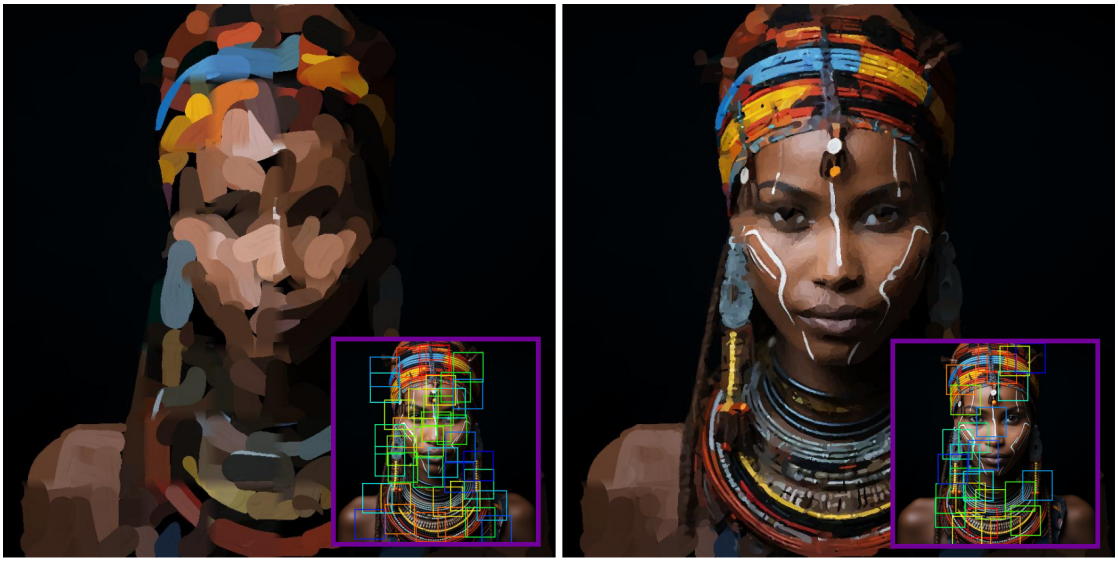
5.3.1 Layered Painting and Patch-Based Optimization

The set of paint strokes is separated into K layers, one for each input semantic region. Applying separate painting styles or details for different regions allows stylistic control over separate scene elements. For each layer k, the canvas is divided into a set of 128×128 patches, where all stroke parameters in each patch are batch-optimized; different layers may have different numbers of patches, as a function of the semantic layer's area.

The optimization is performed in a series of P painting passes, designed to progress from general, broader strokes to intricate, finer details, akin to the approach of a traditional artist [13]. For each painting pass p, our method applies an optimization phase to each of the K semantic layers, following a coarse-to-fine procedure (see Figure 5.4).

Let P denote the total number of painting passes, and let's denote the number of semantic layers as K. We define k_p as the layer index being refined in pass p, where $k_p \in \{0, 1, 2, ..., K-1\}$. Overall, our goal is to minimize a reconstruction loss L_{rec} summed over the entire image. We specify this loss later in the chapter. At a given pass, the goal is to optimize just the strokes that appear in some layer k. The objective for each layer k during optimization is to optimize a subset of the loss just for the patches and strokes in this layer: $\arg\min_{\bar{y}_k} \sum_{i=1}^{N_k} \mathcal{L}_{rec}(X_{k,i}, C_{k,i}(\bar{y}_k))$, where $X_{k,i}$ is the i-th patch of the semantic region k of the input image, and $C_{k,i}(\bar{y}_k)$ denotes the i-th patch of the canvas at layer k, parameterized by the stroke parameters $\bar{y}_k$. The number of layers K adapts to each painting based on the unique semantic content of the input image, and it is determined automatically by the segmentation network. In cases where the input image lacks semantic distinction, or if semantic assistance is disabled, a single layer approach is adopted ($K = 1$), involving the entire canvas. We use the DETR model [2], which consists of a CNN (ResNet) backbone followed by an encoder-decoder Transformer, trained on the COCO Panoptic Segmentation task.

Starting with the input image X, we derive all segmentation layers and compute each binary mask, which is then used to initialize the vector of stroke parameters $\bar{Y}_{k_p}$. Upon the completion of the optimization steps, strokes that fall outside the designated segmented area are filtered out based on a threshold γ.



(a) Coarse layer (b) Fine layer

Figure 5.4: Coarse-to-fine approach for a painterly portrait of a woman. (a) Initial coarse layer lays foundational strokes for a later refinement (b), guided by dynamic attention maps (insets).

Dynamic Attention Maps. While painting, any human painter must continually decide where next to paint brush strokes, and different painters will make these choices differently. Our system represents this process by dynamic attention maps, with two distinct strategies: uniform, and selective (see Figure 5.3 (b)). The former is based on a uniform distribution across the canvas, yielding patches in a consistently-spaced grid covering the entire image (or a portion of the image corresponding to a semantic layer k).

The latter identifies the V patches with the highest error for attention, where V is a user-determined style parameter. This error is computed as L_1 loss between the canvas

and image patches. We found that just using L_1 loss is enough to provide a good painting guidance, making the process more efficient in comparison to using perceptual losses.

The uniform approach ensures an unbiased and uniform distribution of detail throughout the painting, and is typically reserved for the foundational stages of a painting, generally producing higher realistic painting than the selective approach, which generates a more human-like, organic painterly style paintings.

Stroke Initialization, Renderer, and Blending. We use the stroke parameter representation and differentiable renderer, G, provided by [15] as presented in Section 4.2.2. To recap, each stroke is parameterized by a 13-dimensional tuple $\{x_0, y_0, x_1, y_1, x_2, y_2, r_0, r_2, t_0, t_2, R, G, B\}$ that encodes the start, middle and end points of a quadratic Bézier curve, radii and transparency at the start and end points, and RGB color of the stroke. For each 128×128 patch in the canvas, we distribute T stroke parameters evenly in a regular array.

Strokes are composited into a patch with soft blending: $M \leftarrow M_{t-1} \odot (1 - G(s_j)_a) + G(s_j)_c$ where s_j are the parameters of the j-th stroke, M is a canvas patch, and $G(s_j)_a$ and $G(s_j)_c$ are the alpha channel and the stroke given by the differentiable renderer G, respectively. The symbol $\odot$ denotes the element-wise multiplication operation. However, for semantic layers, the corresponding binary mask α_k is used: $M_k \leftarrow M_{k_{t-1}} \odot (1 - G(s_j)_a) + G(s_j)_c \odot \alpha_k$

5.3.2 Optimization and Loss Functions

At a given pass p, for each layer k in such a pass, we optimize all stroke parameters from all N_k patches in batch. That is, we optimize a matrix $S_k^{N \times T \times 13}$ altogether, where T is the stroke budget, and instead of computing the loss function over the entire canvas, we use a patch loss described earlier (see Figure 5.3 (c)). This allows us to perform these optimizations much more efficiently than optimizing over the entire canvas. Formally, given a set of canvas patches and the same set of reference image's patches, our overall loss function is:

$$\mathcal{L}_{rec} = \frac{1}{N}\sum_{n=0}^{N} \alpha\mathcal{L}_{\text{pixel}_\text{n}} + \beta\mathcal{L}_{\text{perc}_\text{n}} \tag{5.1}$$

where N is the total number of patches. For our pixel loss, we use the L_1 function, $\mathcal{L}_{\text{pixel}} = |C - I|$, where C is a canvas and I a reference image. For perceptual loss, we use the feature vectors from a pretrained VGG16 [38]. Specifically, let $V_{ij} = \{V_{ij}^1, ..., V_{ij}^k\}$ and $W_{ij} = \{W_{ij}^1, ..., W_{ij}^k\}$ be a set of k feature vectors extracted from image I and canvas C_T, respectively. We use cosine similarity as follows:

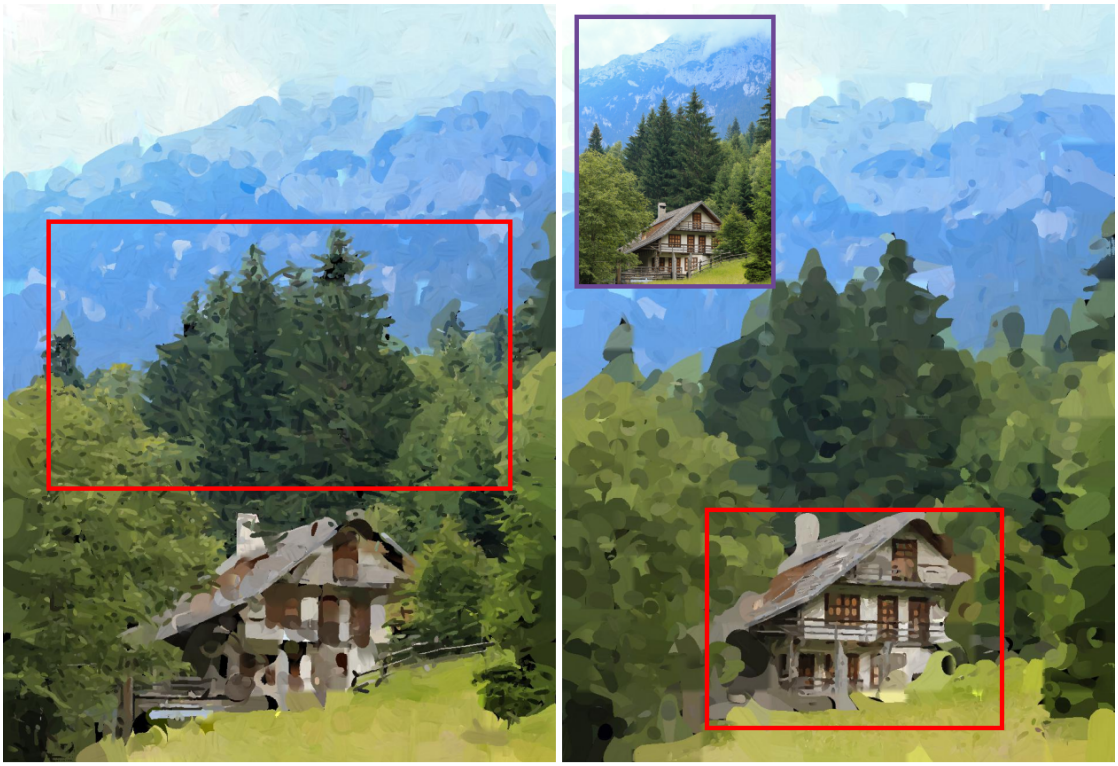
$$\mathcal{L}_{\text{perc}} = \cos\theta = \sum_{k}^{K}\sum_{i,j} \frac{V_i W_j}{\|V_i\|\,\|W_j\|} \tag{5.2}$$

where i, j index the spatial dimensions of the feature maps V and W, and K the extracted layers from VGG16 trained on ImageNet [33]. We use layers 1, 3, 6, 8, 11, 13, 15, 22 and 29, and optimize via gradient descent and use Adam optimizer [18] to find the set of strokes S_{k_p}.

5.4 Experiments

We now demonstrate stylistic effects achievable with our method, and compare with state-of-the-art methods.

The semantic control provided by our method allows a user to apply different styles to different regions, akin to classical artistic techniques where backgrounds might be abstracted for emphasis on detailed foregrounds and vice-versa, reminiscent of techniques observed in classical artworks [14], and specific subjects or objects may be highlighted. The versatility of this approach, resulting in varied visual interpretations based on the areas of emphasis, is illustrated in Figure 3.1 (a) and (c), Figure 5.5, Figure 5.6, Figure 5.7, and Figure 5.9.



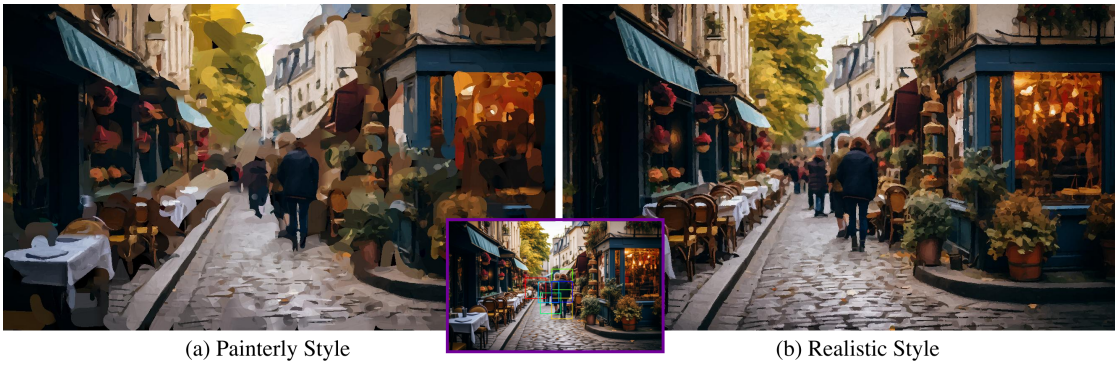
(a) Emphasis on foliage (b) Emphasis on building

Figure 5.5: Our approach offers enhanced precision over the canvas's semantic regions, directing specific emphases in the artwork: (a) accentuates the foliage, capturing its essence, while (b) concentrates on highlighting the architectural details of the building.

5.4.1 Style Variations

Painting techniques and strategies lead to different styles, and our algorithm can output a set of stylistic outputs, ranging from realism to abstraction and painterly styles.

Realism. We begin by demonstrating a style that precisely reproduces the target image. Previous methods cannot achieve accurate image reproduction through optimization; this limits their ability to define new styles, since many styles cannot be accurately optimized, e.g., fine-scale details cannot be preserved. While most painterly styles involve greater degrees of abstraction, this style illustrates the superior performance of our sequential patch-based optimization, which can achieve accurate image reproduction.



(a) Painterly Style (b) Realistic Style

Figure 5.6: Different painterly styles given the same input. (a) Painterly style achieved by the use of our segmentation pipeline and selective dynamic attention maps. (b) Realistic style achieved by using uniform dynamic maps.

For this style, we use a single layer ($K = 1$), without semantic segmentation, and run four coarse-to-fine painting passes ($P = 4$). The number of strokes s per patch increases with each layer, following the formula $s_p = (p + 5)^2$. Stroke thicknesses are halved across consecutive passes, given by the formula: $a_p = 2^{1-p}a_1$, where a_p and a_1 are the p-th and thicknesses. This approach ensures that the painting captures finer details, rendering a realistic appearance. Paintings employing this methodology are showcased in Figure 3.1 (c), Figure 5.4, Figure 5.6 (b), Figure 5.7 (c) top, and Figure 5.8 (fourth row). We use perceptual losses along with pixel losses to capture higher frequency areas. That is, we optimize the stroke parameters following Equation (5.1), where $\alpha = 1$ and $\beta = 0.01$. This style achieves superior reconstruction details than previous methods as seen in Section 5.4.2, and in Figure 3.1 (c), and Figure 5.8 (fourth row). Analyzing these paintings, high-frequency areas are captured with higher definition than previous methods, that is, strokes follow edges and shapes more accurately, resulting in higher-fidelity paintings.

Painterly Style. The goal of this style is to generate a more loose representation of the input image, yet capturing enough details so that the original scene is represented. Leveraging semantic segmentation, our approach mirrors the human painting process,

(a) Control - Vegetation (b) Control - Motorcycle

Figure 5.7: Demonstration of control over the painting. (a) The painting focal point falls in the buildings while leaving the vegetation abstracted. (b) The motorcycle is emphasized by adding more detail than the background, which is coarsely painted.

working piece-by-piece, focusing on one region or aspect of a painting at a time, and generating different painterly styles. Instead of a uniformly detailed canvas, semantic control offers refined governance over stroke distribution, tailored to distinct semantic zones. Our dynamic attention maps, inspired by visual working memory, are configured to a selective mode, honing in on areas necessitating enhanced detail. The stroke parameters are optimized solely based on pixel loss, that is, we set $\beta = 0$ in Equation (5.1). For each painting pass p, we found enough to have a stroke budget of 16 strokes per patch, which yields a distinctive painterly style. This approach prioritizes artistic expression over mere replication, yielding a rendition that captures the essence rather than

the exact likeness of the input image, as demonstrated in Figure 5.2 (b), Figure 5.5, Figure 5.6 (a), Figure 5.7 (a, bottom), (b, top), and Figure 5.9 (b,c). In these figures, we can see that this style has the loose quality of typical impressionistic paintings, yet forms are defined and the subject matter is clear.

Semantic Abstraction. Unlike the previous painterly style, which still prioritizes a good balance between accuracy and expression, this style leans with a heavier emphasis on expression. By adjusting the optimization procedure over painting passes P and semantic layers K, we can generate abstract representations, offering a novel painting synthesis. To do this, the user associates different optimization parameters such as P, stroke budget, and number of selective dynamic attention maps V to different semantic layers k. Since this style does not focus on high-fidelity, stroke parameters are optimized based on L_1 loss, setting $\beta = 0$ in Equation (5.1). This generates organic and abstract representations of the object of interest, focusing on expression over representation, and can be applied universally or selectively to accentuate specific semantic zones. See how in Figure 5.2 (a), and Figure 5.9 (a), the abstract paintings loosely define the subject matter, prioritizing heavy abstract expressionism, in contrast to the painterly style, which captures the subject matter, but still leaving room to the viewer's interpretation. A practical illustration: buildings can be rendered in finer detail to spotlight their significance, whereas adjacent elements like streets, pedestrians, or foliage are abstracted, adding an artistic bias to the scene (refer to Figures 5.5 and 5.7). As a result, the painting looks more intentional, organic and fluid, similar to how human artists paint. Figure 5.2 (a) and Figure 5.9 (a) are excellent examples of the human-like qualities of this style, with brushwork being loose and gestural, creating a rich tapestry of dynamic shapes. Other examples using this style are shown in Figure 5.8 (bottom row), where semantic layers are entirely abstracted from their context. For further details on the parametrization of this style, please refer to supplementary material.

	Control	Style Var.	Large-Scale	High-low Freq.
PT	✗	✗	✓	✗
SNP	✗	✓	✓	✗
L2P	✗	✗	✗	✓
Ours	✓	✓	✓	✓

Table 5.1: Comparison with previous methods on attributes like painting control, style variations, large-scale parametric painting, and high-low frequency capture.

5.4.2 Comparison with State-of-the-Art

We evaluate our work against three notable parametric methods: an optimization-based method "Stylized Neural Painting" (SNP) [40], a learning-based method with Transformer "Paint Transformer" (PT) [23], and a RL method "Learning to Paint" (L2P) [15]. Our method stands out by supporting any resolution or aspect ratio, preserving artwork quality, unlike L2P and PT which resize outputs, often reducing quality. Moreover, SNP is limited to square formats, restricting its application. An overview of algorithmic attributes is shown in Table 5.1[1] . While all methods can output paintings at high resolutions (1024×1024 or higher), L2P's results are blurry and seams are visible. For a fair comparison, all paintings have been generated using the same number of strokes.

Qualitative Comparison A visual comparison of our method against earlier techniques is presented in Figure 5.2, alongside additional comparisons in Figure 5.8 and Figure 5.9. For more extensive comparisons, please see the supplemental materials. In Figure 5.2, we observe that SNP and PT tend to produce impressionistic-like paintings. However, their brushwork appears patchy and lacks flexibility, failing to accurately cap-

[1]High and low-frequency areas of an image refer to the level of detail present in different parts of the image. High-frequency areas are regions with a lot of detail and rapid changes in brightness or color, such as sharp edges, fine textures, or intricate patterns. Low-frequency areas, on the other hand, are characterized by smooth and gradual variations, such as flat or slowly varying regions like clear skies or untextured walls. High-frequency content typically corresponds to the detailed information in the image, while low-frequency content is associated with the broader, more homogeneous areas.

L2P [15]

SNP [40]

PT [23]

Figure 5.8: 1/2. Comparison of four parametric painting techniques—L2P, SNP, PT, and our approach (shown in next figure)—across three different scenarios. Our method's outcomes show improved detail rendition and variability. Note L2P's realistic yet blurry style, SNP's broad painterly effect which struggles to capture fine details (tree branches or windows framing), and PT's inconsistent detail and brushstroke scale. Zoom-in recommended for full appreciation of textural nuances.

Ours - Realistic

Ours - Abstract

(a) Ours - Expressionist (b) Ours - Painterly (c) Ours - Painterly

Figure 5.9: 2/2. Our realistic approach finely details both high and low-frequency elements, accurately rendering tree branches and window frames. Painterly styles with a different degree of abstraction. This style generates organic human-like paintings, biasing the painting towards different structures. Zoom-in recommended for full appreciation of textural nuances.

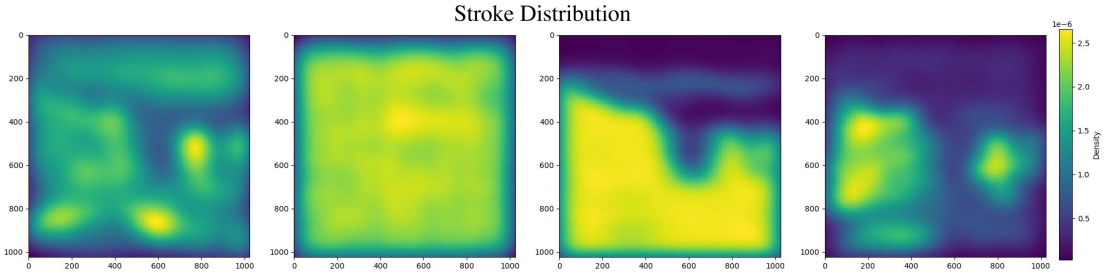
Method	Realism	Visual Appeal
Ours vs. PT [23]	81.82%	79.55%
Ours vs. SNP [40]	100%	70.45%
Ours vs. L2P [15]	95.45%	56.82%

Table 5.2: Qualitative evaluation based on user preference. This table shows a 2-way comparison between our method and previous methods.

ture edges. This issue is more pronounced in Figure 5.8, where PT exhibits a stark contrast in brush sizes across different semantic regions, along with noise in areas of high detail. SNP, with its rigid and uniform patches, struggles to render finer details effectively. Among previous methods, L2P shows a better ability to capture both micro and macro structures realistically, but it tends to produce blurry results at higher resolutions and lacks versatility in painting styles. This limitation of L2P is evident in Figure 5.10, where it fails to differentiate between semantic areas or objects, uniformly treating the entire canvas without specific emphasis.

User Study. We assessed our painting method's effectiveness through a user study involving 30 participants, contrasting our outputs with three existing techniques. Participants judged the paintings based on realism and visual appeal across six painting sets, for a total of 12 image rating tasks, using a two-way comparison (2 images side by side), and collecting 360 responses. As shown in Table 5.2, our method is preferred over existing methods in all tasks. Overall, our method substantially outperforms previous methods in realism, selected by 92.42% of the users. Our method is also preferred in the visual appeal task, voted by 68.94%. When evaluating our method on visual appeal, our strongest competitor is L2P, selected by 43.18% of the participants. For more details, refer to Appendix B.

Quantitative Comparison To further substantiate our claims, we compare the performance of our model on reconstruction of the original input against previous methods, and use pixel loss L_1 and perceptual loss L_{perc} as evaluation metrics, reported in Ta-



(a) SNP (b) L2P (c) PT (d) Ours

Figure 5.10: Stroke distribution comparison of Figure 5.8 middle column. SNP (a) and PT (c) exhibit a disparse clustering, while L2P (b) paints evenly in a scene-agnostic way. Our method (d) combines the selective focus of SNP with the clarity of PT, indicating a refined approach to applying strokes.

ble 5.3. We select 10 images from each domain, resize them to 512x512 and set the number of strokes to 4000. Our method outperforms previous methods in high-fidelity realistic paintings across all evaluated domains, which is consistent with the qualitative results obtained in our user study.

To visually quantify how each method places strokes on the canvas, Figure 5.10 compares the stroke distributions for the middle column paintings shown in Figure 5.8 and Figure 5.9. Our method (d) shows a more intentional stroke placement and clarity in stroke application. In contrast, SNP (a) shows a scattered, uniform treatment of scene elements, while our approach strategically focuses on two key areas, corresponding to the vegetation, enhancing the composition's structure. L2P (b) applies strokes uniformly in a scene-agnostic way, lacking the focal emphasis of our method. PT (c) creates abrupt density transitions due to its multi-scale approach, unsuitable for grad-

	Ours		SNP		PT		L2P	
	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$
Objects	**0.038**	**0.680**	0.052	0.554	0.085	0.480	0.053	0.586
Buildings	**0.026**	**0.720**	0.049	0.515	0.071	0.490	0.045	0.632
Streets	**0.031**	**0.709**	0.041	0.582	0.072	0.486	0.041	0.620
Animals	**0.016**	**0.745**	0.029	0.573	0.054	0.535	0.026	0.647

Table 5.3: Quantitative results on realistic style across different image domains. All methods use the same number of strokes (4000) on a 512x512 canvas.

ual tonal shifts, and fails to disentangle high-density areas, not differentiating between scene elements like trees and roads. Our method, however, clearly delineates regions with varying stroke densities, offering a nuanced and intentional subject representation. See supplemental for additional details.

Efficiency We compare the efficiency of the different previous methods in a single RTX A6000 Nvidia GPU for a single image (see Table 5.4). While neural methods require training, they are faster at inference time than optimization methods. However, when comparing with previous optimization methods [40], our method is significantly faster (4x) for the same number of strokes (1600) and resolution size (1024x1024). This is specially important because optimization methods require an optimization process per image. For a fair comparison, we use the uniform setting for our dynamic maps, and set the style to realism.

5.5 Conclusion

This chapter presented a semantic-based painting method that is more controllable, produces more visually appealing paintings, and achieves better realistic paintings than previous methods, as validated in the user studies and quantitative evaluations. Contrary to previous methods, our work offers a controllable framework that, besides realism, also aims for painterly and abstracted styles without the need for style-transfer like techniques. Furthermore, it is able to paint at any aspect ratio without further resizing steps.

5.5.1 Limitations and Discussion

As we have seen, our method is able to provide a higher control over the painting process. However, it is still limited in the styles that it can output. Perceptual and pixel losses, combined with the optimization strategies presented, seem not to be enough to achieve a higher stylistic variation. A technical constraint that limits our approach

Method	Optimization-based		Neural-based	
	Ours	Optim [40]	Transf [23]	RL [15]
Res. 1024	**99.60s**	447.10s	**0.514s**	0.537s
Req. Training	No	No	Yes	Yes
Use Dataset	No	No	No	Yes

Table 5.4: Inference efficiency for different methods, measured in seconds for a single image at 1024x1024 and 1600 strokes.

from exploring other losses is that the reconstruction of the canvas from the patches is a non-differentiable operation, and by extension, we cannot apply a loss that evaluates the entire painting. As a consequence, and looking ahead, our research will evolve to include the development of a differentiable pasting module. This addition is designed to work in conjunction with our existing patch loss strategy, opening new avenues for stylization research at the general-canvas level.

In the next chapter, we shift focus to the topic of identity preservation in human face avatars. We plan to explore a new representation approach, transitioning from the traditional uniform stroke-based painting to a more diverse primitive-based composition. The core principle remains consistent: creating artistic renditions from input images through a parametric model, yet with a new perspective on the representation medium.

References

[1] Ardavan Bidgoli et al. “Artistic Style in Robotic Painting; a Machine Learning Approach to Learning Brushstroke from Human Artists”. In: *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2020, pp. 412–418. DOI: 10.1109/RO-MAN47096.2020.9223533.

[2] Nicolas Carion et al. “End-to-end object detection with transformers”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer. 2020, pp. 213–229.

[3] Rebecca Chamberlain and Johan Wagemans. “The genesis of errors in drawing”. In: *Neuroscience & Biobehavioral Reviews* 65 (2016), pp. 195–207.

[4] Dale J Cohen. “Look little, look often: The influence of gaze frequency on drawing accuracy”. In: *Perception & psychophysics* 67 (2005), pp. 997–1009.

[5] Yaroslav Ganin et al. “Synthesizing programs for images using reinforced adversarial learning”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1666–1675.

[6] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. “Image Style Transfer Using Convolutional Neural Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[7] Kuba Glazek. “Visual and motor processing in visual artists: Implications for cognitive and neural mechanisms.” In: *Psychology of Aesthetics, Creativity, and the Arts* 6.2 (2012), p. 155.

[8] Jörg Marvin Gülzow, Liat Grayver, and Oliver Deussen. “Self-Improving Robotic Brushstroke Replication”. In: *Arts* 7.4 (2018). ISSN: 2076-0752. DOI: 10.3390/arts7040084. URL: https://www.mdpi.com/2076-0752/7/4/84.

[9] David Ha and Douglas Eck. “A neural representation of sketch drawings”. In: *arXiv preprint arXiv:1704.03477* (2017).

[10] Paul Haeberli. “Paint By Numbers: Abstract Image Representations”. In: *Proc. SIGGRAPH*. 1990.

[11] Aaron Hertzmann. “A Survey of Stroke-Based Rendering”. In: *IEEE Computer Graphics & Applications* 23.4 (2003).

[12] Aaron Hertzmann. “Paint By Relaxation”. In: *Proc. CGI*. 2001.

[13] Aaron Hertzmann. “Painterly Rendering with Curved Brush Strokes of Multiple Sizes”. In: *Proc. SIGGRAPH*. 1998.

[14] S. Hodge. *The Painting Techniques of 5 Famous Artists, from Botticelli to Vermeer*. 2019. URL: https://www.artsy.net/article/artsy-editorial-painting-techniques-5-famous-artists-botticelli-vermeer.

[15] Zhewei Huang, Wen Heng, and Shuchang Zhou. “Learning to Paint With Model-based Deep Reinforcement Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[16] Biao Jia et al. “LPaintB: Learning to Paint from Self-Supervision”. In: *Proc. Pacific Graphics*. 2019.

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. “Perceptual losses for real-time style transfer and super-resolution”. In: *European conference on computer vision*. Springer. 2016, pp. 694–711.

[18] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[19] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[20] Tzu-Mao Li et al. “Differentiable vector graphics rasterization for editing and learning”. In: *ACM Trans. Graph.* 39.6 (2020).

[21] Thomas Lindemeier et al. “Hardware-Based Non-Photorealistic Rendering Using a Painting Robot”. In: *Computer Graphics Forum* 34.2 (2015), pp. 311–323. DOI: https://doi.org/10.1111/cgf.12562. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12562. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12562.

[22] Peter Litwinowicz. “Processing Images and Video for an Impressionist Effect”. In: *Proc. SIGGRAPH*. 1997.

[23] Songhua Liu et al. “Paint Transformer: Feed Forward Neural Painting With Stroke Prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 6598–6607.

[24] Barbara J Meier. “Painterly rendering for animation”. In: *Proc. SIGGRAPH*. 1996, pp. 477–484.

[25] John F. J. Mellor et al. *Unsupervised Doodling and Painting with Improved SPIRAL*. arXiv:1910.01007. 2019.

[26] Haoran Mo et al. “General Virtual Sketching Framework for Vector Line Art”. In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2021)* 40.4 (2021), 51:1–51:14.

[27] Peter O’Donovan and Aaron Hertzmann. “AniPaint: Interactive Painterly Animation from Video”. In: *IEEE Transactions on Visualization and Computer Graphics* 18.3 (2012), pp. 475–487. DOI: 10.1109/TVCG.2011.51.

[28] Florian Perdreau and Patrick Cavanagh. “Drawing experts have better visual memory while drawing”. In: *Journal of Vision* 15.5 (2015), pp. 5–5.

[29] Paul Rosin and John Collomosse. *Image and Video-Based Artistic Stylisation*. Springer, 2013.

[30] Peter Schaldenbrand, James McCann, and Jean Oh. “FRIDA: A Collaborative Robot Painter with a Differentiable, Real2Sim2Real Planning Environment”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 11712–11718. DOI: 10.1109/ICRA48891.2023.10160702.

[31] Peter Schaldenbrand and Jean Oh. “Content Masked Loss: Human-Like Brush Stroke Planning in a Reinforcement Learning Painting Agent”. In: *Proc. AAAI*. 2021.

[32] Johannes Schmid et al. “OverCoat: An Implicit Canvas for 3D Painting”. In: *Proc. SIGGRAPH*. 2011. DOI: 10.1145/1964921.1964923. URL: https://doi.org/10.1145/1964921.1964923.

[33] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[34] Jaskirat Singh and Liang Zheng. “Combining Semantic Guidance and Deep Reinforcement Learning For Generating Human Level Paintings”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 16387–16396.

[35] John Tchalenko. “Segmentation and accuracy in copying and drawing: Experts and beginners”. In: *Vision research* 49.8 (2009), pp. 791–800.

[36] John Tchalenko and R Chris Miall. “Eye–hand strategies in copying complex lines”. In: *Cortex* 45.3 (2009), pp. 368–376.

[37] Ning Xie, Hirotaka Hachiya, and Masashi Sugiyama. “Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting”. In: *IEICE TRANSACTIONS on Information and Systems* 96.5 (2013), pp. 1134–1144.

[38] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *Proc. CVPR*. 2018.

[39] Ningyuan Zheng, Yifan Jiang, and Dingjiang Huang. “StrokeNet: A Neural Painting Environment”. In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=HJxwDiActX.

[40] Zhengxia Zou et al. “Stylized Neural Painting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 15689–15698.

Chapter 6

Cross-modal Latent Space Alignment for Image to Avatar Translation

MANUEL LADRON DE GUEVARA, JOSE ECHEVARRIA, YIJUN LI, YANNICK HOLD-GEOFFROY, CAMERON SMITH, DAICHI ITO. PUBLISHED AT ICCV 2023.

As we transition from the exploration of stroke-based painting techniques discussed in the preceding chapters, we enter the realm of digital identity representation through avatars. An avatar, in this context, is a digital persona, a vectorized embodiment that represents an individual within virtual spaces. This chapter delves into why and how we extend the investigation of painterly styles and stroke-based methods to the creation of avatars.

The impetus for this line of inquiry is twofold: firstly, to address the growing significance of avatars as a form of personal expression in an increasingly digital world; and secondly, to push the boundaries of how artistry and technology can coalesce to represent human likenesses in other forms other than realistic portraiture using the same brushstroke representation. Here, we will see how we can create avatar representations from real portrait images through the use of different stroke primitives. At a high-level, the framework is the same as the ones presented in previous chapters: a mapping from an image to a vector of parameters that define an artistic representation of the input

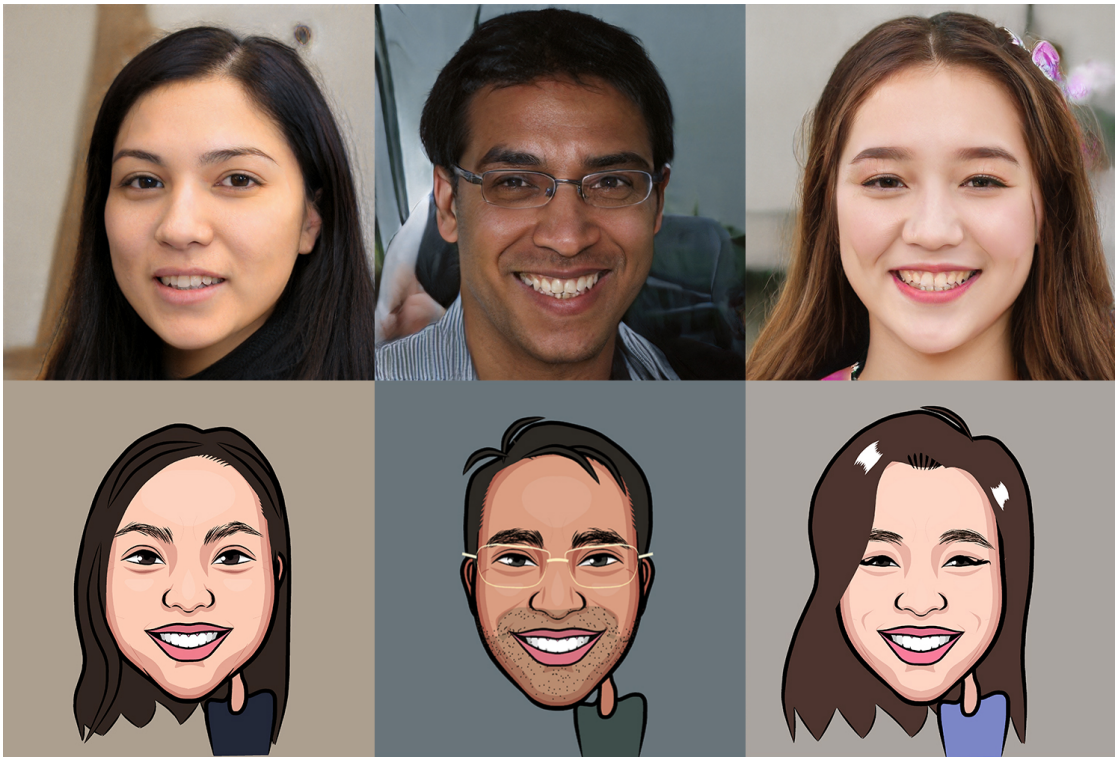


Figure 6.1: Generated Avatars (bottom row) from their corresponding inputs (top row.)

image.

We present a novel method for automatic vectorized avatar generation from a single portrait image. Most existing approaches that create avatars rely on image-to-image translation methods, which present some limitations when applied to 3D rendering, animation, or video. Instead, we leverage modality-specific trained on large-scale unpaired portraits and parametric avatars, and then learn a mapping between both modalities via an alignment module trained on a significantly smaller amount of data. The resulting cross-modal latent space preserves facial identity, producing more visually appealing and higher fidelity avatars than previous methods, as supported by our quantitative and qualitative evaluations. Moreover, our method's virtue of being resolution-independent makes it highly versatile and applicable in a wide range of settings.

Publication website: Cross-modal Latent Space Alignment for Image to Avatar

6.1 Introduction

An *avatar* can be defined as a digital representation or virtual character by which people represent themselves and other beings in a virtual platform or community [4]. Such a representation has been ubiquitous in our daily life, from personal use on social media to marketing strategies by companies [23]. Given an image and a style, the task is to generate a new representation that successfully conveys the content of the image in the given style, while preserving the person's identity as much as possible.

Generally, avatars are typically created in two ways. First, it can be created by skilled artists. To streamline the avatar creation, many artists use predefined facial presets from a library and combine them part by part to obtain the final avatar (e.g., the "manual" results from [33]). This scheme is commonly used for messaging and video games avatars. However, it can be a tedious process that frequently alters the subject's identity due to the limited predefined assets. In contrast, automatic image generation has been employed as a time-efficient and large-scale alternative for this task. Those methods typically focus on caricature generation [3, 29, 7, 17, 11, 35]. While such methods could translate real photography to an avatar or caricature, they all operate in image space, which poses some limitations when used in virtual environments or for animations.

Parametric vector graphics propose a solution that overcomes the drawbacks of static images to some extent. In particular, Wolf et al. [33] propose a tied output synthesis network to infer parameters from generated image avatars, but their method works with very simple datasets like face emoji [31], which avoids the matter of stylization. Hu et al. [9] and Shi et al. [27, 28] focus on 3D photorealistic-looking avatars. However, these works do not focus on capturing a specific style, nor producing 2D parametric avatars.

In this chapter, we propose an alternative to image-based avatars that enables animations and video compositing without losing quality or resolution. To do so, we utilize a

parametric representation to depict avatars; that is, we employ a set of parameters that define each facial attribute. To train such an approach in a supervised way requires a dataset of paired portraits and their respective avatars, which is prohibitively expensive to obtain. To address this, we present a weakly-supervised novel cross-modal alignment framework that translates rich representations from the portrait domain to the parametric avatar domain. To alleviate the lack of paired data, we first learn modality-specific latent spaces from large-scale unpaired data. Once such latent spaces are learned, we match their latent representations by a cross-modal module trained on a small amount of paired data, which preserves the facial identity from the portrait and the style of the parametric avatar. As a result, our framework is able to translate identity features in image space to parametric space, and at the same time, apply style features from the original vector-based parametric space.

The goal of this work is to propose a proficient approach for converting individual images into parametric avatar representations that accurately preserve the individual's identity. In summary, our contributions are two-fold:

- A flexible parametric representation that encodes facial attributes, enabling the representation of a wide range of human appearances.
- A novel cross-modal framework that, when provided an input image, generates higher quality avatars with better preservation of the individual's identity than previous methods.

6.2 Related Work

Handcrafted priors. Early caricature and avatar methods were procedural algorithms where hand-designed rules produced an exaggerated representation of a portrait, either by displacing the facial vertices by a constant factor from the average face [2], exaggerating the size of the facial components [24], using facial feature nodes [20], or using an interactive feature grid for caricaturization [8]. The use of key anthropomorphic mea-

surements was proposed by Varshney et al. [32], and later Le et al. [16] use exaggerated anthropometric rations between facial components to manipulate a given image. These methods present some limitations in style variation and identity preservation, mainly due to their predesigned rules. This led to an early work using K-Nearest Neighbors [19] where a shape exaggeration module is trained on a small paired dataset of facial keypoints on both original and caricature images.

Learned priors. Many recent methods leverage deep learning algorithms that aim at an end-to-end image to avatar generation [3, 17, 29, 7, 11, 35]. Most approaches use a common image-to-image framework, in which both input and output exist in image space. Within this framework, CycleGAN [36], StarGAN and StarGAN v2 [6, 5], UNIT [21], MUNIT [10] or U-GAT-IT [15] are common strategies to achieve a desired stylized output. However, these approaches do not explicitly disentangle facial identity from caricature style, nor explicitly model face exaggerations. To address this, some approaches use warping modules. Dense fields are used in [34] to control exaggerated facial shapes without changing the input style into a caricature. Gong et al. [7] use flow estimation and differentiable warping to generate cartoons. Other work uses point-based deviations to achieve similar results [29, 17].

Generative imaging. More recent techniques leverage newer models such as StyleGAN [12, 13] to explicitly preserve identity [35] in an unsupervised manner. Toonify [26] creates caricatures by swapping layers of two StyleGANs trained on real images and caricatures. Jang et al. [11] extend this idea by introducing shape exaggeration blocks.

Weak supervision. A recent line of work based on few-shot and transfer learning is used for caricature generation by Li et al. [18], where they achieve successful face translations by adapting a pretrained GAN via the use of weight regularization-based losses. Ojha et al. [25] improve on this work by using a pretrained StyleGAN and introducing cross-domain correspondence losses.

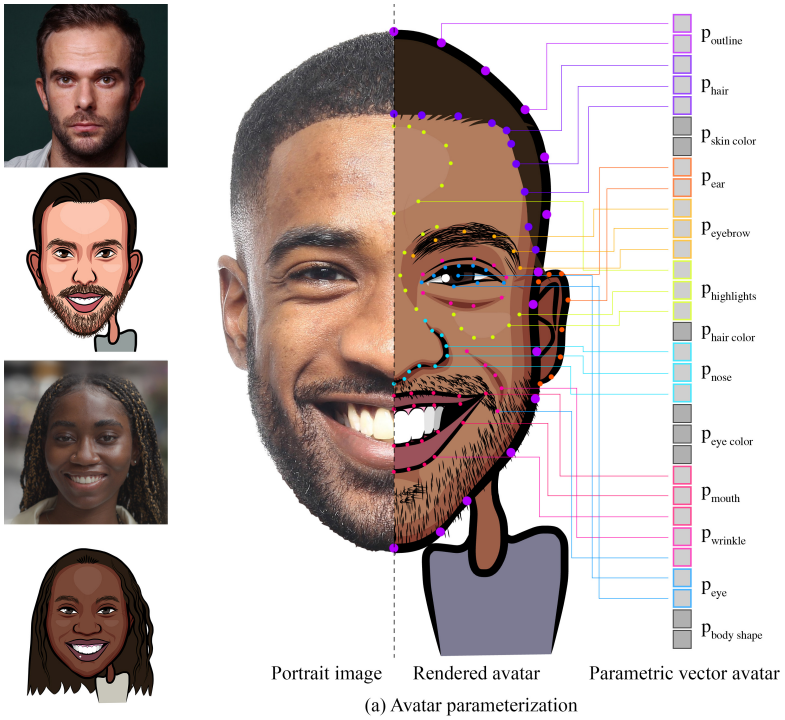
Parametric methods. There exist very few methods in the literature that try to resolve a cross-modal image-to-parametric avatar translation. Tied Output Synthesis (TOS), introduced by Wolf et al. [33], learns a mapping from a portrait to both a vector in parameter space and the corresponding image generated by this vector, handling the issue of domain adaptation. While this approach can generate emojis, the generated avatars lack identity preservation due to their renderer. Moreover, their method was tested on a simple face emoji dataset [31], and might need to be adapted to generate more complex avatars. Shi et al. [27] introduce a face-to-parameter style-transfer based method to create 3D realistic avatars. This method, however, is based on a linear PCA that requires a slow optimization step. In a follow-up [28], the same authors sidestep the optimization using face segmentation and recognition. However, they produce 3D realistic-looking models which convey their own artistic style, probably unamenable to new styles.

Our parametric method borrows ideas from the few-shot and domain adaptation fields. Unlike previous works, we focus on a 2D avatar representation that is easily transferable to 3D, and perform latent space translation, avoiding bottlenecks such as differentiable renderers. Moreover, to overcome the lack of paired data, we adopt ideas from domain adaptation methods, that is, we first leverage larger amounts of unpaired data to learn "expert" models, to then perform a cross-modal translation under the guidance of such models with weakly-supervised data.

6.3 Dataset

In the following, we describe our parametric system to encode facial features and our dataset.

Parametric System. Our avatar representation is characterized by its flexibility and high level of expressiveness, enabling it to accurately depict the vast diversity of facial features across different regions of the world. Our parametric system is specifically de-



(a) Avatar parameterization

Figure 6.2: Avatar Parameterization (right) and training samples (left). We encode the avatar as a vector of 629 parameters, which include point coordinates (represented as dots in the avatar above) which define spline curves, color, and line-weights of facial features. This vector is rendered by a non-differentiable vector graphics engine.

signed to capture the defining characteristics of any face using a minimal number of parameters. Each avatar is parameterized by a 629-dimensional vector $\bar{y} \in \mathbb{R}^{629}$, which defines x, y coordinates, color, and line weights for all facial features, including lips, teeth, chin, eyes, and skin color, among others. A detailed breakdown of these parameters can be found in the supplementary material in Appendix C. The resulting avatar vector is grouped by parameters of facial components such that $\bar{y} := \{\bar{y}_{eyes}, \bar{y}_{nose}, \bar{y}_{mouth}, ...\}$, as shown in Fig. 6.2.

The parameters described in our work can be rendered at any resolution without any loss of quality or clarity. This is achieved by utilizing a vector graphics engine that creates images using mathematical equations to generate lines, curves, and shapes, instead of pixels. Our work also demonstrates the potential for using these parameters in ani-

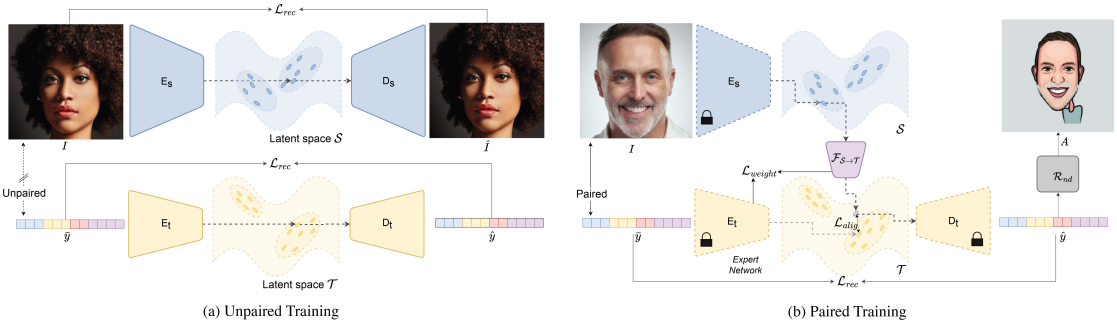
mated graphics and 3D environments. However, we acknowledge that these applications are beyond the scope of our current work and are discussed further in the supplemental material.

Dataset. We asked 393 volunteers of various ages and ethnicities to provide us with selfies. All participants are recruited from within Adobe, adhering to internal policies. Consequently, the age range of volunteers is approximately between 20 and 65 years, though specific ages were not recorded. Approximately 61% of the volunteers are male. The volunteers reflect the diversity typically found in a tech company in California. The recruitment was conducted via email, and beyond obtaining their avatar, no other compensation was offered. Subsequently, a technical research artist at the Adobe Research lab, and a member of the research team, was tasked with crafting avatars of the participants through our parametric vector engine that accurately portrayed their identities. This results in 393 pairs of real captured portraits and their corresponding avatar parameters as determined by the artist.

Due to the limited number of paired samples, developing a reliable avatar translation method directly on this dataset is presently infeasible. To address this challenge, we augment the number of paired samples using both synthetic faces generated by StyleGAN and automatic augmentations, such as warping, on real portraits. Specifics regarding these augmentations are provided in the supplementary material. With these augmentations, our dataset comprises a total of 9970 paired samples.

6.4 Method

Our goal is to learn a cross-modality mapping between a real image and a vector of parameters that define an avatar version of the input image in a given style. Typically, paired data in the form of image portrait and avatar parameters $(X, \bar{Y})$ is limited due to their prohibitive time and cost to obtain, and directly training this mapping leads to overfitting issues. Moreover, to successfully learn such a mapping, we need to overcome



(a) Unpaired Training (b) Paired Training

Figure 6.3: Model overview. (a) Unpaired training stage: we train modality-specific networks to learn rich latent spaces. (b) Paired training stage: we introduce our cross-modal alignment module to translate between different latent spaces to transfer key facial features to a parametric domain.

some challenges surrounding multimodal approaches, such as domain shift or alignment between domains [1]. To address both problems, we first leverage larger amounts of unpaired data to learn rich modality-specific latent spaces, and then use a small amount of paired data to learn a mapping $F : \mathcal{S} \longrightarrow \mathcal{T}$ over lower-dimensional latent spaces, which helps to preserve the identity of an image x and the style of a vector of parameters $\bar{y}$.

By design, we disentangle hair and accessory parameters from the rest of the face parameters. Inspired by [9], we allow the model to focus solely on facial structures. A separate pipeline that leverages pretrained face attributes automatically retrieves the best hair and accessory parameters.

6.4.1 Unpaired Modality-Specific Representations

Let $\{x_i\}_{i=1}^{N}$ be the set of training images where $x \in \mathcal{X}$ are images in the domain of portrait photos and $\{\bar{y}_i\}_{j=1}^{M}$ the set of training avatar vectors where $\bar{y} \in \mathcal{Y}$ is a vector of parameters that define an avatar. We want to find rich lower-dimensional representations of each modality. For this, we learn encoder-decoder networks per modality such that a portrait encoder $E_s : \mathcal{X} \in \mathbb{R}^{3\times h\times w} \longrightarrow \mathcal{S} \in \mathbb{R}^u$, and a vector encoder $E_t : \mathcal{Y} \in \mathbb{R}^v \longrightarrow \mathcal{T} \in \mathbb{R}^u$ take their respective input modalities, and output latent vectors with

the same dimensionality $\mathbb{R}^u$. Both, the portrait decoder $D_s : \mathcal{S} \longrightarrow \mathcal{X}$ and the vector decoder $D_t : \mathcal{T} \longrightarrow \mathcal{Y}$ are tasked to reconstruct the original, and both encoder-decoder functions satisfy:

$$\arg\min_{E,D} \mathbb{E}[\mathcal{L}_{rec}(x, D \circ E(x)] \tag{6.1}$$

For image modality, the reconstruction loss becomes

$$\mathcal{L}_{rec} = |x - D \circ E(x)| + \lambda_p \mathcal{L}_{perc}, \tag{6.2}$$

$$\mathcal{L}_{perc} = \frac{1}{Kij} \sum_{k}^{K} \sum_{ij} ||V_{ij}^k - W_{ij}^k||_2^2 \tag{6.3}$$

where i, j index the spatial dimensions of the feature maps V and W, and K the extracted layers from VGG19 trained on ImageNet [30]. For parameter vector modality, the reconstruction loss is simply $\mathcal{L}_{rec} = (x - D \circ E(x))^2$. Both encoder-decoder networks are trained independently in parallel. After training, we discard the image decoder D_s. The image encoder is modeled by a convolutional neural network, and the image decoder uses transpose convolutions to generate an image from a latent vector. The parameter vector encoder-decoder network is defined by multilayer perceptrons with non-linearity activation functions. Refer to the supplementary material for details about the networks.

6.4.2 Cross-Modal Alignment

After learning rich latent spaces from our large-scale data, the goal is to find a mapping between them. The cross-modal network $F : S \longrightarrow T$ takes in a feature latent vector $z_s = E_s(x)$ and outputs a new latent vector $z_m = F(E_s(x))$ in the translated space, where $S, T \in \mathbb{R}^u$. We model F as a multilayer perceptron with one hidden layer and non-linearity activation functions. We train F on a weakly paired dataset in the form $\{x_i, \bar{y}_i\}_{i=1}^N$, where $x_i, \bar{y}_i$ is the i-th tuple of paired image and parameter vector,

respectively. We fix the weights of E_s and D_t, perform a forward pass through these networks, and use a reconstruction loss $\mathcal{L}_{rec} = (\hat{y} - \bar{y})^2$, where $\hat{y} = D_t(F(E_s(x)))$. While training, F is learning an intermediate latent space M that wants to be as close to T as possible. This translation, however, requires further regularization terms to enforce a closer alignment between such latent spaces.

Cross-Modal Alignment Loss. To enforce an explicit alignment between the new translated space M and the vector parameter latent space T, we can use the encoder E_t to extract the learned latent representation $z_t = E_t(\bar{y})$ of the input parameter vector $\bar{y}$, and align the translated vector z_m to z_t. Our new cross-modal alignment loss becomes:

$$\mathcal{L}_{cm} = \lambda_1 \left(z_m - z_t\right)^2 + \lambda_2 \left(1 - \frac{z_m \cdot z_t}{||z_m|| ||z_t||}\right) \tag{6.4}$$

Weight Alignment Loss. The mapping network F tries to project vectors into the same latent space as the target encoder E_t, a network that has been previously pre-trained on larger amounts of data, enough to learn rich representations of the inputs. Drawing inspiration from few-shot settings [18], the goal of this loss is to impose a strong regularization over the weights of the mapping network by taking guidance from the weights of E_t. To address the difference in network shape between E_t and F, this loss is enforced only in the last layer F, which shares the same dimensionality. In simple terms, E_t is treated as the expert network, and since the last layer of F is responsible for projecting a hidden vector into the same latent space as E_t, we want F to imitate E_t as closely as possible. Our weight alignment loss is as follows:

$$\mathcal{L}_w = ||\theta_f - \theta_t||_2^2 \tag{6.5}$$

where θ_f and θ_t represent the weights of the last layer of the mapping network F and the parameter vector encoder E_t. The final loss becomes:

$$\mathcal{L} = \lambda_r \mathcal{L}_{rec} + \lambda_c \mathcal{L}_{cm} + \lambda_w \mathcal{L}_w \tag{6.6}$$

6.4.3 Hair Pipeline

Hair and accessories are variable components of faces and avatars that can be easily changed or adjusted. As such, we model this aspect independently from the rest of the face [9]. Some of these parameters are categorical, making them unsuitable for regression losses. The diversity of accessories and hairstyles also yields a distribution with properties that differs significantly from facial features, posing problems unifying the training with facial parameters. To predict hair and accessory attributes, we rely on a pretrained attribute regressor [14]. This function $\mathcal{H}(I) \rightarrow \hat{a}$ uses a ResNet50 backbone to estimate attributes such as age and hairstyle from input images. Refer to the supplementary material for more details.

We precompute $\hat{a}$ vectors of a database of images and avatar parameters pairs. Let x_j be a query of an unseen portrait image at inference time. To find hair and accessory parameters, we compute $\hat{a}_j = \mathcal{H}(x_j)$ and use nearest neighbors to find the closest attribute vector from our database. We then merge the hair and accessory parameters of the best match with the facial parameters $\hat{y}_j$ generated by our cross-modal pipeline, as shown in Figure 6.4. Our hair database has a total of 470 different hairstyles to query from.

6.4.4 Rendering Engine

The rendering engine is responsible for translating avatar parameters into a corresponding pixel representation. In our implementation, we use Python, and the PIL library as our primary drawing library. Our vector representation is limited to a set of primitives, including cubic Bezier curves for outlines and contours, polygons for color filling, highlights and shadows, and composites to create eyebrows and hair, as shown in Figure 6.5. The renderer works by adding layers, ordered according to facial features. For instance, the facial structure is drawn before the eyes, just as an artist would draw a face. Each face attribute is encoded as a vector of point coordinates in the x and y directions, lineweights (widths and lengths), and RGB parameters. For instance, the nose of an

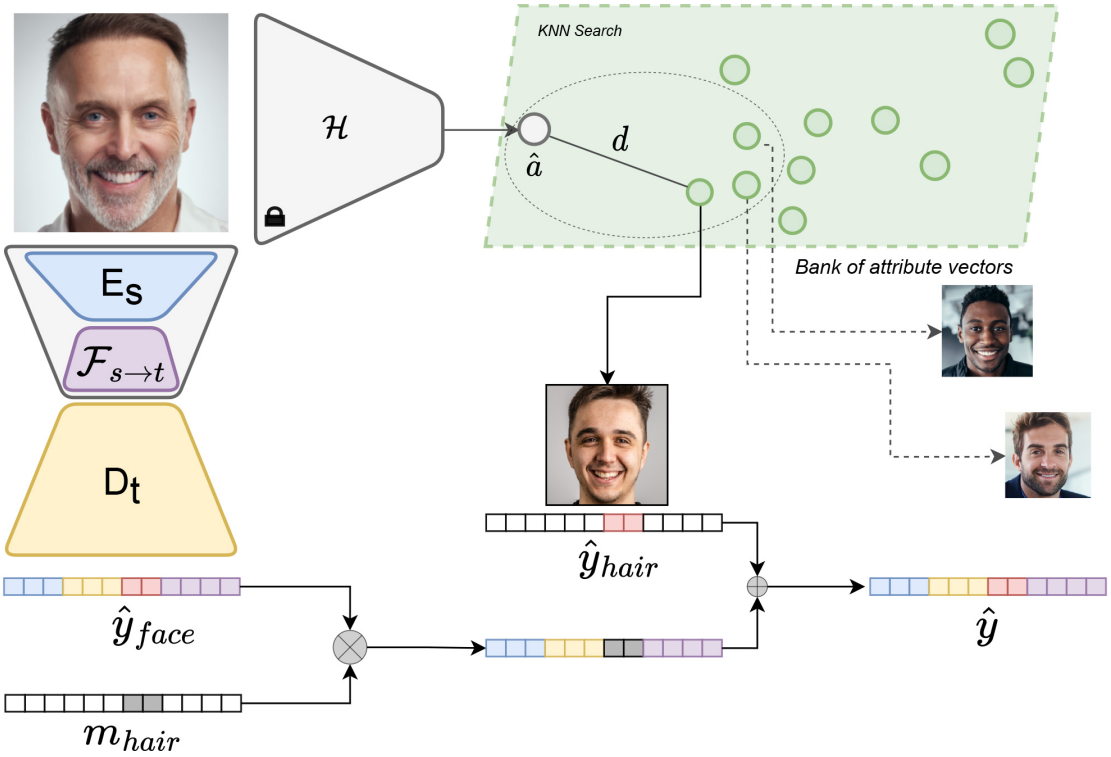


Figure 6.4: Inference including hair retrieval pipeline. We use a pretrained attribute predictor $\mathcal{H}$ to pre-compute attribute vectors a of our faces from our hair bank. We use the K-nearest neighbors algorithm to retrieve the closest vector to a query vector $\hat{a}$ from a bank of a vectors. d indicates smallest distance in the KNN search.

avatar is represented by a set of coordinate points, start and end radius, and color values: $\bar{y}_{nose} := \{(x, y)^N, r_s, r_e, R, G, B\}$. Our engine then fits curves through the input x, y coordinates and applies lineweights and colors. In order to achieve a natural stroke effect, we taper the end points of the lines.

6.4.5 Implementation Details

Unpaired Training. We first train our image and parameter vector autoencoders independently and in parallel until convergence. For our image autoencoder, we use FFHQ dataset [12] and our face dataset for a total of 90000 aligned 256x256 resolution faces. Our parameter vector autoencoder is trained on 100000 vector avatars. Note that the

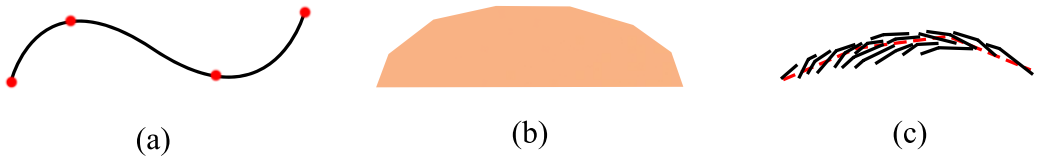


Figure 6.5: Our method supports vector graphics formed by (a) cubic Bezier curves, for contours and outlines, (b) polygons, for highlights and shadows, and (c) custom composites, for eyebrows and hair, with colored strokes and fills.

two autoencoders are not related; as explained in Section 6.4.1, we trained them on an unpaired dataset. We apply several random augmentations on images, including Gaussian blur, rotation, horizontal flips, and grayscale images. We use Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.99$, an initial learning rate of 0.0002, batch size of 24 and we set λ_2 (Equation (6.2)) to 0.1. We set the latent spaces to be $\mathcal{S}, \mathcal{T} \in \mathbb{R}^{512}$. All experiments using our method were trained on a single Tesla V100 GPU.

Paired Training. In this stage, we freeze the weights of E_s, E_t, and D_t, and only update the weights of the mapping network F. We use the same optimizer configuration as before, but increase the batch size to 48. We use a learning rate scheduler on loss plateau with patience 1 and factor 0.5. We set lambdas in Equation (6.6) as follows: $\lambda_r = 1, \lambda_c = 1, \lambda_w = 10$. In total, we have 9970 pairs of portraits and avatar parameters, and 3995 pairs of portraits and avatar images.

6.5 Experiments

In this section, we provide a quantitative and qualitative evaluation of our method, and compare it to several state-of-the-art methods. We train previous methods on our paired dataset from scratch using their official implementations, and use the default parameters. All methods are trained using a 256x256 image resolution. We then provide ablation studies to evaluate the performance of different components of our method, such as

the impact of losses, amount of data, and face encoders. To ensure a fair comparison, we train the paired stage of our model with the same amount of data that we train previous methods (3995 image-avatar pairs) when evaluating against previous methods (Figure 6.6, Table 6.1, and Table 6.2).

6.5.1 Comparison to State-of-the-art Methods

We evaluate our method against generic image-to-image translation methods, a GAN adaptation method, and a recent caricature method. Following [11], we choose U-GAT-IT [15] and StarGAN-v2 [5] among image-to-image translation methods as they provide visually good results and generate fewer artifacts than older methods. We also compare our method to MSGAN-pix2pix [22], as it is designed specifically for paired datasets. We choose a GAN adaptation method [25] because our paired dataset is based on a few thousand examples, and these methods generally prevent StyleGAN-2 [13] from overfitting. Lastly, we compare our method against a recent network specifically designed to work on caricature generation, StyleCariGAN [11]. This method requires 2 pre-trained StyleGANs, one trained on real faces and the other one on caricatures or avatars. Before training this model, we train a StyleGAN-2 model on our avatar dataset. At inference, [25, 11] require GAN-inversion to project an input image to a latent code before translating to the avatar domain.

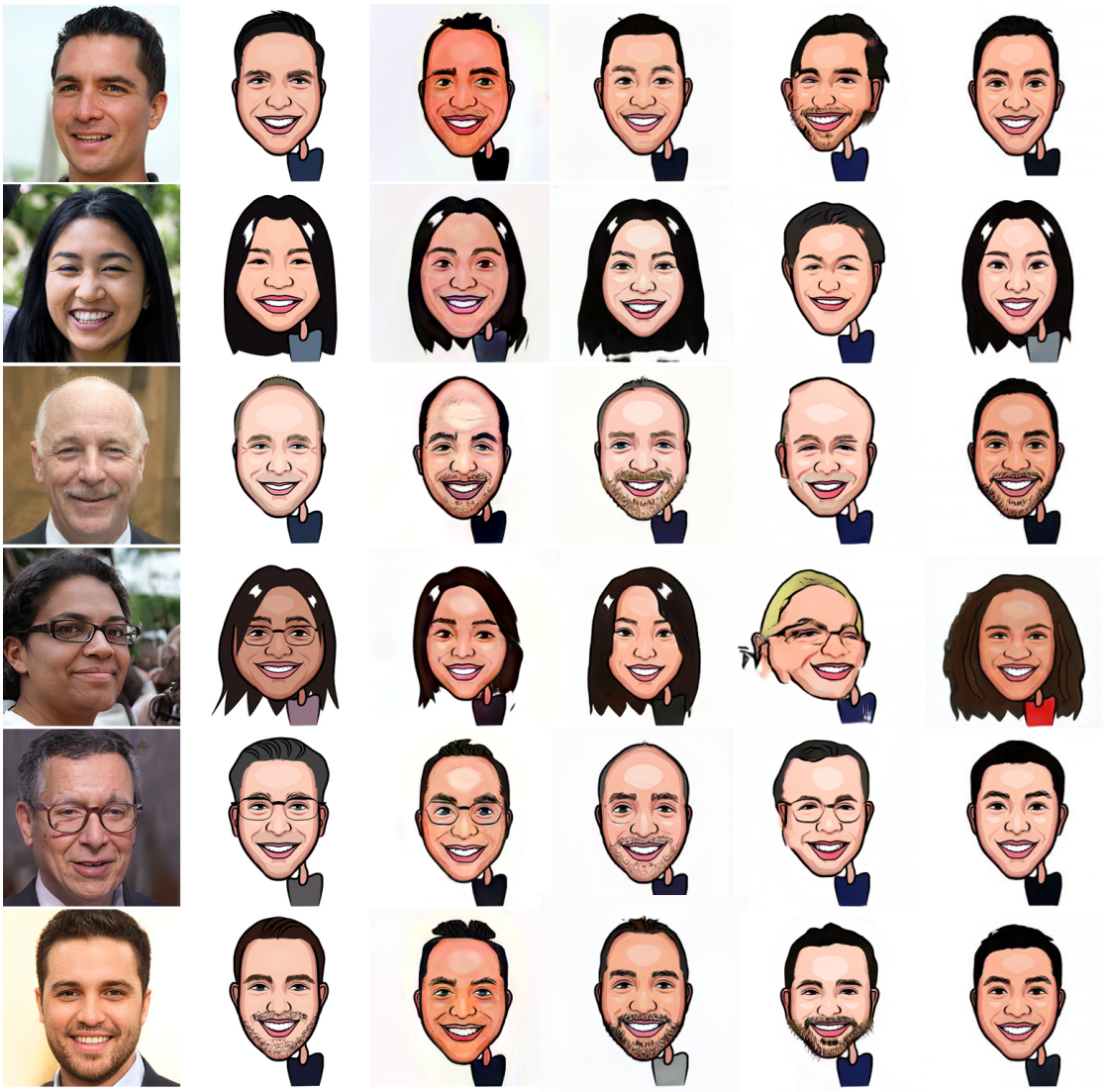


Figure 6.6: Comparison with state-of-the-art methods for avatar generation and image-to-image translation. All the methods are trained on the same dataset with the same amount of data. Our method is able to preserve identity with higher accuracy and more consistently than previous methods.

In Figure 6.6, we present a qualitative evaluation of the performance of several state-of-the-art methods. U-GAT-IT exhibits a lack of consistency in preserving correct facial structure, often resulting in squashed or deformed avatars. This observation is further

supported by the results presented in Table 6.2. Nevertheless, U-GAT-IT generally performs well in maintaining identity, even when changing the facial structure. StarGAN-v2, on the other hand, produces stylistically pleasing outputs with consistently well-structured faces, but struggles with preserving identity. The GAN Adaptation method maintains identity to a large extent, but the quality is compromised with some coloring artifacts. StyleCariGAN generates good quality avatars, but it lacks consistency in preserving identity. Results for MSGAN-pix2pix are available in the supplemental.

In general, our proposed method shows superior performance in terms of both quality and identity preservation. As demonstrated in Figure 6.6, our method consistently maintains identity across various demographic factors, such as age, gender and ethnicity. Our parametric approach circumvents common issues that are often encountered in generative pixel-based methods, such as lack of structural consistency and generation of artifacts.

To further substantiate our claims, we report evaluation metrics in Table 6.1. Specifically, we compare the performance of our model against previous methods using pixel loss ($L1$) and perceptual loss (LPIPS) metrics to quantify the difference between the generated output and a ground truth avatar in a held-out dataset consisting of 30 avatars. Our method outperforms previous methods in terms of both pixel and perceptual losses. It is worth noting that StyleGAN2-based methods generally achieve better scores than other approaches.

6.5.2 User Study

We conduct a perceptual study to evaluate identity preservation and avatar quality. We compare our method against the same previous methods: GAN Adaptation [25], U-GAT-IT [15], StarGAN-v2 [5], MSGAN-pix2pix [22], and StyleCariGAN [11]. We structure the study with 2-way comparison tasks, where each comparison shows our method against another method chosen randomly. The evaluation of identity preservation is conducted by presenting users with a portrait photograph and asking them to select the avatar that best preserves the identity of the person in the photograph, that

Method	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{lpips} \downarrow$
U-GAT-IT [15]	0.3164	0.2803
GAN-Adapt. [25]	0.2558	0.3622
StarGAN-v2 [5]	0.2111	0.2135
MSGAN-pix2pix [22]	0.2059	0.3747
StyleCariGAN [11]	0.2441	0.3530
Ours	**0.1864**	**0.1895**

Table 6.1: Quantitative evaluation on a held-out dataset. All models are trained on the same dataset containing 3955 avatars.

Method	Identity	Quality
Ours vs. U-GAT-IT [15]	78.76%	99.1%
Ours vs. GAN-Adapt. [25]	74.63%	85.8%
Ours vs. StarGAN-v2 [5]	73.46%	74.30%
Ours vs. MSGAN-pix2pix [22]	81.70%	88.50%
Ours vs. StyleCariGAN [11]	72.83%	87.6%

Table 6.2: Quantitative evaluation on user preference. This table shows a 2-way comparison between our method and previous methods.

is, the avatar that looks more like the photograph. The avatars are presented side by side, and their location is randomized in each question. In the second part of the study, we evaluated avatar quality. To ensure an unbiased assessment, participants were instructed to disregard factors related to image quality, such as resolution, and to focus solely on the quality of the avatar itself, i.e., how well the avatar maintains the overall facial structure, facial attribute proportions, relative distances between facial features, or consistency of colors as they pertained to facial features.

We asked 113 users, and each one is asked 15 identity questions and 5 quality questions. In total, we collect 2260 responses. As shown in Table 6.2, our method is preferred over previous methods in all tasks. Overall, 81.17% of users select our method across all tasks, with 75.28% showing preference for our method on identity preservation and 87.06% for avatar quality.

When evaluating our method on avatar quality, the strongest competitor is the StyleGAN2-

based method, StarGAN-v2, selected by 25.7% of users. StyleCariGAN, MSGAN, and GAN-Adaptation obtain similar preference percentages, 12.4%, 11.5% and 14.2% respectively. U-GAT-IT is chosen by less than 1% of users. We hypothesize that this is due to the face deformations shown in Figure 6.6. More details about the user study are presented in Appendix C.

6.5.3 Ablation Study

Effect of Proposed Loss Functions. In this ablation study, we examine the performance of our mapping function F under various objectives, and keep the other networks fixed. Our reconstruction loss, and cross-modal alignment loss compare the predicted latent code produced by the image encoder and mapping network, represented as $z_m = F(E_s(x))$, with the output of a pretrained parameter encoder, denoted as E_t and referred to as the expert network, given by $z_t = E_t(\bar{y})$.

The weight regularization term is designed to impose constraints on the weights of the last layer of the mapping function F. This is based on the hypothesis that the weights in the final layer of F should closely resemble those in the final layer of E_t, as both networks are designed to be a 2-layer MLP and aim to project vectors into the same latent space.

Figure 6.7 shows the effect of the proposed losses during cross-modal training. When optimizing with a parameter reconstruction loss $\mathcal{L}_{rec}$ without using any alignment loss (column b), the model preserves overall identity but fails to capture smaller

Method	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{lpips} \downarrow$
Ours w/o alig	0.1856	0.1876
Ours w/o mse alig	0.1841	0.1864
Ours w/o csim alig	0.1834	0.1869
Ours w/o weight alig	0.1833	0.1852
Ours	**0.1832**	**0.1835**

Table 6.3: Quantitative evaluation on the effect of each loss function.

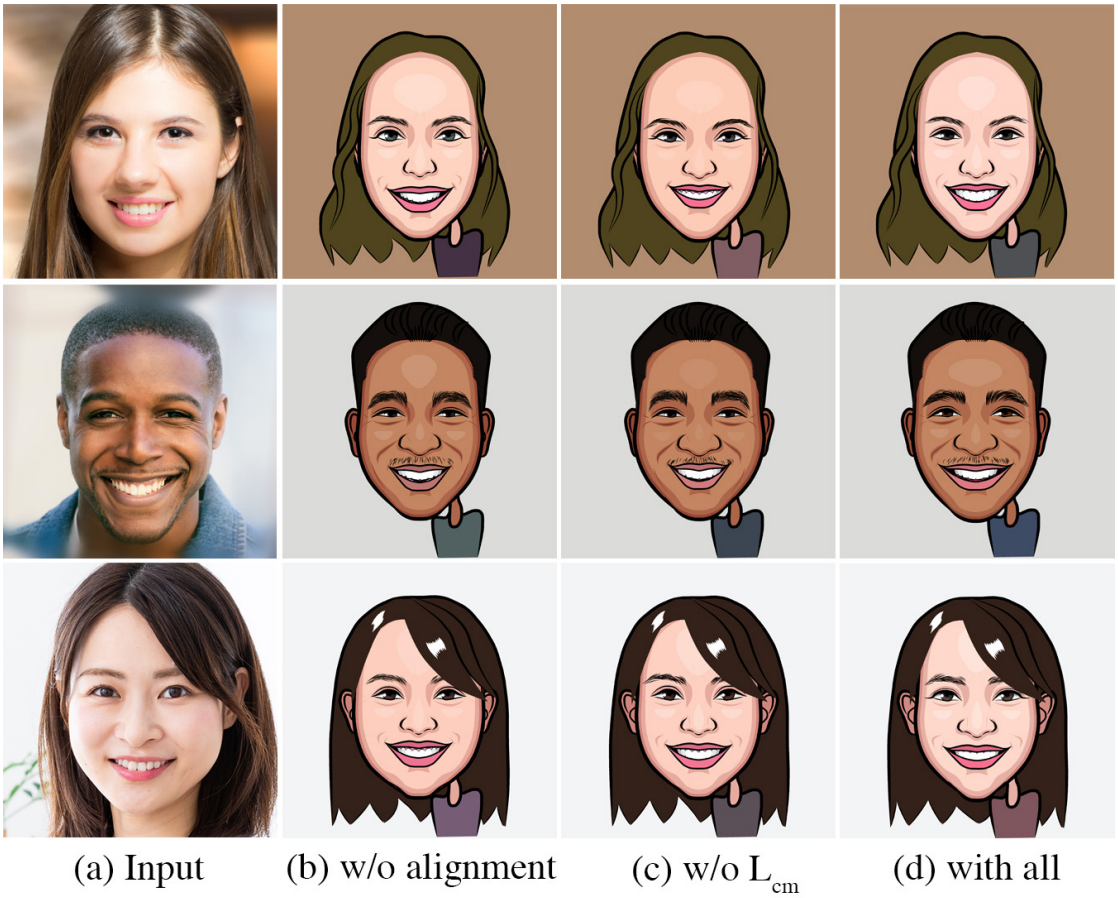


Figure 6.7: Model variations without proposed losses. Readers are encouraged to zoom in to observe how details affect identity preservation. (b) Uses only L_{rec} between generated and ground truth parameters. (c) Only uses weight alignment loss. (d) Uses all proposed losses.

details that enhance identity preservation such as the relative position of facial features or a correct size (see the mouth size in the middle row). Similar results are observed (column c) when training without Equation (6.4). However, using all proposed losses results in better positioning and sizing of key facial features like eyes and mouth (see distance between eyes and nose in the example shown in the upper row). A detailed analysis of the effect of the different loss functions is provided in the supplementary material.

(a) Input (b) Ours$_{\text{L}}$ (c) Ours$_{\text{M}}$ (d) Ours$_{\text{S}}$ (e) Ours$_{\text{XS}}$

Figure 6.8: Effect of paired dataset size. (b) Our model trained on 9970 paired images. (c) Our model trained on 3995 pairs. (d) Our model trained on 1000 image-avatar parameter pairs. (e) Our model trained on 500 samples.

Effect of Paired Data Size. The impact of the size of paired data on our model is examined in this section. As obtaining paired data is an expensive process, we investigate the effect of reducing the amount of paired data in the training of our mapping network $\mathcal{F}$. We compare the performance of our model trained on different sizes of paired data: Ours$_L$, trained on 9970 pairs Figure 6.8 (b), Ours$_M$, trained on 3395 pairs Figure 6.8 (c), Ours$_S$, trained on 1000 pairs Figure 6.8 (d), and Ours$_{XS}$, trained on 500 pairs Figure 6.8 (e). While models L and M are capable of preserving identity, using fewer than 3000 image-avatar pairs leads to degradation in both identity preservation and avatar quality, specifically in maintaining facial structure and relative positioning of facial features.

6.6 Conclusion

We have proposed a method that generates parametric avatars directly from a single portrait image. Our framework, which utilizes a cross-modal mapping between two previously trained latent spaces, has allowed us to capture the vast diversity of human appearance and preserve identity in the generated avatars. Through our alignment

losses, we have successfully guided the translation process and demonstrated that our approach outperforms previous methods both qualitatively and quantitatively. One key advantage of our approach is the use of parametric avatars, which obviate the resolution constraints often encountered in image-based approaches. This also facilitates an easily transferable solution that is readily compatible with animation or 3D software applications. Our contributions offer a step forward in the development of avatar representation, which have the potential to impact a wide range of fields, from gaming to virtual reality.

Limitations and Future Work. Our method generates a diverse set of avatars, but limitations still exist. The use of a hair and accessory databank limits matching for complex hairstyles and accessories, such as hats. Additionally, our parametric avatars are restricted to front-view only. Future work includes integrating hair and accessories within the generative pipeline and allowing for pose variance.

This chapter concludes the series of technical experiments central to this dissertation. Throughout, we have explored various methodologies aimed at resolving key challenges in generating artistic outputs, focusing particularly on control, stylization, and identity preservation. The next chapter will serve as the culmination of this dissertation. In it, I will present a comprehensive summary of our research findings and engage in a thoughtful discussion about the interplay between artificial intelligence and creativity, reflecting on the broader implications and future directions of this dynamic field.

References

[1] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. "Multimodal machine learning: A survey and taxonomy". In: *IEEE transactions on pattern analysis and machine intelligence* 41.2 (2018), pp. 423–443.

[2] Susan E Brennan. "Caricature generator: The dynamic exaggeration of faces by computer". In: *Leonardo* 18.3 (1985), pp. 170–178.

[3] Kaidi Cao, Jing Liao, and Lu Yuan. "Carigans: Unpaired photo-to-caricature translation". In: *arXiv preprint arXiv:1811.00222* (2018).

[4] Edward Castronova. "Theory of the Avatar". In: *Available at SSRN 385103* (2003).

[5] Yunjey Choi et al. "Stargan v2: Diverse image synthesis for multiple domains". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8188–8197.

[6] Yunjey Choi et al. "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8789–8797.

[7] Julia Gong, Yannick Hold-Geoffroy, and Jingwan Lu. "Autotoon: Automatic geometric warping for face cartoon generation". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 360–369.

[8] Bruce Gooch, Erik Reinhard, and Amy Gooch. "Human facial illustrations: Creation and psychophysical evaluation". In: *ACM Transactions on Graphics (TOG)* 23.1 (2004), pp. 27–44.

[9] Liwen Hu et al. "Avatar digitization from a single image for real-time rendering". In: *ACM Transactions on Graphics (ToG)* 36.6 (2017), pp. 1–14.

[10] Xun Huang et al. "Multimodal unsupervised image-to-image translation". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 172–189.

[11] Wonjong Jang et al. “StyleCariGAN: caricature generation via StyleGAN feature map modulation”. In: *ACM Transactions on Graphics (TOG)* 40.4 (2021), pp. 1–16.

[12] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[13] Tero Karras et al. “Analyzing and improving the image quality of stylegan”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8110–8119.

[14] Siavash Khodadadeh et al. “Latent to Latent: A Learned Mapper for Identity Preserving Editing of Multiple Face Attributes in StyleGAN-generated Images”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022, pp. 3184–3192.

[15] Junho Kim et al. “U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation”. In: *arXiv preprint arXiv:1907.10830* (2019).

[16] Nguyen Kim Hai Le, Yong Peng Why, and Golam Ashraf. “Shape stylized face caricatures”. In: *International Conference on Multimedia Modeling*. Springer. 2011, pp. 536–547.

[17] Wenbin Li et al. “Carigan: Caricature generation through weakly paired adversarial learning”. In: *Neural Networks* 132 (2020), pp. 66–74.

[18] Yijun Li et al. “Few-shot image generation with elastic weight consolidation”. In: *arXiv preprint arXiv:2012.02780* (2020).

[19] Lin Liang et al. “Example-based caricature generation with exaggeration”. In: *10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings.* IEEE. 2002, pp. 386–393.

[20] Pei-Ying Chiang Wen-Hung Liao and Tsai-Yen Li. “Automatic caricature generation by analyzing facial features”. In: *Proceeding of 2004 Asia Conference on Computer Vision (ACCV2004), Korea*. Vol. 2. 2004.

[21] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. “Unsupervised image-to-image translation networks”. In: *Advances in neural information processing systems* 30 (2017).

[22] Qi Mao et al. “Mode seeking generative adversarial networks for diverse image synthesis”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 1429–1437.

[23] Fred Miao et al. “An emerging theory of avatar marketing”. In: *Journal of Marketing* 86.1 (2022), pp. 67–90.

[24] Zhenyao Mo, John P Lewis, and Ulrich Neumann. “Improved automatic caricature by feature normalization and exaggeration”. In: *ACM SIGGRAPH 2004 Sketches*. 2004, p. 57.

[25] Utkarsh Ojha et al. “Few-Shot Image Generation via Cross-Domain Correspondence”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 10743–10752.

[26] Justin NM Pinkney and Doron Adler. “Resolution dependent gan interpolation for controllable image synthesis between domains”. In: *arXiv preprint arXiv:2010.05334* (2020).

[27] Tianyang Shi et al. “Face-to-parameter translation for game character auto-creation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 161–170.

[28] Tianyang Shi et al. “Fast and robust face-to-parameter translation for game character auto-creation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 02. 2020, pp. 1733–1740.

[29] Yichun Shi, Debayan Deb, and Anil K Jain. “Warpgan: Automatic caricature generation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10762–10771.

[30] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[31] Yaniv Taigman, Adam Polyak, and Lior Wolf. “Unsupervised cross-domain image generation”. In: *arXiv preprint arXiv:1611.02200* (2016).

[32] Lav Varshney. “Face metamorphosis and face caricature: A user’s guide”. In: *School of Electrical and Computer Engineering, Cornell University* (2004).

[33] Lior Wolf, Yaniv Taigman, and Adam Polyak. “Unsupervised creation of parameterized avatars”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1530–1538.

[34] Ruizheng Wu et al. “Attribute-driven spontaneous motion in unpaired image translation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 5923–5932.

[35] Lan Yan et al. “IPGAN: Identity-Preservation Generative Adversarial Network for unsupervised photo-to-caricature translation”. In: *Knowledge-Based Systems* 241 (2022), p. 108223.

[36] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. 2017.

Interlude: Reflections on AI and the Nature of Creativity

In the preceding chapters of this dissertation, we have delved deeply into the technical intricacies of machine learning algorithms and their application in the realm of digital art. We have conducted thorough experimentation and analysis to investigate how artificial intelligence can be utilized to innovate, augment, and comprehend artistic processes, thereby advancing the forefront of the field. As we transition from the detailed examination of these computational methods to the concluding reflections of our work, it is pertinent to pause and broaden our perspective.

This interlude serves as a reflective bridge between the empirical research detailed in Chapter 6 and the concluding synthesis in the next chapter. Here, we will take a step back from the algorithmic focus and turn our attention to the wider implications of this intersection between technology and creativity. This is not a summary of findings, but a contemplative space where we consider the ongoing dialogue between artificial intelligence and the human conception of creativity.

Within these pages, we will revisit the notion of creativity as it is understood both in the early and current psychological research, and in artistic domains, highlighting

how this work contributes to this ongoing discourse. The discussion is centered on the creative models and fundamental theories that underpin our understanding of this quintessentially human trait, now being replicated and augmented by artificial systems. I also contemplate the contextual relevance of this work, considering how the technological advances presented in this thesis sit within the broader realm of human creative endeavor.

Discussion

The integration of machine learning (ML) in the realm of art not only opens new avenues for exploration but also invites a complex set of debates encompassing ethics, the extent of technical knowledge, the role of data, and the human element in artistic creation. One of the fundamental questions at the heart of this discussion is the extent to which AI algorithms can emulate the rich, multifaceted nature of human artistry. Human art encodes reasoning, perception, aesthetics, creativity, psychology, memories, and social context. Capturing this complexity in an ML model poses a significant challenge due to the depth and subtlety of these elements. It also opens the question of whether algorithms need to encode all such human traits in order to generate art. The relation between current technologies and art is rather complex, and needs, at least, a comprehensive evaluation from different and divergent points of view.

Between the multifaceted debates surrounding machine learning's role in art and the emerging realities faced by artists in the digital age, lies a transitionary space—a period of adjustment and reflection. As we acknowledge the depth and complexity that human artistry brings to the table, it is important to consider the immediate effects of AI on the artistic community. While generative systems bring about concerns of economic impact and authenticity [30], they also pave the way for new forms of expression and methodologies in creative work [13]. A prevailing view among scholars and industry professionals is that generative AI promises to free creative minds from the more laborious and repetitive tasks of the artistic process, such as the meticulous arrangement

of basic elements to compose a painting or an illustration. This technological shift is expected to free up artists to focus more on the essence of creation: the conceptual and ideational aspects that fuel innovation and artistic expression.

It is in this interlude of technological evolution where we must weigh the implications of AI-generated content. On one hand, there is the undeniable efficiency and scalability of AI tools that challenge traditional artistic methods. On the other, the essence of art as a human endeavor, imbued with emotion, experience, and cultural significance, continues to be irreplaceable. This juxtaposition invites a deeper contemplation of the future trajectory of AI in art, questioning not just its current impact but also its potential to redefine the creative landscape.

This thesis posits that AI should be regarded as an active collaborator in the artistic process, transcending the traditional role of a simple instrument. The term 'tool' suggests a passive contribution, failing to capture the depth of engagement and the augmentation of capabilities that AI now offers. Contemporary generative models, such as large language models (LLMs) and advanced image generators, enhance our capacity for brainstorming and ideation, indicating a shift towards a partnership model where AI acts as a co-creative force alongside the artist.

In the context of generative models, particularly LLMs and vision-enhanced LLMs [64], there is a notion that they can create a comprehensive model of the world. Ilya Sutskever, a prominent figure in AI, posits that building a statistical model for predicting the next word in a sequence enables these models to encapsulate a broad spectrum of world knowledge [60, 59]. This is largely because the data feeding these models is a reflection of human thought, perception, and emotion. A similar phenomenon is observed in image generative models, which, after processing millions of images and their descriptions, demonstrate a seemingly intuitive understanding of user prompts.

However, does this capability equate to ML models possessing human-like reasoning? Currently, the answer leans towards the negative. ML models operate fundamentally as statistical models, and though they occasionally produce outputs resembling human-like creations, this is the result of numerous underlying computations rather than

genuine cognitive processes. It is crucial to recognize that these models, while not imbued with consciousness, possess capabilities that extend beyond the function of simple tools. In the domain of art, for instance, they demonstrate potential as co-creators or collaborators, contributing to the creative process. Furthermore, the question of whether these ML models can be considered "creative" in a true sense remains open and subject to interpretation. To address this question, a clear and comprehensive definition of creativity is required—a definition that can effectively encompass the nuances of both human and machine-generated art.

Contributions to the Ongoing Dialogue on AI and Creativity

In the realm of AI's role in creative processes, there is a dynamic and ongoing debate regarding the extent to which technology can emulate, supplement, or even enhance human creativity. This thesis contributes to this discourse by exploring the intersection of AI algorithms and artistic expression, particularly in the context of painting and stylization.

The development and implications of the algorithm presented in Chapter 5 resonate with a key aspect of this debate. While it is inspired by human painting strategies, integrating more nuanced human-like attributes such as perception, context, and inspiration into AI models proves to be a complex challenge, one that extends beyond the current scope of our work. This aligns with views in the field that AI, in its current state, may not fully replicate the depth and spontaneity of human creativity, as highlighted by authors like Aaron Hertzmann [23]. However, and as stated earlier, models do encode some human knowledge because the training data that feeds the models is a compilation of many attributes of human culture and knowledge, from scientific papers, to the arts [60, 59, 13]. Consequently, the extent to which the model's intrinsic attributes can naturally contribute to the generation of creative outputs, as opposed to relying on externally imposed human-driven directives, remains a compelling question for future exploration.

Moreover, other dimensions of creativity, such as autonomy and discovery, are pivotal in the discourse on AI and creativity. These aspects, which are central to the human

creative process, are not yet fully encapsulated in our algorithms, and remain open for future work. In the broader context of AI research, there is a growing interest in exploring these dimensions through methodologies like reinforcement learning (RL). However, our approach does not incorporate RL, highlighting a potential avenue for future research.

By situating this work within these broader conversations, it becomes evident that while significant strides have been made, there remains a rich landscape of unexplored territory in the quest to blend AI with human-like creativity. Our research contributes to this ongoing dialogue by providing insights and raising further questions about the capabilities and limitations of current AI technologies in artistic creation.

This thesis navigates the intersection of technological prowess and artistic inquiry. On one hand, it is anchored by three main technical chapters, each corresponding to papers either published or pending publication in esteemed computer graphics and vision conferences. This emphasis reflects the ML research community's drive for technical advancement, often prioritizing innovation in the field to the detriment of artistic examination. On the other hand, this work endeavors to infuse an artistic consciousness into its framework. The chapter dedicated to economical painting (Chapter 3), for example, approaches problem-solving from a primarily artistic viewpoint, diverging from the literature's traditional focus on iterative technical enhancement. It underscores my conviction that forging deeper, more integrative collaborations between artists and computational researchers is essential for the genesis of algorithms that truly serve the creative community.

Scientific Research on Art, Creativity, and Creative Models

There is a number of authors that write about creativity or try to model "creative algorithms" to understand the potential and limitations of current systems with regards to this field.

For instance, Aaron Hertzmann argues that computers cannot be considered artists [23]. However, there are some nuances to consider. If creativity means productivity and

problem-solving, and by extension, the output is what matters when defining creativity, the answer is yes, computers are already creative [25]. However, defining creativity from outputs falls short in relation to more complex forms of creativity, especially that akin to human creativity. Hertzmann's argument echoes Ludwig Wittgenstein, who believed that we need first to clarify our linguistic and conceptual habits when we want to understand waht we mean by terms like "intelligence" or "creativity" [3]. In other words, since language is a tool we need to find a proper contextual definition for evaluating whether AI systems are "creative", or their outputs can be considered art .

In fact, algorithms might model some aspects of how human creativity works, but creativity is much more complex. Creative processes involve moments of inspiration and are often open-ended [53, 24], attributes that current AI algorithms do not have, and thus, it becomes difficult to consider AI creative [25].

Other researchers have tried to elaborate a manifesto on attributes that define creativity as a socio-cultural phenomenon [19]. They argue that creative is a complex multidimensional system constituted by psychological, social and material phenomenon. This system is culturally mediated action, relational and meaningful. Anna Jordanous uses an algorithm to find keywords to find an empirical definition of creativity, with attributes such as "originality", "autonomy", or "discovery", defining creativity [31].

There are some authors that avoid extreme positions in relation to AI, since this this type of vision relies on a continuous subordinate or dominant roles between machines and humans [44]. For instance, according to Gilbert Simondon, there is a biological and technological evolution that does not separate nature and technics, since what is defined by human nature is already part of a technological system [50]. Haraway and Braidotti prompt us to rethink what we understand as human in terms of assemblages with machines and other species [35, 6].

Some other researchers and artists warn about the mislead and dangers of anthropomorphizing AI systems, as they shift the responsibility from the people and organizations that build these systems [30]. Furthermore, the same group of researchers and artists deny any of these AI systems' outputs to be called art, since they define art as

a uniquely human endeavor connected to human culture and experience. To this end, the next section will review some of the fundamental psychological creativity theories, which will provide a deeper context for defining art and creativity.

Given the advancement of generative AI, and as stated earlier, algorithms, for the first time, have the potential to become co-creators of the artistic process. Tools are by definition passive agents that humans utilize in their work. While it is important to resist anthropomorphizing AI algorithms and to recognize that these technologies lack autonomy and human characteristics, they possess the capability to assist humans in brainstorming, dialogue, and sketching out initial concepts. The debate over whether to label AI as a co-creator, collaborator, or an advanced tool is largely semantic, yet it underscores the nuanced role that such systems play in the creative process.

On Fundamental Psychological Creativity Theories

This section summarizes some fundamental psychological creativity theories from the mid nineties.

Historically, creativity was placed in the out-self, and it was believed to be mystical and divine, a factor that might have affected its potential for study in scientific fields [54]. Equally damaging to the scientific study of creativity has been viewing creativity only under a popularized pragmatic lens, with Edward De Bono and Osborn as main proponents of this approach, both agnostic to theoretical roots and strictly focused on practice. While De Bono suggested some tools such as focusing on aspects of ideas that are pluses, minuses, and interesting (PMI), or that of "thinking hats" that allow individuals to metaphorically structure their thinking into blocks or components, such as white hat for data-based thinking or black for critical thinking, Osborn [45] developed the concept of brainstorming to help people solve problems creatively in a collective and constructive manner rather than in a critical and inhibitory way.

The first major theoretical approach to the study of creativity that occurred in the twentieth century was the psychodynamic approach and it is based on the tension between conscious reality and unconscious drives. Freud [17] proposed that creative work

is a way to express unconscious wishes in a publicly acceptable fashion. Kris [34] introduced the concepts of adaptive regression and elaboration for the study of creativity. While adaptive regression, also known as primary process, refers to the unmodulated thoughts that occur, principally during sleep, drug intoxication, or psychoses, elaboration or secondary process, refers to transforming such primary process material through a reality and ego-controlled lens. Even though the psychodynamic approach offers insight into creativity, it suffered from a lack of resources from the scientific psychology field. Moreover, this approach relied on case studies of eminent creators, which sits opposite to controlled and experimental methods on larger samples, as valued by the scientific psychology community.

The rarity of such eminent creators had limited research in creativity and pushed Guilford [22] to start psychometric approaches in everyday subjects, using paper-and-pencil tasks that mainly tested divergent thinking. One of these tests was the Unusual Uses Test which the objective was to think of as many uses as possible for a common object such as a brick. Such divergent thinking tests became the de-facto procedures for measuring creativity due to their simplicity and objective assignment. However, such psychometric revolution of measuring creativity also had a negative impact, often criticized as trivial and inadequate.

Another approach was the cognitive approach, which seeks to understand mental representations and processes underlying creative thought. For example, Finke and his colleagues [16] developed the Geneplore model which distinguishes two main processes in a creative thought: the generation phase where individuals construct preinventive structures with properties to promote discovery, and the exploration phase, where such properties are further explored to come up with creative ideas. On a parallel but opposite track there is work on understanding creativity under a social-personality approach which focuses on personality, motivational and environmental variables. Researchers have found common personality traits that might characterize creative people such as self-confidence, aesthetic orientation and risk-taking. Others found that intrinsic motivation is important for creativity. At a societal level, cultural diversity, availability of

resources and creative value are factors of creativity. Furthermore, Wehner, Csikszentmihalyi and Magyari-Beck [62] found that different fields see creativity differently and describe it in terms of the fable of the blind men and the elephant.

Sternberg [54] names a few confluence approaches to the study of creativity, as it seems like multiple components must converge for creativity to occur. For instance, Amabile [1] describes creativity as a confluence of intrinsic motivation, domain-relevant knowledge, and creativity-relevant skills. Csikszentmihalyi [9, 10] describes a "systems" approach to highlight the interaction between individual, domain, and field. Sternberg and Lubart [55, 57, 58, 56] describe an investment theory of creativity where creativity requires a confluence of six resources such as intellectual abilities, knowledge, styles of thinking, personality, motivation, and environment. Overall, confluence theories seem to provide a holistic view of creativity, whereas unidisciplinary approaches have tended to view only parts of the phenomenon resulting in an incomplete and narrow vision of creativity.

Experimental research has also manipulated the perception of the problem itself, identifying that creative work often results from problem finding [18] rather than problem solving. An experiment in different media information by Greenfield et al. [21] concluded that video information elicited low originality scores as compared to textual or audio information. Another experiment using insight tests by Weisberg and Alba [63] led them to reject the Gestalt prediction about fixity and sudden insights, since participants could not resolve the problem after receiving additional hints.

On experiments related to mental synthesis and perception, Finke [14] tested creativity by generating forms from objects parts, and found that instructions can inhibit creativity, either by misdirecting individuals towards unproductive lines of thought or by eliciting emotional and attitudinal blocks of rigidity. On a different track, Hoppe and Kyle [27] and Smith [51] demonstrated that affect can influence creative performance, measuring performance at different levels of induced affective states and anxiety.

Martindale and colleagues [39, 38, 40] found correlations between higher level of arousal and performance on tests of creative thinking. They also found that creative

subjects have good short-term control arousal, or sensitivity as well as attention focus. This is related to the concept that original ideas are remote, and the higher the capacity for broad attention the more likely remote ideas will be found.

While these theoretical frameworks provide invaluable insights, they also highlight the inherent challenges in fully capturing the essence of creativity. The dynamic interplay between individual traits, environmental factors, and societal influences suggests that creativity is not just an individual phenomenon but a process deeply embedded in the fabric of culture and context. This realization prompts us to consider how contemporary tools like AI and digital media might intersect with these established theories, potentially reshaping our understanding of creativity in the digital age. As we integrate these historical perspectives with modern technological advancements, we find ourselves at the cusp of a new era of creativity, one that is increasingly collaborative, interdisciplinary, and fluid. This juncture invites further exploration into how emerging technologies can complement, challenge, and transform traditional notions of creativity, opening doors to novel forms of artistic expression and discovery.

Reflection on Creativity in Algorithms

As we have seen in the previous sections, it is still unclear how to frame and evaluate creativity within the field of artificial intelligence. As the field of AI is vast and comprises many subfields, I shall focus only on those subfields that are relevant to this PhD research, such as vision, perception, RL, and natural language processing (NLP). The aforementioned fields are what I am referring to when using the acronym AI hereon. The first critical issue is to understand a meaningful mapping between human creativity and AI creativity. There has been a considerable amount of effort throughout the late nineteenth century until today in trying to consolidate theories and models, some of which are complementary and others contradictory, to understand human creativity, and yet there seems to be a loose consensus about defining a criterion for creativity [32, 54]. Judging creativity and novelty is partly a subjective matter, and it depends on a contextual human judgement. Creativity is sometimes associated with self-consciousness and

unexpectedness. This would make any random and surprising process "creative". However, we expect creativity to be meaningful as well. Autonomy is another attribute that might define creativity. In the context of algorithms, any program that deviates from the original intention of the programmers, should also be considered "creative". Even in human history, the definition of creativity has changed. In ancient times, creativity was associated with the ability to remember, reconstruct, and reproduce things that already existed, and in medieval times, true creativity was associated with divine intervention [3].

Even though there have been some major creative breakthroughs in AI in the last decade, such as the advent of generative models [20, 26, 49, 46] based on prior technological advancements [33, 61, 52, 47, 48], understanding and evaluating AI creativity is still very premature and requires a deeper survey of recent literature. Nonetheless, maintaining a more critical approach for deeming machines—software and hardware—as intelligent or creative as a result of a particular task and not as a whole, as Colton refers to its own algorithm in [7], seems a better place to start.

To position creativity within my work, it is necessary to distinguish between creative scientific breakthroughs and creative artistic breakthroughs, although in some cases this separation is not clear. Such separation resembles some coherence with the proposed psychometric measures by Guilford [22] based on divergent thinking, aimed at revealing self-expression and spontaneity, and convergent thinking, insight and problem-solving tests that take a more orderly and logical approaches [32]. The cited work in the previous paragraph constitutes a selection of some of the most important and influential scientific articles in AI, and all of them are creative because they are novel, surprising and valuable, following the definition of creative ideas given by Boden [5], as well as durable [54]—definition of creativity, pg. 94. They build the technology necessary for other creative applications to exist, such as the work done by Zhu and colleagues [29, 67, 66] on conditional generative models and more precisely, image-to-image translation tasks. These applications tend to lie more on the artistic side of AI, although they also have, to some extent, scientific novelty. Sometimes, works like generative adver-

sarial networks (GANs) by Goodfellow et. al., [20] are both fundamental scientific and artistic breakthroughs, and thus its creativity can be measured by taking analytical and expressive approaches.

In this regard, and directly related to my research, painting algorithms have arguably been considered a quintessential AI-creative process [8, 5] explicitly citing the seminal work AARON by Cohen [41]. While painting is an unambiguously creative domain because, among other things, the number of forms used in paintings may be indicative of flexibility and originality [54], my work situates such creativity in between scientific and artistic value. That is, just because an algorithm can contribute to a scientific novelty—an unseen method—to generate paintings, does not mean that the paintings have artistic value. This is the case for recent painting algorithms such as Paint Transformers [36] or learning-to-paint [28]. They both offer novel computational models, but their paintings are reconstructions of input images; in the case of Paint Transformers look like digital oil painting, and in the case of learning-to-paint they look like plain digital paintings. Furthermore, artistic value is generally gauged and recognized by a community, given by its context, which increases the complexity on evaluating the outputs of these algorithms. In this dissertation, some creativity is achieved by (a) generating a new method to paint and (b) achieving unseen implicit styles. I say some creativity because styles are fixed at the beginning of the process and because the algorithm models a specific task, in opposition to a greater artificial artist—this is discussed later. In this context, my understanding of creativity, at its highest level, closely follows the definition provided by Boden and Stenberg, and it should be applied in both, scientific (novelty in problem solving) and artistic (novelty in artistic ability) contribution. Next, I will provide more details as to how to assess creativity and the criteria applied to the method's outcomes. Note that this is just one of many ways to assess creativity, and as we have seen, this concept is contingent to contextual references.

A natural approach to achieving an algorithm that can be considered creative might be starting with a psychological creativity theory as inspiration. Among the list of theories presented in [54], the flow theory by Csikszentmihalyi [11], which describes an

optimal state of intrinsic motivation, the Geneplore model by Finke and colleagues [15], by which there are two processes of generation and exploration of ideas, the primary and secondary process of cognition supported by Kris, Mednick and Mendelsohn [34, 42, 43], and Martindale's principle of least effort [37] might be feasible to describe in computational means. In fact, Elgammal et al. [12], loosely build a GAN model based on Martindale's theory and define style novelty by finding regions in the latent space that deviate from style norms, yet without moving too far away from the distribution of art—to avoid falling into the negative hedonic response [4, 65]. There seems to be a mismatch, however, between trying to model a psychological creativity theory and the mainstream trend in computer vision of modeling a specific task, be it an image-to-image translation, a painting algorithm, or an object detection task. After all, individual tasks do not need complex creativity theories in order to work. This research is an example of this phenomenon: we model a (partially creative) task of painting with implicit novel styles.

A more ambitious work would rely on a theory more heavily, and would not model just a task, but an artificial being or system. This would help have a more interpretable mapping between human and AI creativity, and the creative act would be, at the very least, more complete. Under this scenario, and assessing and evaluating creativity by its process as well as the product, and targeting an H-creativity [5], a larger entity, i.e. system, would need to be modeled, such as a probably incomplete representation of an artificial "artist", and not just the task of painting itself. To do this, a model should have fundamental constituents such as a memory block which might consist of visual and textual information to recreate knowledge, and a self-evaluative mechanism to recreate self-awareness of creativity—one of the traits that creative subjects share. This is further supported by Colton: *"A poet with no critical ability to judge its own work is no poet at all"* [8]. While modeling personality is completely out of the question, I might as well imagine modeling some environment that fosters creative behavior, and thus tap into the systems theory by Csikszentmihalyi and working environments by Amabile and Gryskiewicz [2], although this might be difficult. Assuming a successful outcome,

that of having novel paintings with respect to its knowledge of the world, the process would be deemed creative and novel, as there is no other algorithm in the literature of painterly rendering that attempts this.

Thus, it seems somewhat clear that assessing whether an algorithm is creative depends on both context and outcome, but also on the main objective. If the objective is to provide a novel scientific contribution, a peer-reviewed publication in top computer vision conferences would probably justify its process-based (not product) creativity. Whereas if the objective is to produce artistic novelty, a knowledgeable audience in AI and art would be required to assess a product creativity. But that alone would not be enough. Artistic novelty should be abundant and diverse, and as Colton mentions in [8], a novel output should not be hand-selected by humans, but rather by the algorithm itself. There are common metrics for this in AI, concretely the Fréchet Inception Distance (FID), which assesses both quality and variability of the generated output. Another factor to measure creativity is controllability and more importantly, interpretability. Being able to interpret the process of, in this case, neural networks, is fundamental to understanding the inner workings of the creative process and drawing meaningful conclusions between outputs and algorithms—or product and process. This becomes crucial in open ended search because, a priori, results might not be even comparable to any other painting and therefore, assessing whether an image is novel but relevant might be problematic. In such cases, interpretability helps to understand and evaluate those results. If the proposed system can automatically generate styles and self-assessments or predictions of such styles, guiding its painting process by its own judgements, then it would become apparent to deem it creative or even intelligent.

In general, advancing the field of creative AI differs significantly from advancing the creative use of AI. The former involves developing AI systems and algorithms that exhibit or facilitate creative processes, often pushing the boundaries of what AI can generate or conceptualize autonomously. This approach seeks to imbue AI with capabilities that mimic, or even surpass, human creativity, focusing on the AI's ability to produce novel, valuable, and surprising outcomes. On the other hand, enhancing the creative

use of AI focuses on how humans can utilize AI as a tool or collaborator to augment their own creative processes. This perspective emphasizes AI as an extension of human creativity, providing new means for expression and innovation. By integrating AI as a collaborative tool in the creative process, we can expand human artistic capabilities while preserving the inherent value of human intuition and imagination in art.

In the concluding chapter of this thesis, I will offer a comprehensive summary of the key findings, discuss the limitations encountered, and propose directions for future research in this field.

References

[1] Teresa M Amabile. “The social psychology of creativity: A componential conceptualization.” In: *Journal of personality and social psychology* 45.2 (1983), p. 357.

[2] Teresa M Amabile and Nur D Gryskiewicz. “The creative environment scales: Work environment inventory”. In: *Creativity research journal* 2.4 (1989), pp. 231–253.

[3] Emanuele Arielli and Lev Manovich. “AI-aesthetics and the Anthropocentric Myth of Creativity”. In: (2022).

[4] Daniel E Berlyne. “Aesthetics and psychobiology”. In: *Journal of Aesthetics and Art Criticism* 31.4 (1973).

[5] Margaret A Boden. “Creativity and artificial intelligence”. In: *Artificial intelligence* 103.1-2 (1998), pp. 347–356.

[6] Rosi Braidotti. “Posthuman, all too human: Towards a new process ontology”. In: *Theory, culture & society* 23.7-8 (2006), pp. 197–208.

[7] Simon Colton. “Creativity Versus the Perception of Creativity in Computational Systems”. In: *Proc. AAAI spring symposium: creative intelligent systems*. 2008.

[8] Simon Colton, Geraint A Wiggins, et al. “Computational creativity: The final frontier?” In: *Proc. ECAI*. Vol. 12. Montpelier. 2012, pp. 21–26.

[9] M Csikszentmihalyi. *Creativity. NY*. 1996.

[10] Mihaly Csikszentmihalyi and Mihaly Csikszentmihalyi. *Society, culture, and person: A systems view of creativity*. Springer, 2014.

[11] Mihaly Csikszentmihalyi and Mihaly Csikzentmihaly. *Flow: The psychology of optimal experience*. Vol. 1990. Harper & Row New York, 1990.

[12] Ahmed Elgammal et al. “Can: Creative adversarial networks, generating” art” by learning about styles and deviating from style norms”. In: *arXiv preprint arXiv:1706.07068* (2017).

[13] Ziv Epstein et al. “Art and the science of generative AI”. In: *Science* 380.6650 (June 2023), pp. 1110–1111. ISSN: 1095-9203. DOI: 10.1126/science.adh4451. URL: http://dx.doi.org/10.1126/science.adh4451.

[14] Ronald A Finke. *Creative imagery: Discoveries and inventions in visualization*. Psychology press, 2014.

[15] Ronald A Finke, Thomas B Ward, and Steven M Smith. “Creative cognition: Theory, research, and applications”. In: (1992).

[16] Ronald A Finke, Thomas B Ward, and Steven M Smith. *Creative cognition: Theory, research, and applications*. MIT press, 1996.

[17] Sigmund Freud. “I. Creative Writers and Daydreaming”. In: *Literature and psychoanalysis*. Columbia University Press, 1983, pp. 19–28.

[18] Jacob W Getzels. “Problem-finding and the inventiveness of solutions.” In: *The Journal of Creative Behavior* (1975).

[19] Vlad Petre Glaveanu et al. “Advancing creativity theory and research: A sociocultural manifesto”. In: *The Journal of Creative Behavior* 54.3 (2020), pp. 741–745.

[20] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).

[21] Patricia Greenfield et al. “Television and radio experimentally compared: Effects of the medium on imagination and transmission of content”. In: *meeting of the Society for Research in Child Development, Boston*. 1981.

[22] J Guilford. *Creativity. American Psychology. 5 (9), 444–454*. 1950.

[23] Aaron Hertzmann. “Can Computers Create Art?” In: *Arts* 7.2 (2018). ISSN: 2076-0752. DOI: 10.3390/arts7020018. URL: https://www.mdpi.com/2076-0752/7/2/18.

[24] Aaron Hertzmann. “Toward modeling creative processes for algorithmic painting”. In: *arXiv preprint arXiv:2205.01605* (2022).

[25] Aaron Hertzmann. *What is Creativity? Can Computers Be Creative?* URL: https://aaronhertzmann.com/2023/09/27/what-is-creativity.html.

[26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[27] Klaus D Hoppe and Neville L Kyle. “Dual brain, creativity, and health”. In: *Creativity Research Journal* 3.2 (1990), pp. 150–157.

[28] Zhewei Huang, Wen Heng, and Shuchang Zhou. “Learning to Paint With Model-based Deep Reinforcement Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[29] Phillip Isola et al. “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[30] Harry H. Jiang et al. “AI Art and its Impact on Artists”. In: *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*. AIES ’23. , Montréal, QC, Canada, Association for Computing Machinery, 2023, pp. 363–374. ISBN: 9798400702310. DOI: 10.1145/3600211.3604681. URL: https://doi.org/10.1145/3600211.3604681.

[31] Anna Jordanous. “Defining Creativity: Finding Keywords for Creativity Using Corpus Linguistics Techniques.” In: *ICCC*. 2010, pp. 278–287.

[32] James C Kaufman and Robert J Sternberg. *The Cambridge handbook of creativity*. Cambridge University Press, 2010.

[33] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[34] Ernst Kris. "Psychoanalytic explorations in art." In: (1952).

[35] Hari Kunzru. *Antropologia do ciborgue: as vertigens do pós-humano*. Vol. 5. Autêntica Editora, 2000.

[36] Songhua Liu et al. "Paint Transformer: Feed Forward Neural Painting With Stroke Prediction". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 6598–6607.

[37] Colin Martindale. *The clockwork muse: The predictability of artistic change.* Basic Books, 1990.

[38] Colin Martindale and James Armstrong. "The relationship of creativity to cortical activation and its operant control". In: *The Journal of genetic psychology* 124.2 (1974), pp. 311–320.

[39] Colin Martindale and James Greenough. "The differential effect of increased arousal on creative and intellectual performance". In: *The Journal of Genetic Psychology* 123.2 (1973), pp. 329–335.

[40] Colin Martindale and Nancy Hasenfus. "EEG differences as a function of creativity, stage of the creative process, and effort to be original". In: *Biological psychology* 6.3 (1978), pp. 157–167.

[41] Pamela McCorduck. *Aaron's code: meta-art, artificial intelligence, and the work of Harold Cohen*. Macmillan, 1991.

[42] Sarnoff Mednick. "The associative basis of the creative process." In: *Psychological review* 69.3 (1962), p. 220.

[43] Gerald A Mendelsohn. "Associative and attentional processes in creative performance." In: *Journal of personality* (1976).

[44] Andreia Machado Oliveira. “Future Imaginings in Art and Artificial Intelligence”. In: *Journal of Aesthetics and Phenomenology* 9.2 (2022), pp. 209–225. DOI: 10.1080/20539320.2022.2150467. eprint: https://doi.org/10.1080/20539320.2022.2150467. URL: https://doi.org/10.1080/20539320.2022.2150467.

[45] AF Osborn. “Applied Imagination (Rev Ed.) New York: Charles Scribner’s Sons”. In: (1953).

[46] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 27730–27744.

[47] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *arXiv preprint arXiv:2103.00020* (2021).

[48] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8748–8763.

[49] Aditya Ramesh et al. “Zero-shot text-to-image generation”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8821–8831.

[50] Gilbert Simondon. “On the mode of existence of technical objects”. In: *Deleuze Studies* 5.3 (2011), pp. 407–424.

[51] Gudmund JW Smith and Gunilla Van der Meer. “Creativity in old age”. In: *Creativity Research Journal* 3.4 (1990), pp. 249–264.

[52] Yang Song and Stefano Ermon. “Generative modeling by estimating gradients of the data distribution”. In: *Advances in neural information processing systems* 32 (2019).

[53] Kenneth O Stanley and Joel Lehman. *Why greatness cannot be planned: The myth of the objective*. Springer, 2015.

[54] Robert J Sternberg. *Handbook of creativity*. Cambridge University Press, 1999.

[55] Robert J Sternberg and Todd I Lubart. “An investment theory of creativity and its development”. In: *Human development* 34.1 (1991), pp. 1–31.

[56] Robert J Sternberg and Todd I Lubart. “Buy low and sell high: An investment approach to creativity”. In: *Current Directions in Psychological Science* 1.1 (1992), pp. 1–5.

[57] Robert J Sternberg and Todd I Lubart. *Defying the crowd: Cultivating creativity in a culture of conformity*. Free press, 1995.

[58] Robert J Sternberg and Todd I Lubart. “Investing in creativity.” In: *American psychologist* 51.7 (1996), p. 677.

[59] Ilya Sutskever. *The Mastermind Behind GPT-4 and the Future of AI*. URL: https://www.youtube.com/watch?v=SjhIlw3Iffs.

[60] Ilya Sutskever. *What’s Next for Large Language Models (LLMs)*. URL: https://www.youtube.com/watch?v=UHSkjro-VbE.

[61] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[62] Lucille Wehner, Mihaly Csikszentmihalyi, and István Magyari-Beck. “Current approaches used in studying creativity: An exploratory investigation”. In: *Creativity Research Journal* 4.3 (1991), pp. 261–271.

[63] Robert W Weisberg and Joseph W Alba. “An examination of the alleged role of” fixation” in the solution of several” insight” problems.” In: *Journal of experimental psychology: general* 110.2 (1981), p. 169.

[64] Chenfei Wu et al. “Visual chatgpt: Talking, drawing and editing with visual foundation models”. In: *arXiv preprint arXiv:2303.04671* (2023).

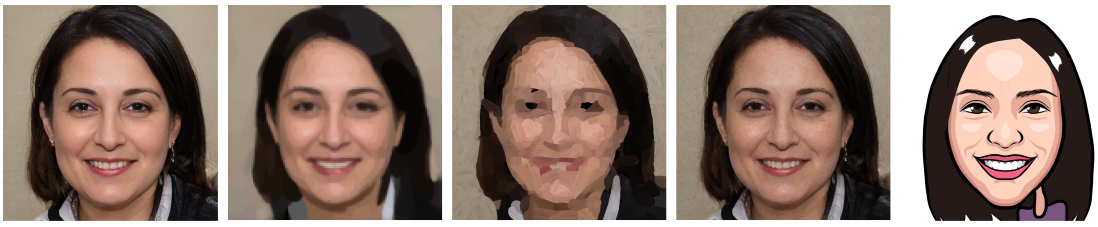
[65] Wilhelm Max Wundt. *Principles of physiological psychology*. Vol. 1. Sonnenschein, 1904.

[66] Jun-Yan Zhu et al. “Toward multimodal image-to-image translation”. In: *Advances in neural information processing systems* 30 (2017).

[67] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. 2017.

Chapter 7

Conclusion



(a) Input (b) Im2Painting (c) Semantic-Based (coarse) (d) Semantic-Based (realistic) (e) Cross-modal alignment (avatar)

Figure 7.1: Comparison of the presented methods' outputs given the same input: (a) Original image. (b) Realistic portrait generated by our method Im2Painting, presented in Chapter 3. (c,d) Coarse-to-fine realistic portrait generated by our Semantic Segmentation-Based method, presented in Chapter 5. (e) Avatar generated by our method presented in Chapter 6.

In the concluding section of my dissertation, I revisit the goals and inquiries that guided my research. This part encapsulates the key takeaways from each chapter and their findings. Here, I will discuss these findings within the realm of the current state of technological advancements and their impact in the art field. To wrap up, I will suggest potential avenues for future investigations in this field.

7.1 Research Summary

This thesis originated from a deep inquiry into the evolving relationship between art and technology, with a particular focus on the role of artificial intelligence (AI). After a brief introduction presented in the initial chapter, Chapter 2, delves into the historical interplay between the evolution of art corresponding to technological advances. It examines pivotal questions that have persisted through time, such as the potential of technology to disrupt traditional artistic practices, the creative capabilities of machines, and the role of generative algorithms in producing distinct or original imagery. A key insight from this exploration is the realization that technology has not diminished art; rather, it has served as a catalyst for the emergence of new artistic forms and expressions [9, 6].

In an era marked by rapid advancements in AI, this thesis represents a journey into understanding, developing, and analyzing algorithms capable of creative output. This involves a detailed investigation into how these algorithms transform input images into varied styles, thereby contributing to the ongoing discourse on AI's capabilities within the artistic domain [11, 15, 6, 1, 9, 18].

The structure of this research is centered around three primary objectives, each addressed in a separate technical chapter. These objectives encompass the exploration of AI algorithms in the context of artistic *stylization, control*, and *identity preservation*. This structured approach not only facilitates a comprehensive understanding of the current landscape of AI in art but also lays the groundwork for future explorations in this exciting and ever-evolving field. We show a comparison of the outputs of all our methods presented in this thesis in Figure 7.1. We also show a comparison of our two painting methods on the landscape domain in Figure 7.2. Columns a and c show different painting strategies from our semantic-based method, presented in Chapter 5. Columns b and d show different styles given by our Im2Painting method presented in Chapter 3. We see an improvement in quality from our semantic-based method; paintings have a more natural feeling than our Im2Painting method.

Stylization This subject is one of the most important problems in the field and the research community has been actively involved in the matter. Stylization, at its core, refers to the process of transforming images or visuals into distinct styles, typically mimicking various artistic techniques or representational aesthetics. The state of the art in stylization algorithms has advanced significantly in recent years, driven by breakthroughs in machine learning, particularly in deep learning and neural network architectures. However, most of these algorithms work in pixel space, and just a few of them work on vector space. It is worth mentioning that stylization and control are, to some extent, related. To recap, some areas of stylization are:

- **Neural Style Transfer:** A landmark development in this area was the introduction of neural style transfer algorithms in 2015. These algorithms use convolutional neural networks (CNNs) to apply the artistic style of one image (the style reference) onto the content of another image (the target image). The pioneering work in this field by Gatys et al. [7] demonstrated the potential of deep learning in capturing and transferring artistic styles. This approach has been refined and expanded upon in numerous subsequent studies.

- **Real-Time Stylization**: Following the initial breakthroughs, there was a significant push towards achieving real-time stylization, making it more accessible and practical for various applications, like videos. This led to the development of models that are optimized for speed without substantial loss in quality. Techniques like model pruning, quantization, and the development of more efficient network architectures contributed to this advancement [13].

- **Generative Adversarial Networks (GANs),** emerged as a powerful tool in the domain of artistic stylization. They consist of two neural networks, a generator and a discriminator, that work in tandem to produce highly realistic outputs. GANs have been used to generate entirely new images in specific artistic styles, offering a level of creativity and uniqueness in the results. Some notable work includes image-to-image translation methods [14, 28].

- **Control and Customization:** More recent developments in stylization algorithms focus on providing greater control and customization to the users. This includes controlling the extent and nature of stylization, localizing style applications to specific regions of the image, and even blending multiple styles in a cohesive manner. Notable work has been done in inpainting[1] algorithms [26].

- **Semantic Understanding and Context-Aware Stylization:** Advanced algorithms are now incorporating semantic understanding to apply styles in a context-aware manner [25, 17]. This involves recognizing different elements in the image and applying styles in a way that respects the semantic and spatial relationships within the image.

- **Diffusion algorithms** [12, 24, 21, 19, 27], particularly in the context of machine learning and artificial intelligence, have recently gained significant attention due to their remarkable capabilities in generating high-quality, coherent, and diverse images. These algorithms represent a novel approach in the field of generative models, which are systems designed to create new data samples that resemble a given dataset. They do not only generate high-quality images, but are able to generate a vast array of styles for the same content.

In this thesis, our focus shifts away from conventional stylization approaches, such as pixel-space modifications applied after rendering from parametric models. Instead, we delve into the intricacies of stylization within the framework of parametric-based models themselves. This exploration is anchored in manipulating various loss functions and optimization techniques, as extensively detailed in Chapters 3 to 5.

[1] Inpainting algorithms refer to a class of techniques used in image processing and computer graphics to reconstruct missing or damaged parts of an image. The goal of these algorithms is to fill in the gaps in an image in a visually plausible manner, such that the filled-in regions are undetectable or blend seamlessly with the surrounding area. Inpainting has a wide range of applications, including restoring old photographs or artworks, removing unwanted objects from images, and repairing areas of images that are corrupted due to data loss or sensor errors.

Particularly, Chapter 3 presents an innovative approach to stylization, rooted in the concept of *economy*. By strategically placing a minimal number of brush strokes, we not only achieve efficiency but also establish a foundation for more sophisticated and controlled stylization processes. This approach is pivotal in creating distinctive artistic styles, as it involves careful management of stroke placement. Central to our argument is the idea that the economical use of strokes serves as a robust test of an algorithm's ability to effectively optimize strokes to accurately recreate images, a fundamental aspect of many natural painting styles.

Our methodology leverages deep learning techniques to map an input image to a corresponding vector of strokes that artistically stylize the image. This process, however, necessitates training on domain-specific datasets to ensure the model's ability to generalize effectively to new images within the same domain. Experimentation with different loss functions and stroke budgets results in the creation of a variety of styles, demonstrating the flexibility and adaptability of our approach. While our method has shown marked superiority in generating realistic paintings compared to previous work, it is important to note that the range of styles it can produce is currently limited. Future work could extend these capabilities, potentially broadening the spectrum of artistic styles that can be achieved through our proposed techniques.

Control The intricate relationship between control and stylization forms a central theme of this thesis. While Chapters 4 and 5 explore a variety of stylized outputs, it is crucial to recognize that these styles originate fundamentally from the meticulous control of the painting process. This control is not merely a secondary effect; it is the primary driver of the unique styles that have been developed and discussed. Appreciating this dynamic is key to fully understanding the scope and implications of the techniques and results that have been explored.

In Chapter 4, we delve into the realm of optimization algorithms, constructing a novel algorithm inspired by early procedural and optimization strategies [8]. Our initial focus is on low-resolution paintings, with the aspiration of applying these principles

to larger-scale works. We begin with a basic setup for optimizing stroke parameters, assessing its efficiency through a simplistic approach as outlined in Algorithm 1. This stage illuminates the critical importance of stroke parameter initialization strategies and their significant impact on the final artwork. Additionally, our exploration into various loss functions sheds light on their relationship with artistic styles. Progressing further, we introduce a hierarchical, coarse-to-fine strategy that simplifies the optimization process, leading to more refined and controlled artistic outputs.

Building on the foundation laid in Chapter 4, Chapter 5 advances this work by developing a robust, high-resolution algorithm. This algorithm, drawing inspiration from human visual working memory and foveal vision, utilizes semantic segmentation networks and dynamic attention maps to exert distinctive ways of gaining control over the artistic process. The result is the ability to create a diverse array of styles, ranging from realism to abstraction, achieved more efficiently and on a larger scale than previous methods without sacrificing quality. Both quantitative and qualitative evaluations underscore the superiority of our approach.

Identity Preservation In the stylization of human faces, maintaining the subject's identity is paramount. Many algorithms, much like some human artists, tend to compromise personal identity during the stylization process. This issue is particularly prevalent in caricatures and digital avatars on gaming and social media platforms, where the avatars created often bear little resemblance to the actual user.

Addressing this challenge, Chapter 6 introduces a deep learning framework for translating images into parametric avatars. Although our focus is not on stylization per se, the framework is versatile and applicable across various styles. The primary objective is to preserve the identity of the subject in the input image to the greatest extent possible. A significant challenge encountered is the limited data available for training deep learning models. With less than 500 paired samples, there is a risk of overfitting and lack of generalization. Our solution involves performing translation within the latent space, rather than the original input representation, drawing inspiration from

transfer learning and zero-shot learning techniques [23, 16]. This process is conducted in two stages: initially, modality-specific deep networks are trained on large datasets to learn rich latent representations of their respective modalities. The second phase involves training a smaller network to bridge these modalities, utilizing our limited paired data. Through this approach, and the application of various loss functions and regularization techniques, our model demonstrates superior performance compared to existing methods.

7.1.1 Contextual Relevance

It is important to situate this research within the context of current diffusion image-based models for conditional image generation [12, 5, 21, 19, 27, 20]. The popularity of such models have exploded in the last two years, with Midjourney, Stable Diffusion [20], and Dalle-3 [2] being the most popular.

A natural question to ask is whether the parametric-based models presented in this thesis are still relevant for stylization of images, given the research efforts in pushing text-to-image diffusion models. These models are being widely adopted by users to generate novel images due to the simplicity of use, novelty, and high quality. For the first time, a user can generate a high-resolution image from a text prompt in seconds. As summarized in Table 1.1, both models, diffusion and parametric, have fundamental differences, and there are some advantages of parametric models over pixel-based, even though they are not as publicly adopted by users.

First and foremost, this research is interested in generating artistic representations from images in a similar way as humans do, that is, by the use of strokes, or primitives. Stroke-based processes are completely different than pixel-based methods, as they need to abstract the information in localized groups and produce the result in a sequential manner. This means that the final construct is being materialized by a chain of steps which each one of the constituents in this chain depends on the previous one. Such a method allows then to model human aspects involving decision making. Potentially, these methods could also steer away from the initial intention as the painting progresses,

making changes “ad-hoc”.

Besides the process, which is an area of research in itself, parametric outcomes allow for other end results than just a static image. Instead, robotic systems can be used to generate the paint [3, 22], or the result can be easily transferred to other animation or 3D software. It is worth noting that diffusion-based models are already working to make a video out of a static image [4].

Vector-based ML models, as opposed to pixel-based ones, offer unique advantages that can catalyze new forms of artistic output. Unlike pixel-based models, which manipulate and generate images at the pixel level, vector-based models work with geometric primitives such as lines, curves, and shapes. This fundamental difference allows vector-based models to perform operations like subtracting strokes or reversing sequences, which are not inherently feasible with pixel-based models. Such capabilities enable more dynamic and fluid alterations in the artwork, facilitating a level of control and manipulation that can lead to unexpected and innovative artistic creations.

7.2 Limitations, Implications for Future Technology, and Future Work

The research presented in this thesis has significant implications for future technological developments in the art and AI domains. Our methodologies have indeed laid the groundwork for future stylization work by providing a greater control over the painting process. Through segmentation and efficient stroke placement, we have opened new avenues for representing inputs in various artistic styles. Specifically, our Im2Painting method, grounded in the principle of economical art, suggests a foundational approach for future exploration in stylization.

The introduction of a controlling mechanism, as demonstrated in our work, marks an advancement over previous parametric methods. This level of control not only empowers users by allowing them to guide the painting process but also lays the groundwork for more sophisticated integration into creative software platforms. Such integration

could augment artistic expression and technological innovation, offering new opportunities for artists and developers alike.

However, in reflecting upon this work, it is crucial to articulate its limitations to provide a clear scope for future research directions. While considerable improvements have been made in the areas of controllability, stylization, and identity preservation, there are several domains were further exploration is neeeded. A notable limitation encountered in the Im2Painting and Segmentation-Based Painting methods, presented in Chapter 3 and Chapter 5, respectively, is their limited range in artistic style variation. This shortfall stems from their design which does not incorporate external stylistic imagery or emphasize stylistic diversity. Instead, these methods are built around optimizing the representation of input images through specific loss functions and optimization strategies, yielding more predictable results. When the artistic objective moves towards more interpretive goals—such as representational, conceptual, or abstract art—these predictable outcomes may not suffice. Future research should therefore aim to move beyond methods focused on high fidelity to explore the creation of art that thrives on ambiguity and interpretation, necessitating a shift away from traditional pixel-based loss functions.

One of our objectives in Chapter 6 was to maintain the individual's identity in avatar generation. Although our framework achieves this goal within the constraints of a limited database, it currently only produces a single frontal view of the avatar. A promising direction for further research is the generation of multiple views and 3D representations directly from parametric vectors, which can facilitate animation and a more holistic digital embodiment from just a single portrait photograph.

One key area where our work could have substantial impact is in the development of more intuitive and creative AI system in digital art software. By incorporating principles of controlled stylization and user-guided processes, future software could offer artists great flexibility and precision in their digital creations. This could lead to a new era of digital art where the boundary between human creativity and AI assistance becomes increasingly seamless. For instance, software could adapt in real-time to an artist's style or the thematic elements of their work, using learned parameters to suggest complemen-

tary strokes or color palettes, thus enhancing the creative workflow.

Furthermore, the educational implications of this work are noteworthy. By simplifying the complexities of artistic creation through controlled AI systems, we can make art more accessible to learners and educators. This could lead to the development of educational tools that use AI to teach art and creativity, potentially changing the way art is taught and learned. For example, AI-driven platforms could be designed to interpret students' artistic intentions and offer real-time guidance, augmenting current technologies with an adaptive environment. Since parametric models can be physically simulated or reenacted by robots, one can imagine a scenario in which a robot acts as a teacher, and students imitate the robotic motions, entertaining a human-robotic interaction powered by parametric-based models.

Our focus on identity preservation in our algorithms opens doors for more personalized and accurate digital representations in various applications, from gaming to virtual reality. User avatars not only enhance the immersive experience but also carry significant personal or cultural symbolism. This aspect of our work could lead to advancements in how digital avatars are created and used, ensuring they accurately reflect the user's identity.

Looking ahead, I propose several concrete avenues for future research and development:

In terms of technical refinement, future PhDs and MS theses could explore other parameterizations of the stroke model, delving into the integration of dynamic brush textures and pressure simulation to more closely mimic the variability found in physical painting techniques. Investigating in how to encode more unexpectedness into these algorithms remain an open question.

On evaluation metrics, the efficacy of our system can be better assessed by developing quantitative metrics that capture the subjective qualities of artistic works, such as aesthetic appeal or emotional resonance. These metrics could be validated through user studies involving artists and non-artist. However it has been argued that user studies in artistic-like outputs on computer graphics and vision are performative, that is, they re-

sult from reviewers' expectations that papers need to have some evaluation, rather than from a careful thought about the value and usefulness of the studies themselves [10].

Future studies should investigate algorithms that adapt to users' unique styles over time, learning from each interaction to better assist in the creative process. This could be done using user feedback in the loop, in contexts of deep learning and reinforcement learning. Likewise, understanding how parametric-based models can be tailored for specific applications such as real-time collaborative art frameworks needs more detailed research.

On a more conceptual level, there is a rich potential for interdisciplinary collaborations that bring together AI researchers, cognitive scientists, and artists to explore the convergence of creativity and technology. Such collaborations can yield new insights into the creative process and inform the development of AI systems that are sensitive to the nuances of human expression.

In conclusion, our work sets a foundation for exciting future research and development. The next steps include expanding the stylistic range of our methods, exploring new areas such as open-ended algorithms, and enhancing the capabilities for content abstraction and transformation. The journey of integrating AI into the creative process is just beginning, and the potential for future technological advancements in this field is vast and inspiring.

Figure 7.2: Comparison of two of the presented methods' outputs given the same input: (a,c) Paintings generated by our Semantic Segmentation-Based method, presented in Chapter 5. (b,d) Paintings generated by our Im2Painting method, presented in Chapter 3.

References

[1] Emanuele Arielli and Lev Manovich. “AI-aesthetics and the Anthropocentric Myth of Creativity”. In: (2022).

[2] James Betker et al. *Improving image generation with better captions*. 2023.

[3] Ardavan Bidgoli et al. “Artistic Style in Robotic Painting; a Machine Learning Approach to Learning Brushstroke from Human Artists”. In: *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2020, pp. 412–418. DOI: 10.1109/RO-MAN47096.2020.9223533.

[4] Andreas Blattmann et al. *Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets*. 2023. arXiv: 2311.15127 [cs.CV].

[5] Prafulla Dhariwal and Alexander Nichol. “Diffusion models beat gans on image synthesis”. In: *Advances in neural information processing systems* 34 (2021), pp. 8780–8794.

[6] Ziv Epstein et al. “Art and the science of generative AI”. In: *Science* 380.6650 (June 2023), pp. 1110–1111. ISSN: 1095-9203. DOI: 10.1126/science.adh4451. URL: http://dx.doi.org/10.1126/science.adh4451.

[7] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. “Image Style Transfer Using Convolutional Neural Networks”. In: *CVPR*. 2016.

[8] Aaron Hertzmann. “A Survey of Stroke-Based Rendering”. In: *IEEE Computer Graphics & Applications* 23.4 (2003).

[9] Aaron Hertzmann. “Can Computers Create Art?” In: *Arts* 7.2 (2018). ISSN: 2076-0752. DOI: 10.3390/arts7020018. URL: https://www.mdpi.com/2076-0752/7/2/18.

[10] Aaron Hertzmann. “The Curse of Performative User Studies”. In: *IEEE Computer Graphics and Applications* 43.6 (2023), pp. 112–116. DOI: 10.1109/MCG.2023.3315759.

[11] Aaron Hertzmann. *What is Creativity? Can Computers Be Creative?* URL: https://aaronhertzmann.com/2023/09/27/what-is-creativity.html.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[13] Haozhi Huang et al. "Real-time neural style transfer for videos". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 783–791.

[14] Phillip Isola et al. "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[15] Harry H. Jiang et al. "AI Art and its Impact on Artists". In: *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*. AIES '23. , Montréal, QC, Canada, Association for Computing Machinery, 2023, pp. 363–374. ISBN: 9798400702310. DOI: 10.1145/3600211.3604681. URL: https://doi.org/10.1145/3600211.3604681.

[16] Yijun Li et al. "Few-shot image generation with elastic weight consolidation". In: *arXiv preprint arXiv:2012.02780* (2020).

[17] Yi-Sheng Liao and Chun-Rong Huang. "Semantic context-aware image style transfer". In: *IEEE Transactions on Image Processing* 31 (2022), pp. 1911–1923.

[18] Andreia Machado Oliveira. "Future Imaginings in Art and Artificial Intelligence". In: *Journal of Aesthetics and Phenomenology* 9.2 (2022), pp. 209–225. DOI: 10.1080/20539320.2022.2150467. eprint: https://doi.org/10.1080/20539320.2022.2150467. URL: https://doi.org/10.1080/20539320.2022.2150467.

[19] Aditya Ramesh et al. “Hierarchical text-conditional image generation with clip latents”. In: *arXiv preprint arXiv:2204.06125* 1.2 (2022), p. 3.

[20] Robin Rombach et al. “High-Resolution Image Synthesis With Latent Diffusion Models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10684–10695.

[21] Robin Rombach et al. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[22] Peter Schaldenbrand and Jean Oh. “Content Masked Loss: Human-Like Brush Stroke Planning in a Reinforcement Learning Painting Agent”. In: *Proc. AAAI*. 2021.

[23] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. “A survey of transfer learning”. In: *Journal of Big data* 3.1 (2016), pp. 1–40.

[24] Ling Yang et al. “Diffusion models: A comprehensive survey of methods and applications”. In: *arXiv preprint arXiv:2209.00796* (2022).

[25] Shuai Yang et al. “Context-aware text-based binary image stylization and synthesis”. In: *IEEE Transactions on Image Processing* 28.2 (2018), pp. 952–964.

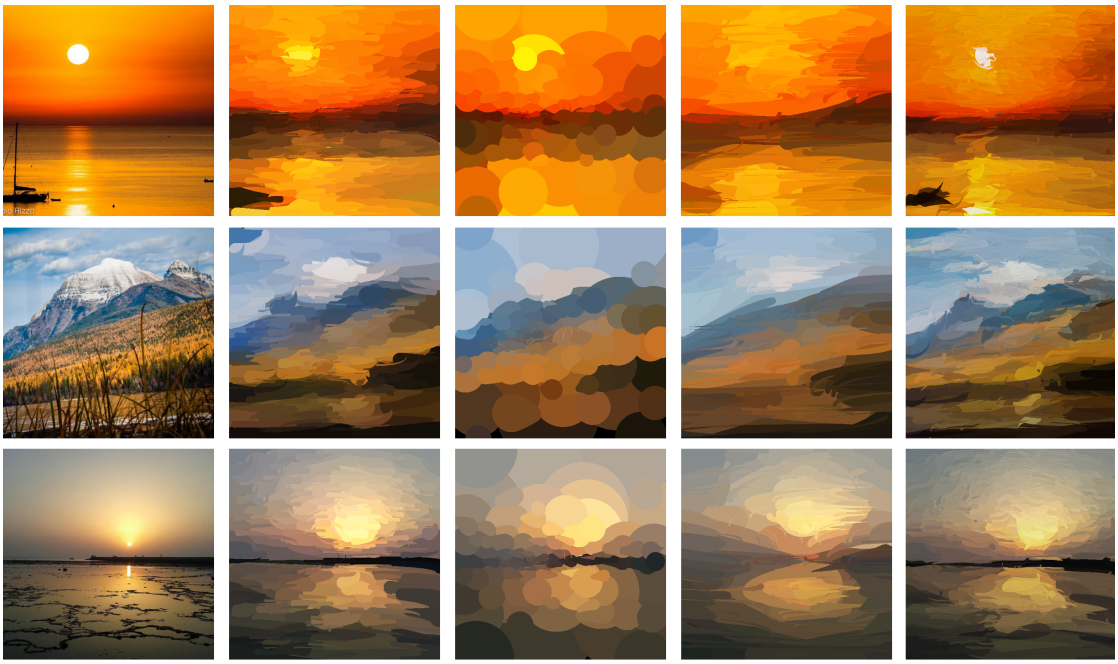
[26] Jiahui Yu et al. “Generative Image Inpainting With Contextual Attention”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[27] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. “Adding conditional control to text-to-image diffusion models”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 3836–3847.

[28] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. 2017.

Appendix A

Im2Painting: Supplementary Material



(a) Original (b) Abstraction - smooth (c) Style - thickness and length (d) Style - rapid motions (e) Style - oil-painting

Figure A.1: 512x512 300-stroke economical paintings. These examples illustrate results with higher resolution, and show how our method can provide nuanced, interpretable control over styles through varying losses and other interpretable factors. (b) Shows a basic style by smooth visual abstraction. (c) Adds constraints on thickness and length that reduces the number of small strokes. (d) adds random noise to strokes resulting in a scratchy style. (e) Oil-painting style.

A.1 Higher Resolution Paintings

We show 512x512 visual abstraction and styles landscapes paintings with 300 strokes (Figure A.1) by using the set of strokes parameters output by the network and rendering with a non-differentiable renderer at a higher resolution. Figure A.1 (b) show smooth abstraction using pixel loss at all itermediate canvases. (c) Shows artistic style with constraints on thickness and length. (d) Show artistic style given by rapid motions. Figure A.1 shows that our method provides a very fine-grained and interpretable control with the potential to expand the styles more, as opposed to previous work that have no control over nuances and details of their paintings.

A.2 Ablations on Model Architecture

This section shows ablations on both presented networks: visual abstraction net, based on FC layers in decoder, and precise and style net based on FC and LSTM layers in decoder. For simplicity, we name the former AbstNet (which uses D_{FC}) and the latter Precise-Style-Net (which uses D_{LSTM}). Note that Precise-Style-Net is used for precise reconstructions as well as artistic styles, while AbstNet is only used for visual abstractions. We first ablate on network sizes and encoder-decoder combinations for our AbstNet.

The left part of Figure A.2 (a-c) shows the full architecture of three abstraction networks, ordered with increasing number of parameters, from left to right. Figure A.2 (d) shows the full architecture of our Precise-Style-Net, which has $\sim$90M learnable parameters. We find a relation between number of learnable parameters and abstraction quality. Smaller networks with $\sim$30M and $\sim$50M parameters (Figure A.2 (a), (b), respectively) are able to produce higher quality visual abstractions. We find that increasing the spatial size of the encoded feature map (a) doesn't affect the output painting significantly. However, reducing the number of parameters of the network increases the abstraction capability, while increasing the number of parameters Figure A.2 (c) ($\sim$100M) results

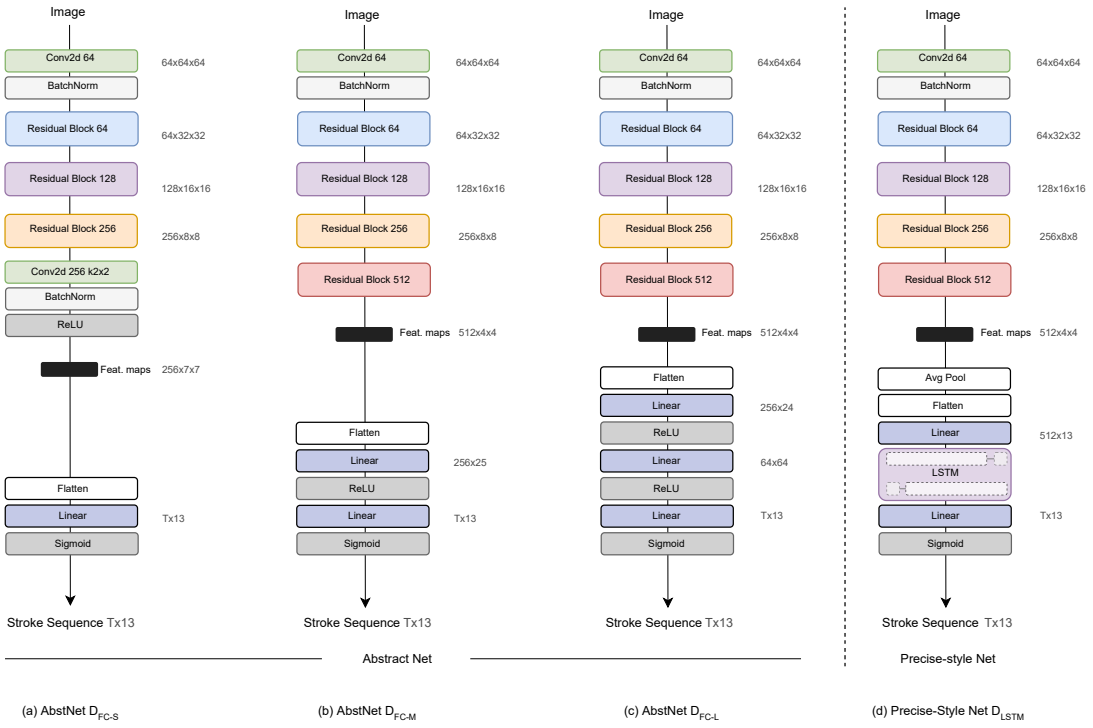


Figure A.2: Model Ablations. Ablation on network architectures for visual abstractions for 300 strokes (a-c) and precise-style network (d). (a) Encoder uses the first 3 residual blocks of Resnet-18 and an additional convolutional layer to extract a larger feature map. Decoder is composed by 1 linear layer followed by non-linear activation. Learnable parameters: ∼31M. **(b) Abstract net used in this chapter**. Encoder uses the first 4 residual blocks of Resnet-18. Decoder has 2 linear layers followed by non-linear activations. Learnable parameters: ∼50M. (c) Encoder is the same as (b). Decoder is composed by 3 linear layers followed by non-linear activations. Learnable parameters: ∼100M. (d) Precise-Style network, used for precise reconstructions and styles. Decoder is composed by a linear layer, followed by a bidirectional LSTM, followed by a final linear layer. Learnable parameters: ∼90M.

in better approximations to the input image and thus, reducing the level of abstraction. The result of these ablations are reflected in the following sections.

A.2.1 Visual Abstraction

Figure A.3 shows model ablations on the coarsest visual abstraction using objective $\mathcal{L} = ||I - C_T||_1$, where C_T is the canvas at the last time step. All paintings are made

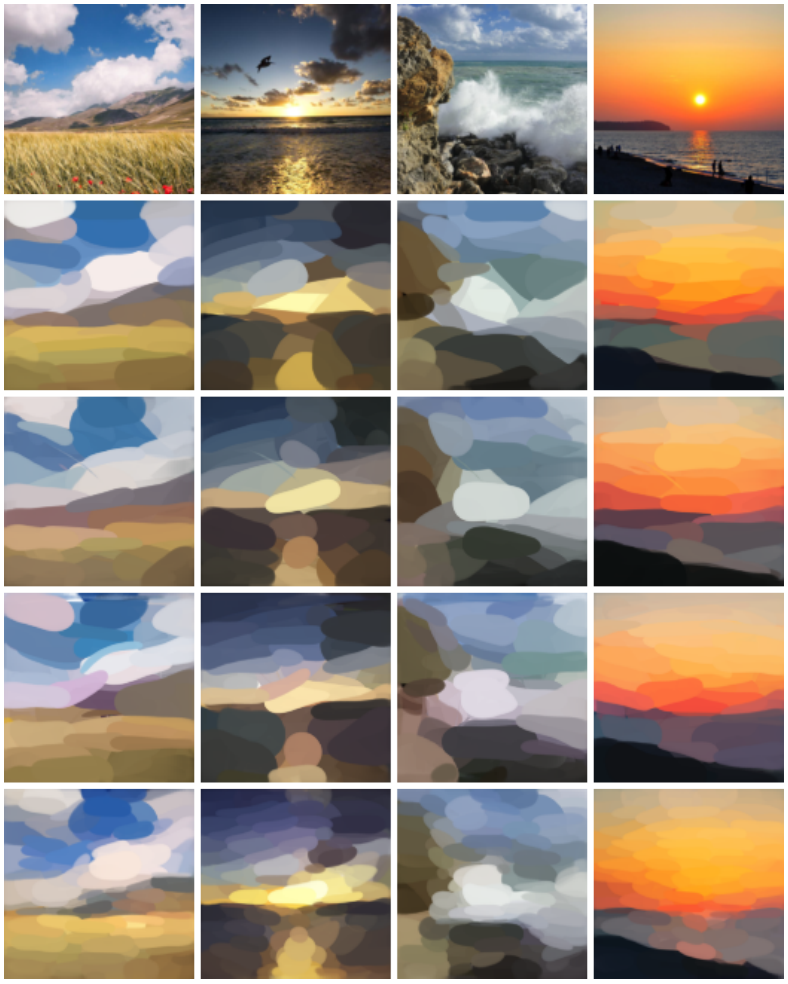


Figure A.3: Model ablations on visual abstraction. All visual abstraction paintings are made with 300 strokes. Top row: original image. Second row: AbstNet using architecture shown in Figure A.2 (a). Third row: AbstNet using architecture shown in Figure A.2 (b). Fourth row: AbstNet using architecture shown in Figure A.2 (c). Bottom row: Precise-Style-Net (Figure A.2 (d)). The goal of this method is to provide a coarse level of abstraction, and we find a relation between network size and level of abstraction. We see how second and third rows are better abstractions than fourth and last rows. There is not a substantial difference between second and third rows, and both are suitable for this abstracted style. For all visual abstractions shown in the paper, we choose to work with the network corresponding to the third row (A.2 (c)).

with 300 strokes. Except for the bottom row, which uses Precise-Style-Net (Figure A.2 (d)), the other rows use different versions of AbstNet, specifically, second row uses Figure A.2 (a), third row uses Figure A.2 (b), and fourth row uses A.2 (c). Smaller networks

produce better (coarser) abstractions than bigger networks, which is desirable for this objective function. We find that the architecture shown in Figure A.2 (b) produces better results for the range of visual abstractions that we present in this work.

A.2.2 Visual Abstraction Painting Progress

Figure A.4 shows canvases rendered every 25 strokes. Top row shows visual abstraction given by applying L_1 loss at the final frame. This corresponds to a one-off optimization schema, which encourages a non-greedy behaviour. As a result, the network wastes the majority of the strokes, achieving a coarse abstraction with the last 50 strokes. Abstraction in middle row is achieved using $\sum_{t=1}^{T/k} ||I - C_{kt}||_1$, where T/k is an integer, $k = 6$, and it corresponds to a sparse optimization schema, backpropagating errors every 50 strokes. Here only the first few strokes are wasted. The bottom row corresponds to a more guided regime which backpropagates the error at every time step. This results in a greedy behaviour and no strokes are wasted.

Figure A.4: Visual abstraction painting progress of different optimization schemas. All three rows use 300 strokes; intermediate canvases are displayed every 25 strokes. Top row: applying the loss only on the last canvas. The network "wastes" the majority of the strokes, with few of them contributing much to the final painting. Middle row: applying the loss only on every 50th canvas. Bottom row: applying the loss at every canvas, as in Equation 3 in main paper.

A.2.3 Artistic Styles

Figure A.5 shows model ablations on artistic style given by constraining both stroke thickness and length. All paintings are made with 300 strokes. Figure A.5 (b) shows

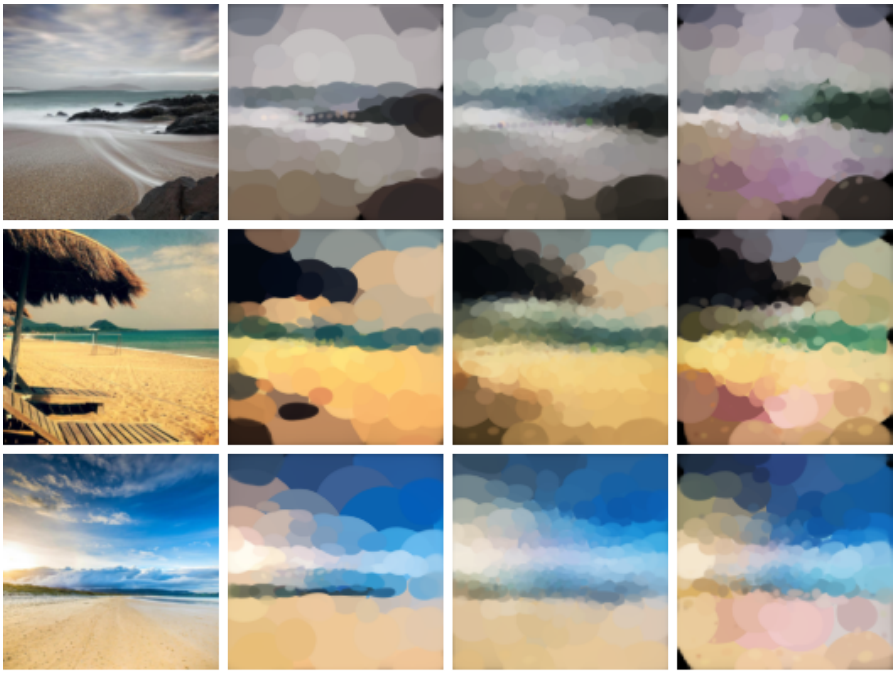


Figure A.5: Model ablations on styles. All paintings are made with 300 strokes. (a) Original. (b) Style produced by our Precise-Style-Net. (c) AbstNet with a similar number of trainable parameters than (b). (d) AbstNet with approximately half of the total number of trainable parameters than the previous two. (b) shows a clear style, defining object boundaries much better than our abstract net, which outputs blurrier paintings.

the results of this style using our Precise-Style network. This network produces clearer results, defining object boundaries better than columns (c) and (d). Even though column (b) and (c) use networks with a similar number of parameters, the LSTM layer seems provide a better structured sequence of strokes. This behaviour happens across all artistic styles and for precise reconstructions.

A.2.4 Precise Representations on Portraits

Figure A.6 demonstrates the improvement that Precise-Style net makes over AbstNet on precise reconstructions on CelebA. We compare our Precise-Style LSTM-based net-

Figure A.6: Model Ablations on Reconstructions. Comparison of StyleGAN-generated face reconstructions using 300 strokes across three architectures: (b) AbstNet with one linear layer (Figure A.2 (a)), (c) AbstNet with two linear layers (Figure A.2 (c)), and (d) Precise-Style-Net with decoder D_{LSTM}. (b) shows coarse strokes with less defined facial features; (c) improves on (b) but lacks in finer details; (d) offers sharper reconstruction with better detail and edge capture.

work with two abstract FC-based networks, one with approximately the same number of parameters than our Precise-Style net ($\sim$100M parameters, Figure A.2 (c)), and a smaller network with $\sim$30M parameters (Figure A.2 (a)). We see how the precise net outputs sharper paintings (Figure A.6 (d)) because it is able to capture finer details, such as main facial wrinkles, and expressions. This is perhaps more noticeable in rows 3 and 4. The paintings given by the smallest network, shown in Figure A.6 (b), are able to maintain the overall structure and composition, however, produce coarser paintings.

Figure A.6 (c) improves upon (b) but fails to accurately eyes and and facial gestures. Metrics corresponding to this study are reported in Table A.1. Even though the larger AbstNet gives a slightly better perceptual score, overall the image is better reconstructed with Precise-Style net.

Figure A.7 compares our method with RL [4] and Transformers [5]. Our results are shown in the second row. The third and fourth row show the RL approach with 300 and full model, respectively. The last row shows PaintTransformer with 300 strokes. Learning-to-paint [4] achieves better reconstructions with hundreds of more strokes. Our model produces better defined and higher precision paintings than previous methods with 300 strokes.

Method	$\mathcal{L}_1 \downarrow$	$\mathcal{L}_{perc} \uparrow$	Num Strokes
AbstNet D_{FC-S}	0.038	0.738	300
AbstNet D_{FC-L}	0.043	**0.750**	300
Precise-Style-Net D_{LSTM}	**0.035**	0.747	300

Table A.1: Quantitative results on same held-out set on CelebA as in Table 1 in main paper, using networks explained in A.6. Precise-Style network achieves a much lower L_1 score than AbstNet. The latter, however, achieves a slightly better perceptual score. However, as shown in Figure A.6, it produces slightly worse reconstructions.

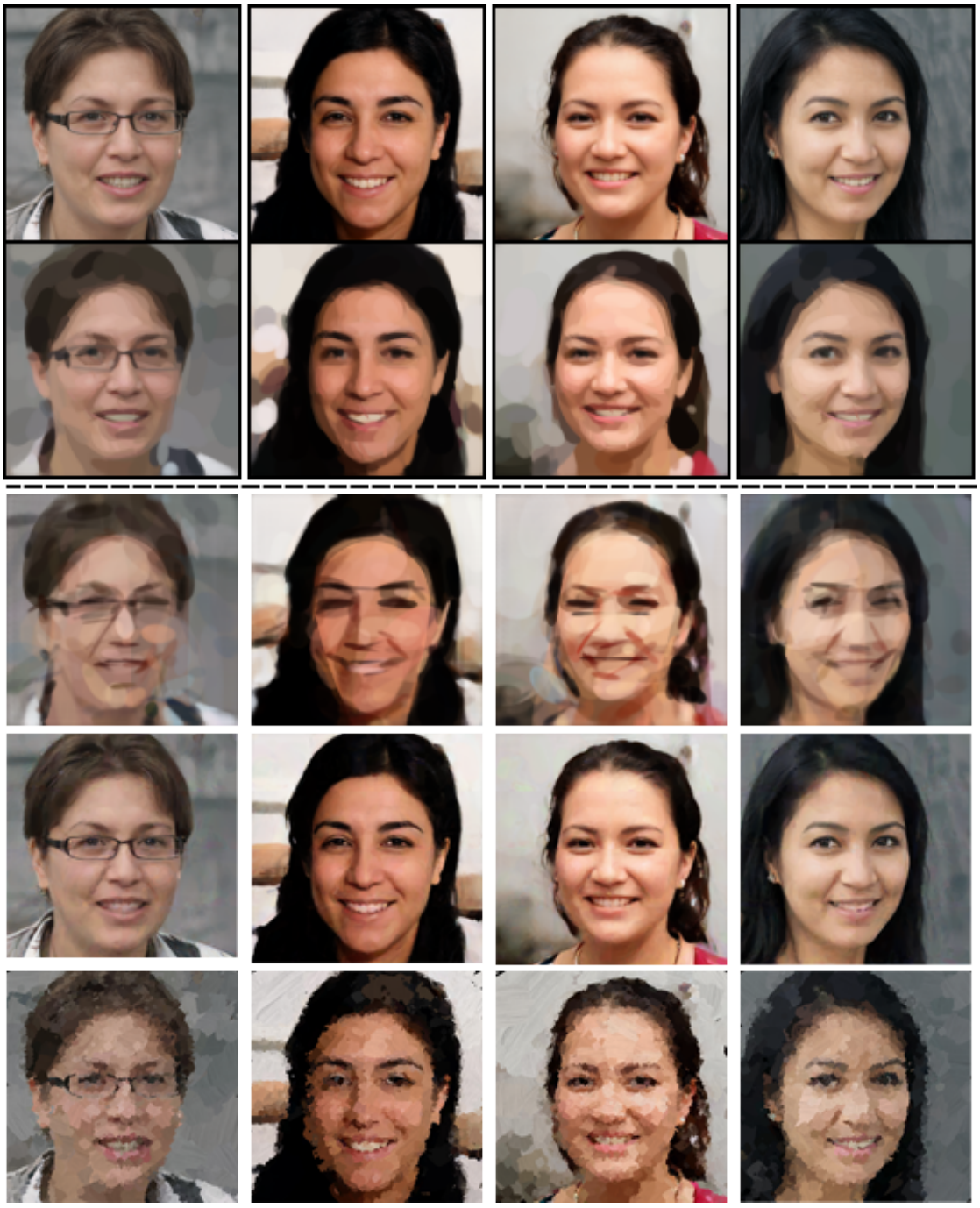


Figure A.7: Comparison with State-of-the-Art-Methods. Top row: Original. Second row: Our results using D_{LSTM} Third and fourth rows: Learning-to-Paint [4] limited to 300 strokes and full model, respectively. Bottom row: PaintTransformers [5] limited to 900 strokes.

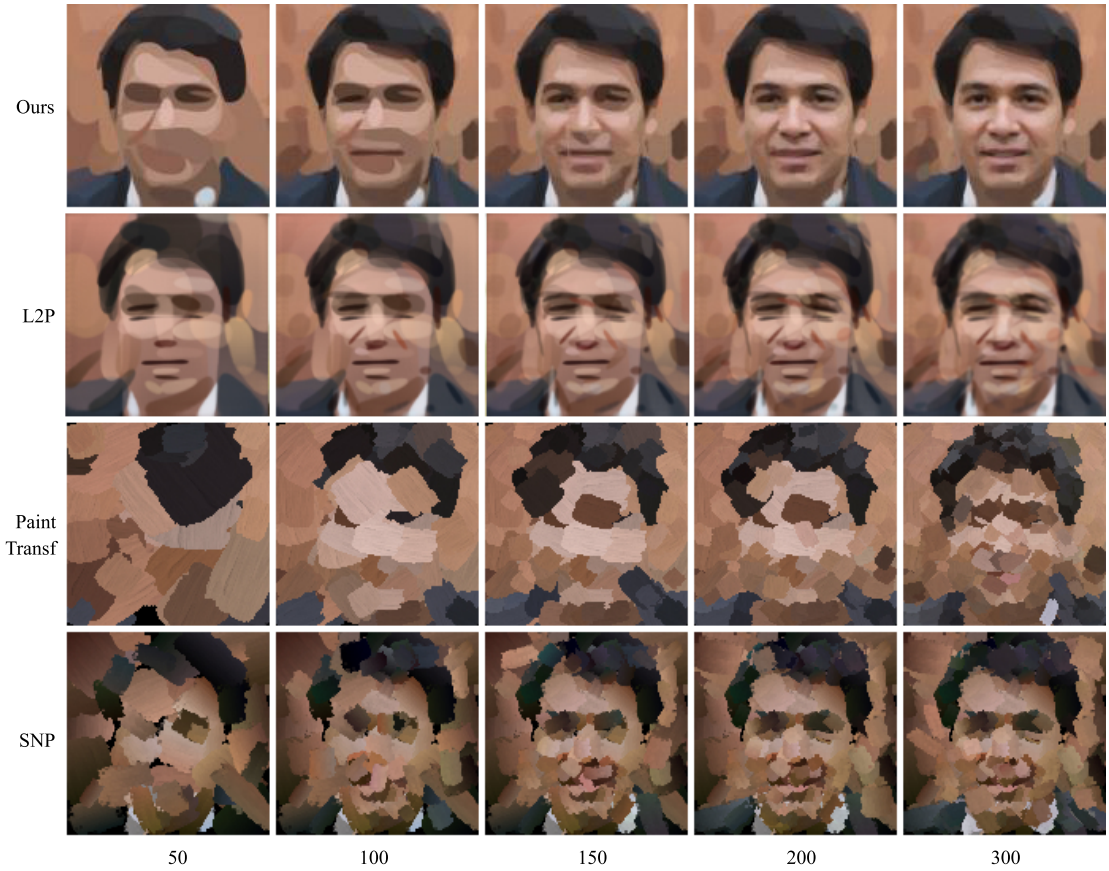


Figure A.8: Painting progress of different baselines. Our method places the first 300 hundred strokes more efficiently than previous methods, and captures facial features with fewer strokes. We see this as a measure of economy, which enables stylistic variations.

Figure A.8 shows a comparison of the painting progress over the first 300 strokes. As discussed in the main paper, our algorithm is more economic, as it needs fewer brushstrokes to paint facial features with more accuracy. Although L2P also achieves very good reconstructions, it needs more paint to do so. In this example, L2P produces more artifacts at the beginning, and the algorithm needs more strokes to correct the face. Paint Transformer and SNP both need over a thousand of strokes to reconstruct the input, as they spray multiple strokes and paint by aggregation.

A.2.5 Comparison with SOTA on Landscapes

We provide further comparison with previous work on landscapes. Table 3.2 in main chapter shows quantitative results using 300 strokes (left), and 1000 or more strokes (right). Metrics are calculated using 100 samples from a held-out set. We use Precise-Style network, which uses LSTM in the decoder, D_{LSTM}. Our network with a fine-tuning step outperforms previous methods. Figure A.9 shows a comparison of our networks with previous work. Our network without fine-tuning step seems to maintain a more consistent use of colors, and the paintings are slightly less noisy than the RL method by Huang *et al.* [4] (Figure A.9 (d)). PaintTransformers [5] (Figure A.9 (e)) and Stylized Neural Painters [7] (A.9 (f)) share the same stroke model, resulting in a coarser representations and needing many more strokes to produce accurate representations of the input image.

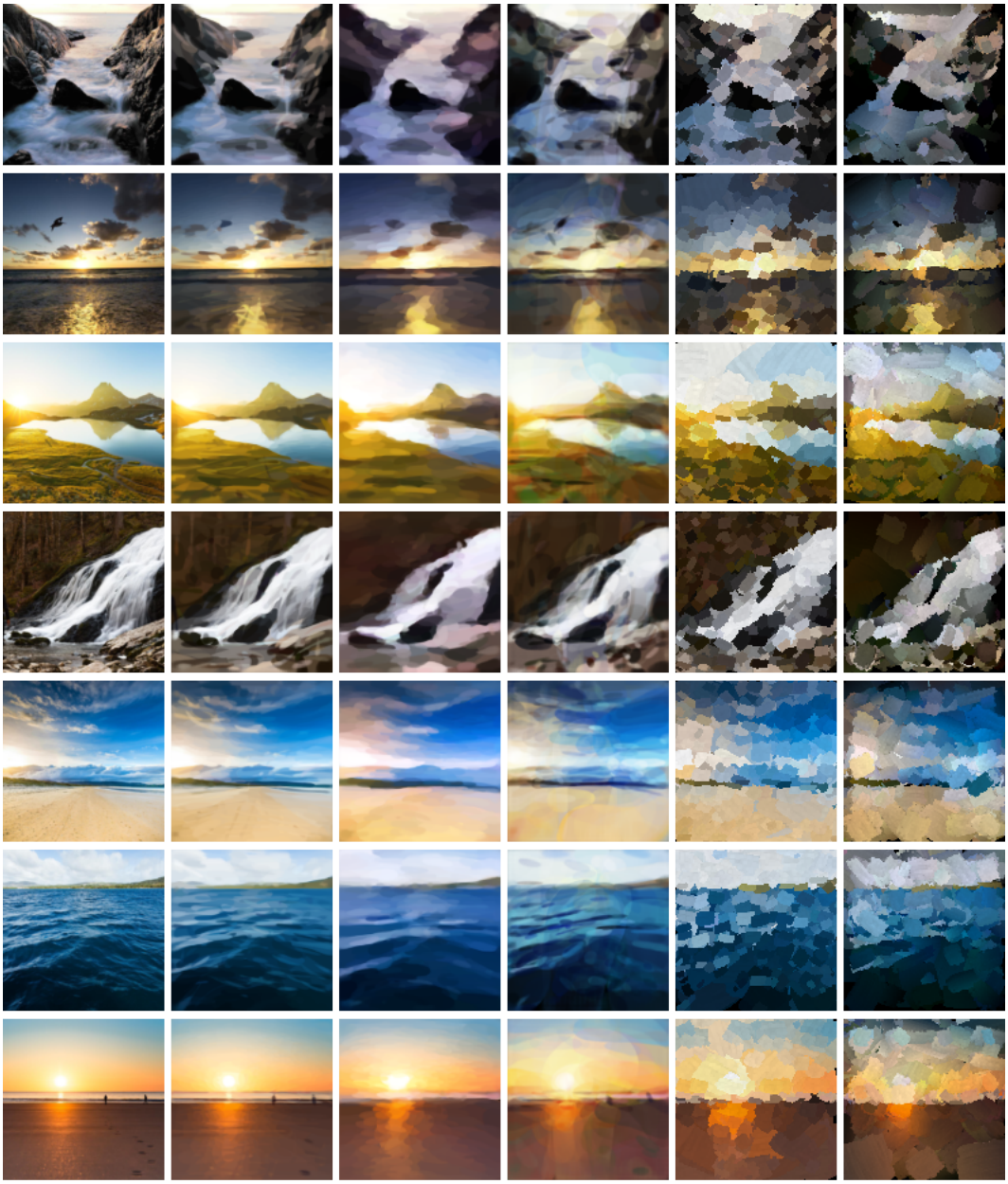


Figure A.9: Precise reconstructions comparison with SOTA. All methods use 300 strokes. Our method with a fine-tuning step (a) is able to reconstruct the input image better. Without fine-tuning step (b) our method still outputs smoother paintings than previous work. This is perhaps more visible in second, third, and last rows. Paint Transformers [5] and SNP [7] output very large strokes and they suffer with reduced stroke budgets.

A.2.6 Stroke Budget Ablations

This section shows results a comparison of our precise network with 300, 600 and 1000 strokes in Figure A.10 columns (b), (c) and (d), respectively. Increasing the stroke budget while maintaining the resolution results in blurrier paintings. We report the metrics for the 300-stroke and 1000-stroke models in Table 3.2 in main chapter. Metrics show that the 1000-stroke outperforms the 300-stroke model, however, the quality of the 1000-stroke painting is much worse than the 300-stroke painting, which indicates that other metrics might be more suitable for painting algorithms.

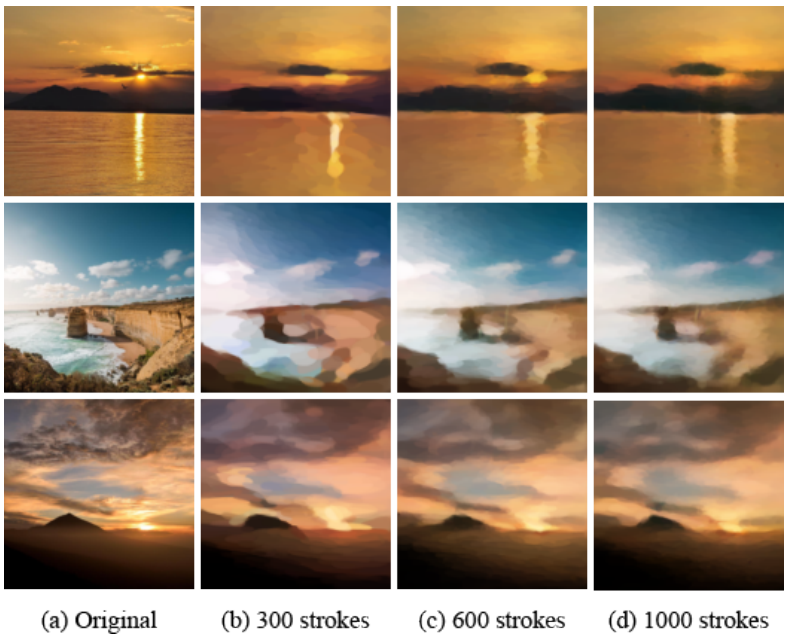


Figure A.10: Stroke budget ablations. Precise-Style-Net using (b) 300 strokes, (c) 600 strokes and (d) 1000 strokes. Adding more strokes produces a blur effect on the painting.

A.3 Loss Functions

Different loss functions in combination with different decoders, D_{FC} and D_{LSTM} give different levels of control of the painting, as shown in previous sections. In this section we show two the effect of two other loss functions.

A.3.1 WGAN-GP Loss

Following [6, 1, 4, 5], we also explore the impact of an adversarial loss for precise paintings. Specifically, we add the same discriminator network, D_{net}, as [4], and use WGAN-GP [2] in our energy function. D_{net} discriminates between real pairs (input image, input image) Q, and fake pairs (input image, painting) P. WGAN-GP loss is:

$$\begin{aligned}\mathcal{L}_{WGAN-GP} = \mathbb{E}_{P\sim\mathbb{P}}[D_{net}(P)] - \mathbb{E}_{Q\sim\mathcal{Q}}[D_{net}(Q)] \\ + \lambda\mathbb{E}_{\hat{Y}\sim\hat{\mathcal{Y}}}[(||\nabla_{\hat{Y}} D_{net}(\hat{Y})||_2 - 1)^2]\end{aligned} \tag{A.1}$$

where P is the pair canvas-input image at the last time step, Q is the pair input image-input image, and $\hat{Y}$ is sampled from Q and P with t uniformity sampled between 0 and 1, $\hat{Y} = tQ + (1-t)P$ with $0 \leq t \leq 1$. We substitute perceptual by WGAN-GP loss in our general formulation, and thus, our energy function becomes:

$$\Phi_{precise_{WGAN-GP}} = \lambda_1 \sum_{t=1}^{T} \mathcal{L}_{pixel_t} - \lambda_2 \mathcal{L}_{WGAN-GP} \tag{A.2}$$

We set $\lambda_2 = 0.01$. We find that adopting an adversarial schema does not improve perceptual losses, as shown in Figure A.11 (c). In some cases, it introduces some artifacts that make the painting a bit noisier, as seen in Figure A.11 bottom row. We hypothesize that this is partially due to the gap between distributions of real images and distributions of 300-stroke paintings.

A.3.2 Contrastive Loss

A contrastive approach might seem suitable for precise paintings. Same as in the previous section, we remove perceptual loss and keep pixel loss. The goal of this loss is to minimize the cosine distance of a vector representation of the input image $\vec{v}$ with a vector representation of the canvas C_T $\vec{w}$, while maximizing the distance between C_T

$\vec{w}$ and a set of negative samples $Z = \{\vec{z_1}, \vec{z_2}, ..., \vec{z_k}\}$. Contrastive loss is calculated as :

$$\mathcal{L}_{contrastive} = -log\frac{exp(dist(\vec{w}, \vec{v})/\tau)}{\sum_{k=1}^{K} exp(dist(\vec{w}, \vec{z_k})/\tau)} \tag{A.3}$$

We set $K = 10$ and randomly sample K images from the training set where $k \neq i$, and i is the index of the vector representation $\vec{v}$ corresponding to the positive sample. Our energy function becomes:

$$\Phi_{precise_{contrastive}} = \lambda_1 \sum_{t=1}^{T} \mathcal{L}_{pixel_t} - \lambda_2 \mathcal{L}_{contrastive} \tag{A.4}$$

We set $\lambda_2 = 0.0001$ for our ablation. Figure A.11 (d) shows the paintings obtained using Equation (A.4). As we see, this loss does not improve results using perceptual loss, and results are inconclusive. We hypothesize that a more robust contrastive learning approach with more data augmentations or robust architectures as done in [3].

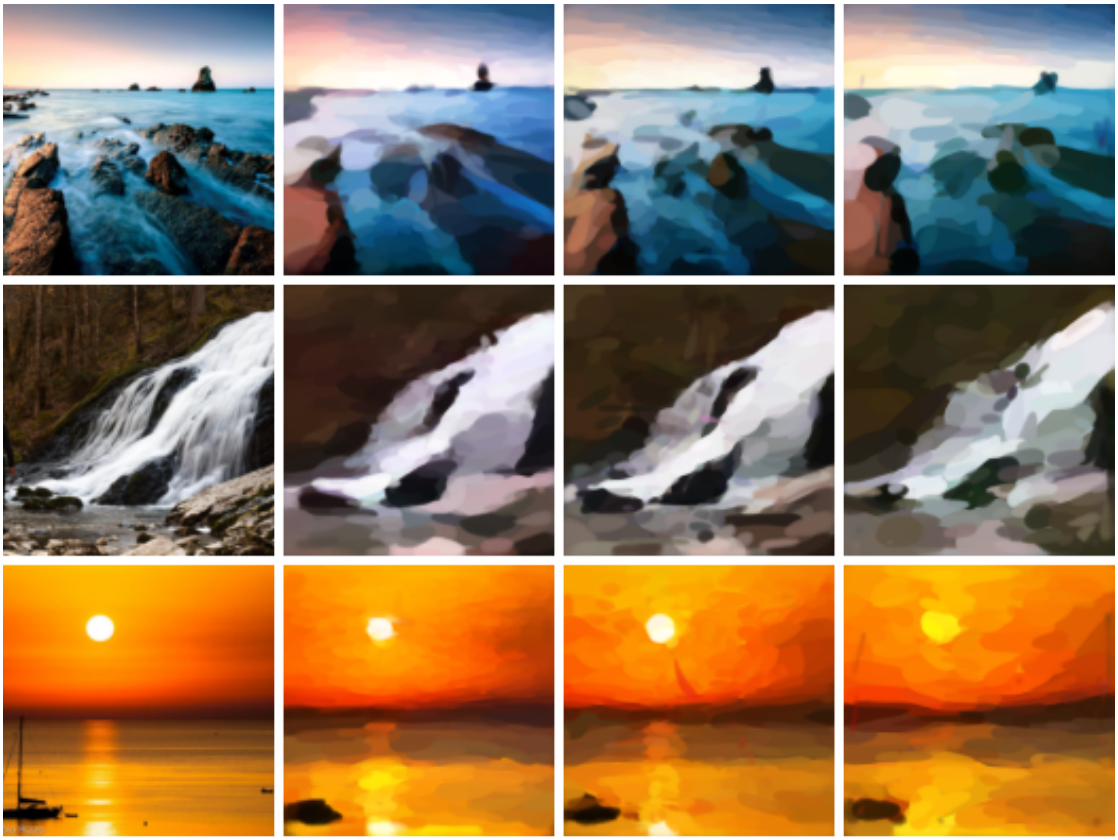


Figure A.11: Effect of using WGAN-GP and contrastive losses. We substitute perceptual loss by WGAN-GP (c) and contrastive loss (d) following Equation (A.3).

A.4 Oil Painting Comparison

We use the thin plate spline (TPS) algorithm to compute the mapping of a bitmap of an oil painting texture to the shape of our strokes. To obtain the control points that are fed into the TPS algorithm, we first divide the original Bézier curve into N parts and offset those points by their corresponding width to both sides. We then map an oil painting bitmap texture to the original shape of the stroke. Figure A.12 shows a comparison between our approach (top) and the oil painting filter on Adobe Photoshop (bottom).

Figure A.12: Comparison of oil painting texture via thin plate spline algorithm (top) and oil painting filter in Adobe Photoshop (bottom)

We see that the latter smooths the curves slightly more than the TPS algorithm. The former maintains the original thinner strokes with more fidelity than the oil painting filter on Photoshop.

References

[1] Yaroslav Ganin et al. “Synthesizing programs for images using reinforced adversarial learning”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1666–1675.

[2] Ishaan Gulrajani et al. “Improved training of wasserstein gans”. In: *arXiv preprint arXiv:1704.00028* (2017).

[3] Kaiming He et al. “Momentum Contrast for Unsupervised Visual Representation Learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[4] Zhewei Huang, Wen Heng, and Shuchang Zhou. “Learning to Paint With Model-based Deep Reinforcement Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[5] Songhua Liu et al. “Paint Transformer: Feed Forward Neural Painting With Stroke Prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 6598–6607.

[6] John F. J. Mellor et al. *Unsupervised Doodling and Painting with Improved SPIRAL*. arXiv:1910.01007. 2019.

[7] Zhengxia Zou et al. “Stylized Neural Painting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 15689–15698.

Appendix B

Segmentation-Based Parametric Painting: Supplementary Material

(a) Expressionist (b) Painterly (c) Abstract

Figure B.1: Styles: (a) Expressionist, consisting of thick, textured, loose and low fidelity representation where subject matter is not a priority. (b) Painterly, think, shaped strokes with a loose touch, enough to balance the scene recognition and not yield a high-fidelity painting style, (c) Abstract, scattered, fewer strokes that capture the essence of the scene.

Paintings at Large Resolution

More paintings at large resolution are shown at the end of this document.

B.1 Optimization Method

This section provides further implementation details of our method. The pseudocode is shown in Algorithm 3. We establish the nomenclature as follows:

- Input image X, canvas C
- Number of painting passes P (int), from 1 to P
- Number of painting layers K (int), from 0 to $K-1$
- Number of Dynamic Attention Maps (DAMs), V (int). If uniform, DAMs follow an array of 128×128 patches uniformly distributed on image X and canvas C. $V = (H//128 * W//128)$
- Number of stroke parameters, T (int), per DAM, different per p, and can be different per k
- Semantic Segmentation Network, W, determines K

B.1.1 Implementation Details

For all styles, we use Adam optimizer [3] with a learning rate of 0.0002 and betas 0.5 and 0.99. We optimize all painting layers for 300 iterations, and set the canvas background color to black. For our semantic segmentation network, we use the DETR model

	Realistic	Painterly	Abstract Expressionist	Abstract
Num. Passes (P)	4	4	1	2
Num. layers (K)	1	var	var	var
Num. Strokes (T)	[36, 49, 64, 81]	[9, 16, 16, 9]	[9]	[9, 16]
Num. DAMs (V)	$(H//128 * W//128)$	[25, 25, 50]	$(H//128 * W//128)$	[25, 25]
DAMs mode	U	U if $p = 1$ else S	U	S

Table B.1: Style hyperparameters by default. Users can fine-tune these parameters for further control over the desire styles.

[1], available at Hugging Face, which consists of a CNN (ResNet) backbone followed by an encoder-decoder Transformer, trained on the COCO Panoptic Segmentation task.

Algorithm 3 Main Painting Process

procedure PAINT($imagePath$, $style$)
    $painter \leftarrow$ INITIALIZEPAINTER($imagePath, style$)
    $Segment.LayersK \leftarrow W(imagePath)$
    for $p \leftarrow 1$ to P **do** ▷ Painting passes
        for $k \leftarrow 0$ to $K - 1$ **do** ▷ Painting layers
            $DAMs \leftarrow$ GETDAMS($mode, p, |V|$)
            $canvas_p \leftarrow$ GETCANVAS($DAMs, p$)
            $strokes \leftarrow$ INITSTROKES($DAMs, T, p$)
            $strokes \leftarrow$ OPTIMIZE($strokes, canvas, style, k$)
            $canvas_p \leftarrow$ UPDATECANVAS($canvas_p, strokes$)
            Append $canvas_p$ to $allCanvasesL$
        end for
    end for
    COMPOSEFINALPAINTING(canvas)
    return final painting
end procedure
procedure INITIALIZEPAINTER($imagePath$, $parameters$)
    Process input image and set up canvas
    Define renderer, segmentaiton network, and perceptual network
    return painter object
end procedure
procedure GETDAMS($mode$, p, V)
    Computes DAMs based on mode and max number V
    return image X crops and DAMs coordinates
end procedure
procedure GETCANVAS($DAMs$, p)
    Crops $canvas_p$ based on DAMs
    return canvas crops
end procedure
procedure INITSTROKES($DAMs$, T ,p)
    Initializes random stroke parameters based on DAMS, T and p
    return Stroke parameters
end procedure
procedure OPTIMIZE($strokes$, $canvas$, $style$, k)
    Optimizes strokes based on style parameteres
    Perform rendering to compute loss
    return strokes
end procedure
procedure COMPOSEFINALPAINTING($canvas$)
    Merge layers and apply final adjustments to canvas
    Save final painting
end procedure

B.1.2 Style Parameters

Our method takes in an image X and a style input as a string between "realistic", "painterly", "abstract", and "expressionist". Note that our optimization method does not apply transfer style techniques. Rather, we adjust optimization hyper-parameters such as the amount of dynamic attention maps (DAMs), stroke budget, or painting passes, to achieve different style variations. We found the combination of such parameters empirically, and we keep the parameters shown at Table B.1 as default. However, users can change it to fine-tune the paintings to further adapt the painting. All styles follow a coarse-to-fine strategy, halving the stroke thickness following the formula: $a_p = 2^{1-p}a_1$, where $a_1 = 0.8 * 128$, that is, the 80% of the length of a 128×128 patch. The set of segmentation layers $\{k\}_0^{K-1}$ is generated by a segmentation network W, such that $\{k\} = W(X)$.

Realistic Style Our realistic style is generated by setting $P = 4$, $K = 1$, setting DAMs to uniform, and thus $V = (H//128 * W//128)$. The stroke budget T on each patch follows $s_p = (p + 5)^2$.

Painterly Style The painterly style is generated by setting $P = 4$, applying a first foundational painting pass consisting of uniform DAMs ($V = (H//128 * W//128)$) and no segmentation distinction ($K_p = 1$ if $p = 1$). For $p > 1$, we activate segmentation layers, that is, $K_p = |W(X)|$, where X is the input image and $|\cdot|$ is the number of the segmentation layers; and switch to selective DAMs. The stroke budget T on each patch is limited to 16 except for the first and last passes which is 9.

Abstract Styles Within the abstract style, our method can yield different degrees of abstraction. For simplicity, we grouped them together in Chapter 5. However, there are two fundamental differences in the way they work. Figure B.1 shows the two abstract styles. The expressionist style (a) is captured by the use of loose, efficiently optimized thick strokes with heavy texture. This produces a painting that prioritizes the effect and

ambient of the scene by not caring too much about accurate shapes. The abstract style (c) is a looser representation than the painterly style, yet capturing shapes and subject matter better than the expressionistic abstract style.

Given a segmentation network W, the expressionist style is generated by setting DAMs to uniform, stroke budget $T = 9$, $P = 1$, $K = |W(X)|$, where X is the input image and $|\cdot|$ is the number of the segmentation layers. The abstract style is achieved by setting DAMs to selective, $P = 2$, stroke budget $T = [9, 16]$, $V = [25, 25]$, and $K = |W(X)|$.

B.2 Comparison with SOTA

We show more stroke distributions in Figure B.2 across different subject matters. SNP and L2P present a more uniform distribution of strokes, barely differentiating between semantic areas or objects in the scene. While L2P presents a clear uniform distribution across the entire canvas, SNP is able to scatterly capture some regions of interest, but still suffer to bias the painting towards the main objects in the scene. PT does the absolute opposite, that is, it fixates a disproportionate amount of strokes to a specific area, which does not match semantic regions, while leaving the rest of the canvas with very few and big strokes. This is clearly visible in the picture at the middle row. The distribution is concentrated in a blob spanning the building and the pavement equally. The same event happens in the last row where the lake and the lavender flowers that are closer to the water receive a higher quantity of strokes.

Ours, however, sits in between L2P / SNP and PT. While focusing on a specific object in the picture, with semantic coherence, it is able to also distribute strokes on other important areas of the picture. For instance, the bottom row shows how flowers, lake and mountains are better captured, while leaving sky with fewer strokes. In the case of the building, strokes are concentrated around it, since this is clearly the most important semantic part of the image. However, our method is able to redirect its stroke distribution according to the user's desire as shown in Figure B.3. In this figure, we

show a painting dissection based on our semantic segmentation pipeline. Except for last row, each column shows emphasis on a different k-th layer. The second row shows a distribution of all the strokes in the painting, while the fifth shows a distribution of only the strokes pertaining to each k-th layer.

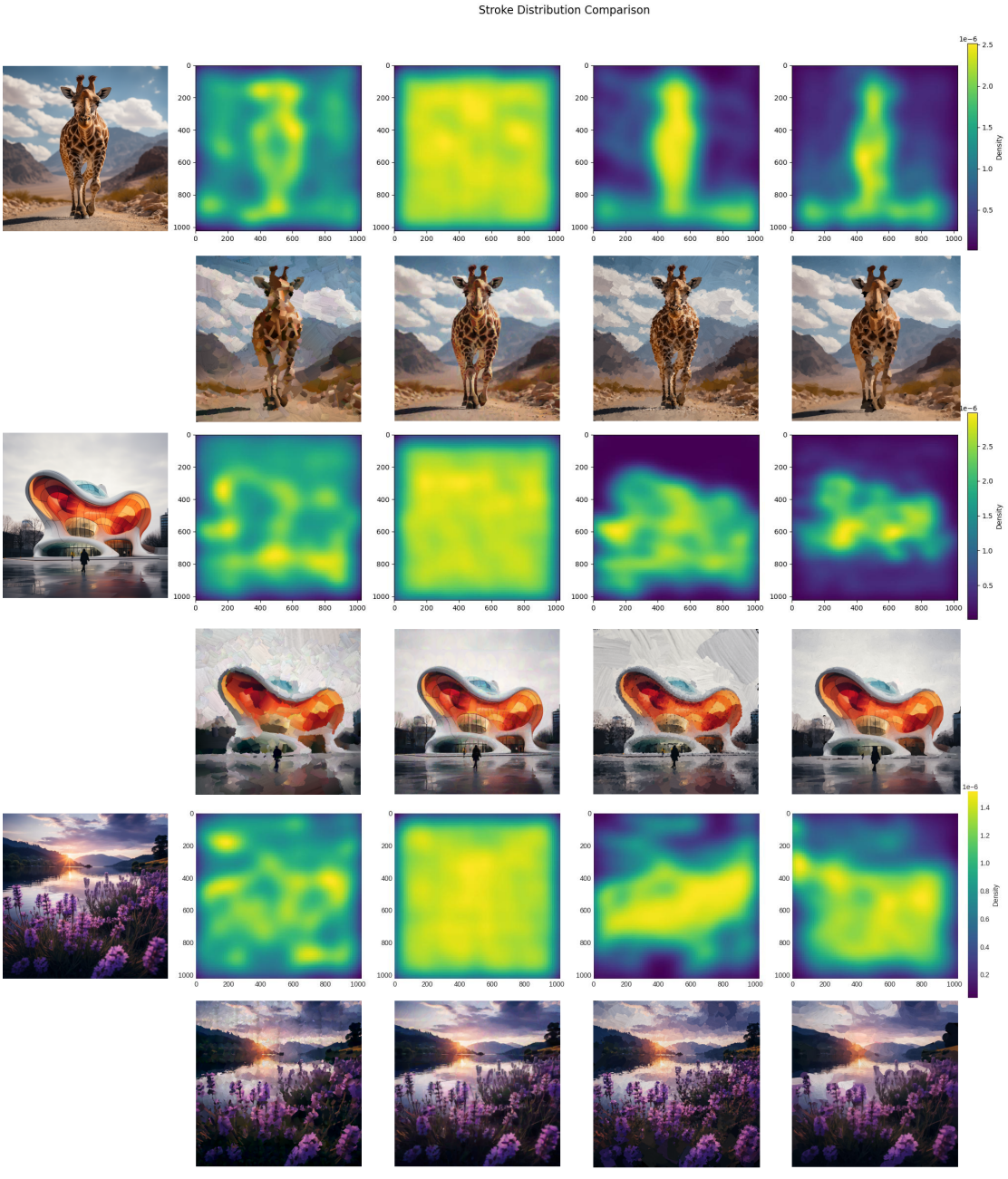
B.3 User Study

The user study is divided in two sections with 6 pairs of images each (6 comparisons side by side). The first section evaluates realistic paintings, and asks which painting is a better painting of an input image. The second section evaluates visual appeal, and asks participants to choose the painting that looks more natural and less computer made. In total, we have 12 two-way side by side comparisons. The participants see first an input photograph, and below the photograph we place a pair of paintings. We ask to choose left or right painting. We compare our paintings with three recent methods: an optimization-based method "Stylized Neural Painting" (SNP) [5], a learning-based method with Transformer "Paint Transformer" (PT) [4], and a RL method "Learning to Paint" (L2P) [2]. We randomize the order in which we show the pairings.

The complete user study is shown in Figures B.10 to B.12. All four pairs from figure Figure B.10 and the first two rows in Figure B.11 correspond to the first section of the user study. These pairings correspond to the left side of Table 5.2 in Chapter 5. IDs are Figure B.10 top row: A5, second row: A3, third row: A4, fourth row: A6. In Figure B.11, top row: A1, second row: A2. The last two rows of Figure B.11 and Figure B.12 belong to the second section of the study. They have the following IDs; Figure B.11 third row: B1, fourth row: B5. Figure B.12, from top to bottom: B3, B6, B2 and B4, respectively.

The survey was conducted using Google Forms and distributed primarily within the School of Architecture at Carnegie Mellon University. Participants spanned a diverse range of academic positions, including undergraduates, Master's and Ph.D. students, and faculty members. In our approach to maintaining participant confidentiality, we

opted not to collect email addresses, which stops us from providing demographic details of the respondents. A total of 30 individuals participated in the study, and no monetary or other forms of compensation were offered for their involvement. The survey, estimated to take less than 10 minutes on average, required each participant to evaluate a series of 12 side-by-side comparisons. To clarify, these comparisons were split into two distinct sections: the first section aimed to assess the realism of the paintings, while the second section focused on evaluating the visual appeal, with an emphasis on identifying the painting that appeared more natural or handcrafted. This two-pronged approach allowed for a comprehensive assessment of both the technical and aesthetic qualities of the generated paintings.



(a) Input (b) SNP (c) L2P (d) PT (e) Ours

Figure B.2: Stroke distribution analysis: SNP and L2P present a uniform distribution across the canvas. L2p is completely scene-agnostic, and SNP cannot distinguish between semantic areas. PT presents a severe contrast between objects and background, and struggles to differentiate between semantic areas of the image. Ours distribute strokes based on semantic areas.

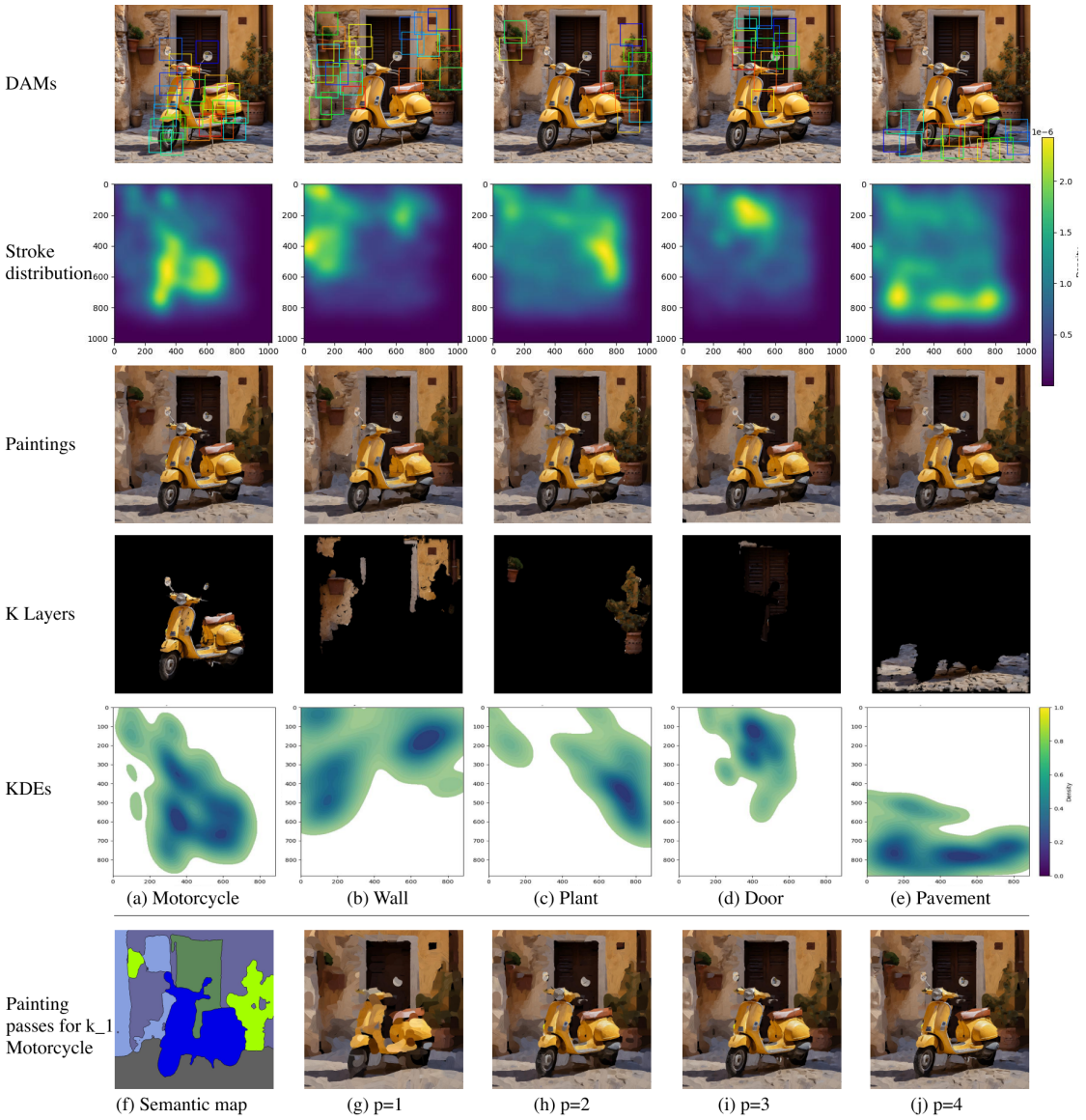


Figure B.3: Painting analysis based on semantic layers and dynamic attention maps (DAMs). Top row shows DAMs per semantic layer k. Second row shows the overall stroke distribution over the canvas based on biasing the painting towards different semantic areas, and the third row shows the corresponding paintings. The fourth row shows each layer k independent from the rest of the painting. The fifth row shows the Kernel Density Estimation of the strokes that correspond to only such layer k. The bottom row shows the semantic map, and the four painting passes, from coarse to fine, with a bias on the vehicle over the rest of the scene.

(a) Ours - Realism (b) L2P

Figure B.4: Comparison of our method (a) against L2P [2] (b) on realistic images. Note how L2P paintings have visible seams and blurry areas.

(a) Ours - Painterly (b) PT (c) SNP

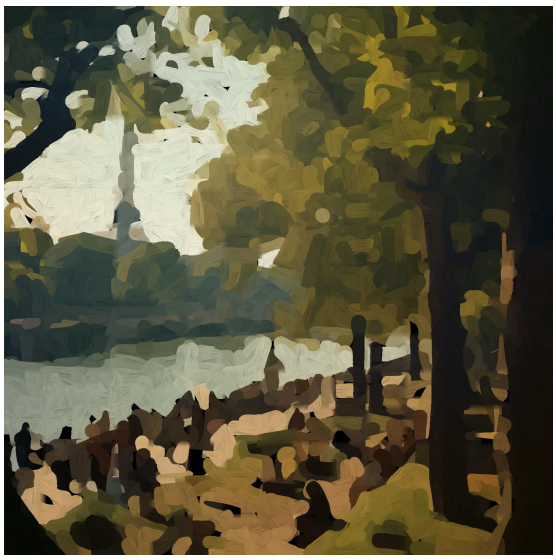
Figure B.5: Comparison of our method (a) against PT [4] (b) and SNP [5] (c) on painterly images.

(a)
Expressionist

(b) Painterly

(c) Realism

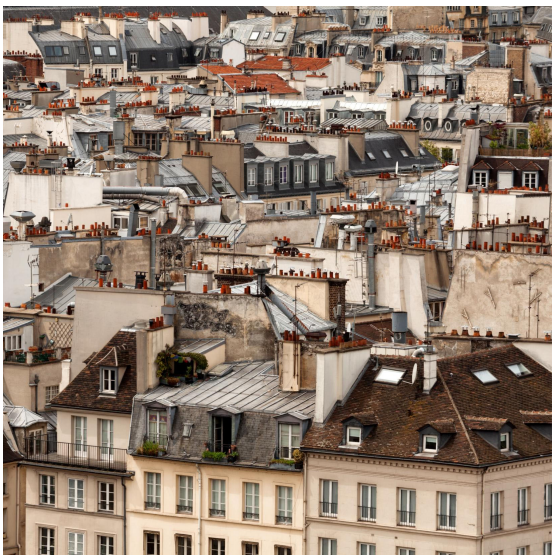
(a) Input



(b) Expressionist

(c) Painterly

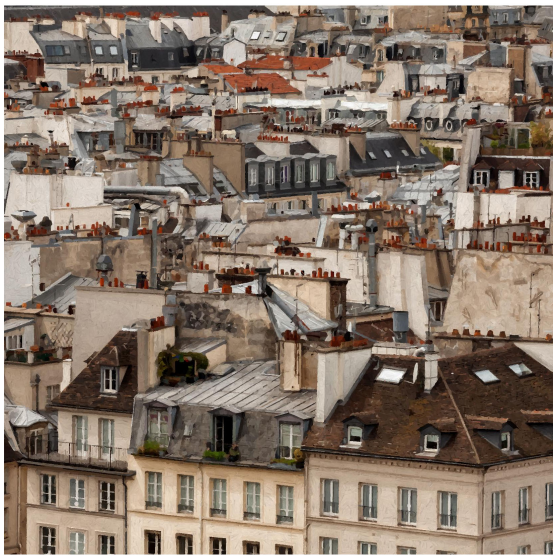
(d) Realism



(a) Input

(b) Expressionist

(c) Painterly



(d) Realism

(a)
Expressionist

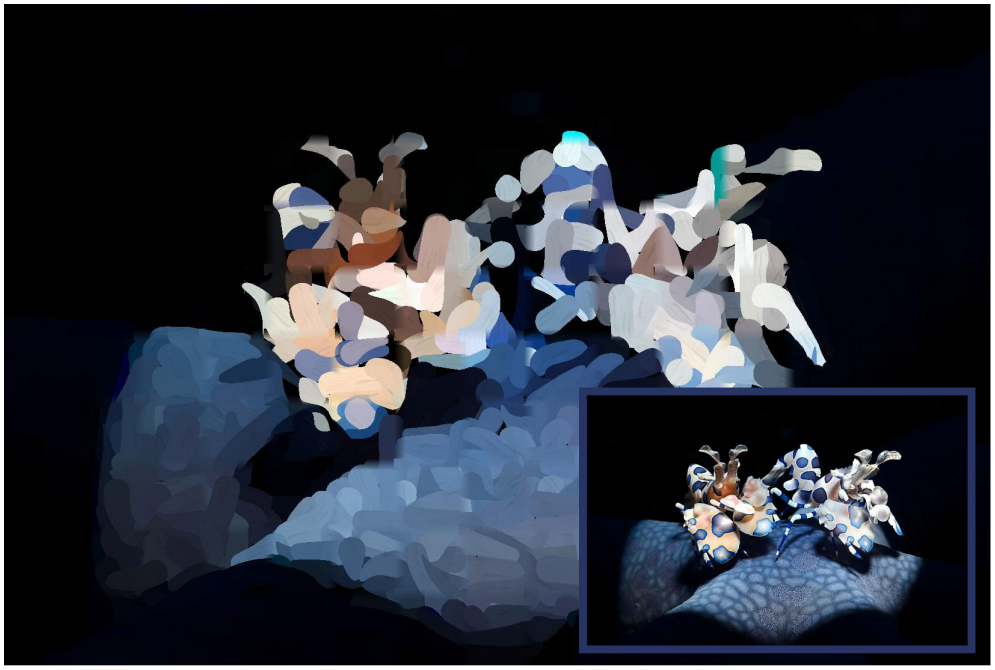
(b) Painterly

(c) Realism

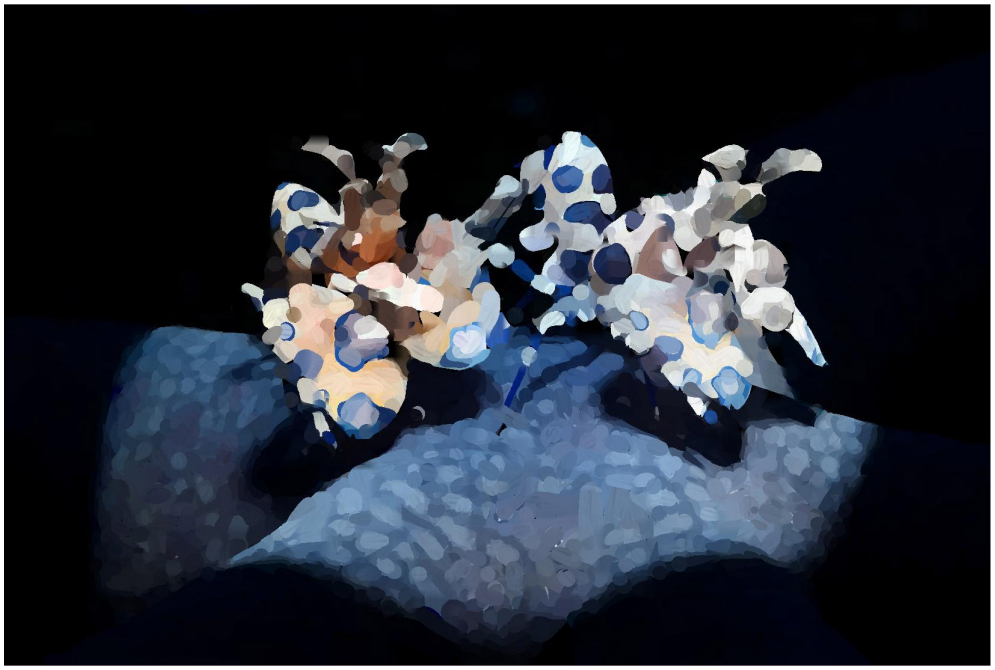
(a)
Expressionist

(b) Painterly

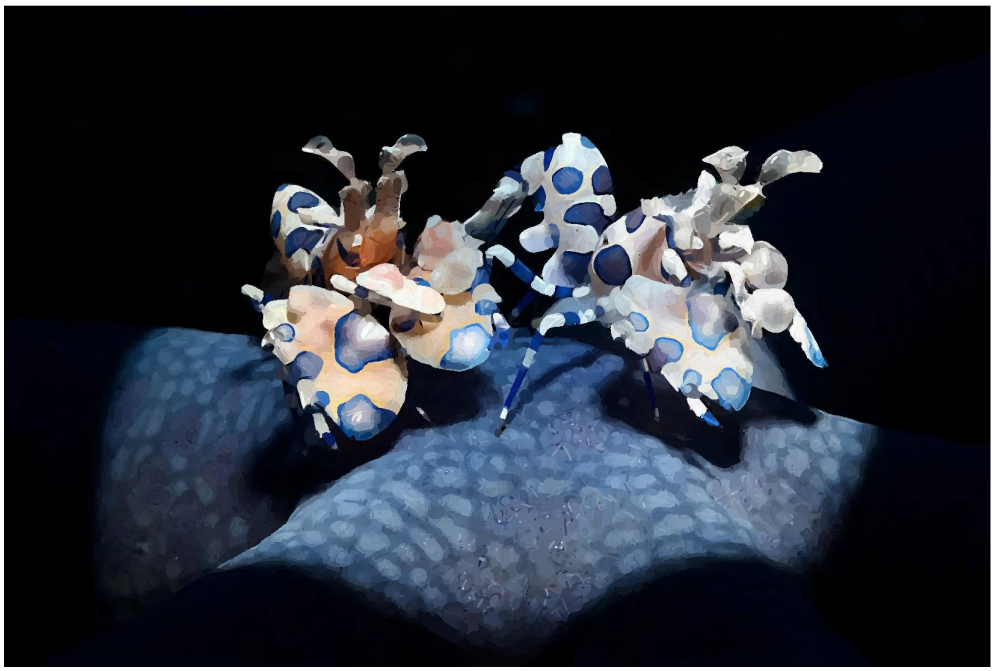
(c) Realism



(a) Abstract



(b) Painterly



(c) Realism

(a) Abstract

(b) Painterly

Coarse-to-fine (from left to right and top to bottom)

Figure B.6: Painterly style with dynamic attention maps corresponding to the last 3 painting layers are shown at the bottom.

Figure B.7: Painterly style with dynamic attention maps corresponding to the last 3 painting layers are shown at the left. Painting process is shown at the right.

Figure B.8: Realistic Style (left) and painterly style (right). Dynamic attention maps for the last painting pass are shown as an inset.

Figure B.9: Realistic painting process. Painted using uniform dynamic attention maps. From coarse (top left) to fine (bottom right)

Figure B.10: Pairings shown in user study from section 1. IDs from top to bottom: A5, A3, A4, and A6

Figure B.11: Pairings shown in user study from section 1 and 2. IDs from top to bottom: A1, A2, B1, and B5

Figure B.12: Pairings shown in user study from section 2. IDs from top to bottom: B3, B6, B2 and B4

References

[1] Nicolas Carion et al. “End-to-end object detection with transformers”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer. 2020, pp. 213–229.

[2] Zhewei Huang, Wen Heng, and Shuchang Zhou. “Learning to Paint With Model-based Deep Reinforcement Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[3] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[4] Songhua Liu et al. “Paint Transformer: Feed Forward Neural Painting With Stroke Prediction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 6598–6607.

[5] Zhengxia Zou et al. “Stylized Neural Painting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 15689–15698.

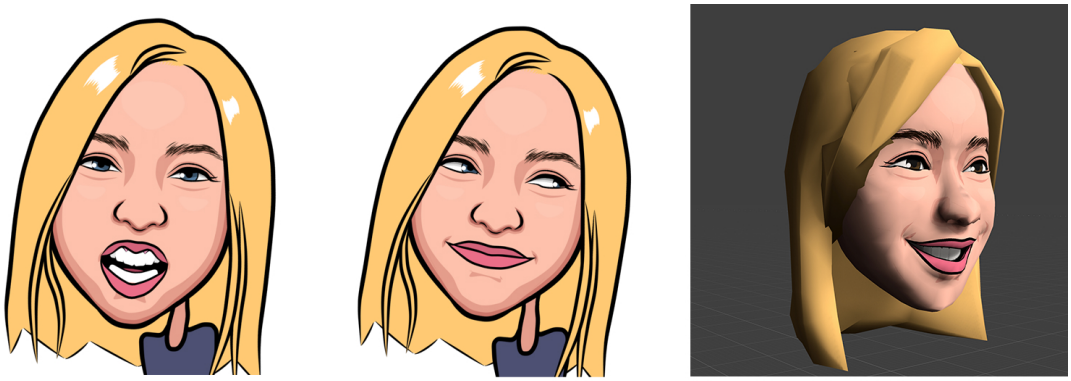
Appendix C

Cross-modal Latent Space Alignment for Image to Avatar Translation: Supplementary Material

C.1 Applications

This work does not delve into the various applications of parametric avatars. Nonetheless, we present the ease with which one can export avatar parameters into different environments. Compared to image-based avatars, parametric avatars offer a wider range of potential applications. Vector-based graphics, in particular, possess the advantage of being more readily exportable to other software programs for animation, 3D modeling, or virtual reality. This attribute is especially significant for game developers and animators who frequently transfer graphics between different software programs. An example of such applications is shown in Figure C.1.

Animation. The first two images are snapshots of the avatar produced with our method being animated in an animation software. The third image shows the same avatar in a 3D environment. Our method allows for a much more streamlined and efficient workflow than pixel-based methods, as the avatars can be modified and used in various different



(a) Animation (b) 3D

Figure C.1: Applications of our system beyond images. (a) We can export our parametric vector to animation software, and (b) 3D modeling software

contexts without having to create new images from scratch.

2D to 3D. Generating a 3D counterpart from our parametric avatar is simpler than reconstructing it from pixel-based methods. Initially, we created a template low-poly face geometry that included the head, ears, glasses, and several hair parts. This geometry was designed such that each control point from our generated parameters had a corresponding point of polygons. We also moved neighboring points accordingly, such as adjusting the head's size by moving all points of the head according to the outline of the face. Certain parameters in our avatars, such as shadows, determined the depth of the eyes. Concerning the face texture, we adapted the 2D caricature/avatar generator to render the front face and some additional parts. Then, we projected it onto the face polysurface.

C.2 Rendering Engine

To facilitate reproducibility, we provide a detailed explanation of our rendering engine, which serves as the intermediary between the avatar parameter values and their corresponding pixel values. Each parameter value (e.g. the x, y coordinates of the control

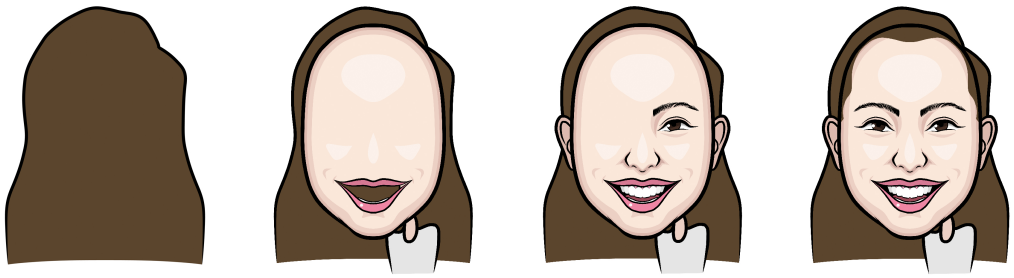


Figure C.2: Rendering composition used in our vector-graphics engine. Facial parts are composited on the canvas layer by layer.

points defining the upper curve of the eyelid) is bounded within a minimum and a maximum value determined based on the relative position in the canvas. For instance, no parameter that defines an eye can be any lower than the middle height of the face. This constraints are predetermined as part of the avatar design.

Our rendering engine works by drawing each component in layers and in a specific order. Specifically, we take the vector of parameters that correspond to the face shape, denoted as $\bar{y}_{facial_shape}$, and draw a set of curves that delineate the facial outline. We then apply the base R,G,B color for the face. Next, we update the canvas to add other facial components such as the mouth, nose and eyes, as shown in Figure C.2.

Our rendering engine, implemented in Python, is relatively straightforward and utilizes the ImageDraw module in the PIL library, which furnishes simple 2D graphics. In particular, we use the polygon function to draw closed shapes and compute the cubic Bezier equation to produce smooth open curves. Additionally, we taper the endpoints of curves to achieve a natural stroke effect. Finally, we fill the shapes with the corresponding color and maintain a black stroke for outlines.

C.3 High Resolution Samples

We append additional samples at a higher resolution at the end of this document to further demonstrate the capabilities of our model. Notably, we observe that our model is

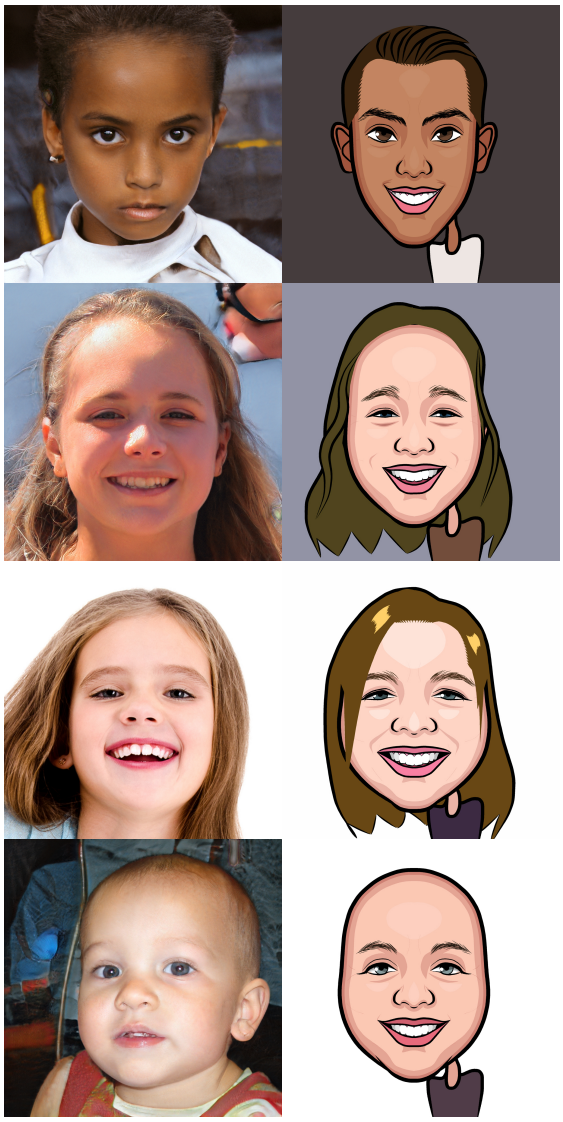


Figure C.3: Examples of avatars of children. Our model generalizes to kids, even though our paired image-parametric avatar dataset does not contain any children.

able to generate avatars of children despite our parametric avatar dataset not containing any children, as evidenced in Figure C.3. It is worth noting that while the image encoder E_s has been trained on a dataset that includes images of children, specifically the FFHQ dataset [5], the parameter encoder and decoder E_t and D_t, respectively, have not been trained on avatar vectors corresponding to child faces. Despite this, our model is able to accurately generate avatars of children, which is a testament to the effectiveness of our parametric representation and cross-modal framework.

C.4 Dataset

Our original dataset contains 393 pairs of portraits and parameters made by our artist. We have a test set of 30 pairs of portraits and parameters. In the following paragraphs, we first provide more details about the vector of parameters that define an avatar, and then explain how we make artificial transformations in the training set to get up to 9970 paired samples.

C.4.1 Parametric Avatar

Our parametrization is designed to represent a wide range of faces from different demographics with the mininum number of parameters. Thus, we group parameters based on facial components such as $\bar{y} := \{\bar{y}_{eyes}, \bar{y}_{nose}, \bar{y}_{mouth}, ...\}$, as explained in Chapter 6, and as shown in Table C.1, for a total of 629 values. All parameters follow a normal distribution, and before feeding parameters to the model, we normalize all parameters to be in the 0-1 range, and zero out the parameters corresponding to hair, glasses and cap, as they are retrieved from a databank. The majority of parameters define x, y coordinates of control points for cubic Beizer curves and polygons, R,G,B color values, and line thickness. The use of shadows and highlights make the avatar face have more or less depth, which is found useful to represent different ethnicity.

Non-trivial facial components. Some other facial parts, however, have more characteristics and variation than outlines and color, such as eyebrows. Besides the basic line shape, eyebrows present different hairline angle, volume, width, curviness, etc. A summary of parameters that define the eyebrows is shown in Table C.2.

C.4.2 Paired Dataset

Obtaining synthetic paired dataset requires performing the same augmentation in both image and avatar modalities. We generate a total of 9577 augmentations from our original paired dataset with two different methods:

Distortion-based Augmentations We first make augmentations for the existing images by slightly distorting facial features like eyes, head shape, nose, etc. We use Photoshop's Face Aware Liquify tool for this process. A script bridges the results of this tool with the avatar parameters to create the same deformation in the corresponding paired vector of parameters. We achieve a total of 3995 paired datapoints. We use this dataset to train our model and to compare with previous methods that we present in Chapter 6.

Interpolations We use a software that makes in-between face photographs given two anchor photographs, A and B. We can control the interpolation amount of A and B we want to maintain, for example 20% of A and 80% of B. In order to have the same interpolation in parameter space, we make a script that interpolates between two vector of parameters to have matching image-parameter vector pairs. Note that we do not interpolate hair parameters using this method because the hair parameters do not ensure smooth interpolations. This is due to the fact that some of the hair parameters are categorical variables. For instance, hair length is a categorical value with class from 0 to 4, and hair bangs has a binary variable which turn them on and off. In total we make 5975 interpolations.

Avatar part	# parameters	Method
Facial shape	56	Learned
Nose	33	Learned
Eyebrows	25	Learned
Eyes	83	Learned
Mouth	54	Learned
Ears	12	Learned
Highlights	80	Learned
Moustache	20	Learned
Beard	22	Learned
Hair	152	KNN Retrieval
Glasses	41	KNN Retrieval
Cap	46	KNN Retrieval
Body	5	Learned

Table C.1: Avatar parameterization by facial components. Hair, glasses and cap parameters are not learned by our main model. Instead, they are retrieved by a separate pipeline which we refer to as "hair pipeline" for brevity.

Parameter	Definition
Eyebrow volume	Density of hair lines
Initial angle	Angle of hair lines on center of the temple
Angle change	Initial angle of hair varies along the eyebrow angle gradually.
Eyebrow curve	Indicates the curviness of the eyebrow
Eyebrow hair width	Width of each hair line
Eyebrow hair length	Length of each hair line
Hair length Random	Length of hair line can be randomized. This parameter indicates random seed
Eyebrow hair spread	The root of each hair line will be perfectly aligned if this is zero. If bigger than zero, they will be randomly scattered.
Hair spread	Control spread of hair on eyebrows.
Eyebrow hair spread Y	Scatters hairlines only in Y direction to make the eyebrow taller
Fewer hair at end	Controls the number of the hair line at the edge of eyebrow
Hair spread at end	Controls taper at the edge of eyebrow
Eyebrow coverage	Scatters hairlines locally in the vertical direction

Table C.2: Breakdown of the parameters that define the eyebrows.

C.4.3 Unpaired Dataset

We can create a large amount of avatar parameters by swapping different facial components from random avatars. We synthetically create 100000 avatar vectors that do not have a corresponding portrait photograph. For our image encoder E_s, we use FFHHQ [5] (70000 aligned faces) and our augmented face dataset (9970 aligned faces) to have a total of 79970 aligned portraits.

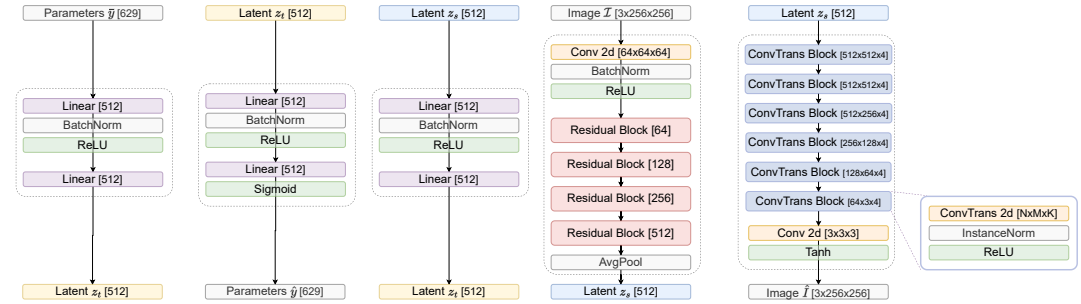
C.5 Model

C.5.1 Model Architecture

The network architecture is shown in Figure C.4. Encoders E_t, D_t and mapping network F are composed by a 2-layer MLP. The first layer is composed by a fully connected layer followed by BatchNorm and ReLU, and the last layer is a fully connected layer. These two networks output a 512-D latent vector. D_t uses a Sigmoid function after the last fully connected layer to output a 629-D vector of parameters in the range 0-1. The image encoder E_s outputs a 512-D latent vector, and the image decoder D_s takes in a 512-D latent vector and generate an image in 256x256 resolution.

	Unpaired training	Paired training
Epochs	20	10
Batch size	24	48
Optimizer	Adam	Adam
Learning Rate	0.0002	0.0002
Scheduler	None	Plateau
Latent Space Dim.	512	512
Optimizing Nets	E_t, D_t, E_s, D_s	F

Table C.3: Training Hyper-parameters



(a) Parameter Encoder E_t (b) Parameter Decoder D_t (c) Mapping Network F (d) Image Encoder E_s (e) Image Decoder D_s (f) ConvTrans Block

Figure C.4: Model architecture of the different components of our model.

C.5.2 Training

Table C.3 shows the hyperparameters used during training in both stages, the unpaired and the paired stages. We first train each modality-specific autoencoder (image and avatar vector) independently on parallel, on large-scale unpaired data. Once the networks converge, we discard D_s, fix E_s, E_t and D_t's weights, and train the new mapping network F. Note that F is trained on paired data only.

C.5.3 Inference and Hair Pipeline

At inference we only have a portrait image I, and we are interested in generating a vector of parameters that define an avatar for this image. For this, we only use our image encoder E_s, our mapping network F, and our parameter decoder D_t.

Our hair pipeline leverages a pretrained network $\mathcal{H}(I) \rightarrow \hat{a}$ that takes a real photograph as input and outputs predictions of 35 attributes of the input photograph. The list of attributes is:

age, male, smile, beard, moustache, sideburns, facialhair, no glasses, reading glasses,sun glasses, swimming goggles,roll, yaw, pitch, anger, contempt, disgust, fear, happiness, neutral, sadness, surprise, bald, hair invisible, white hair, gray hair, blond hair, brown hair, red hair, black hair, eye makeup, lip makeup, glasses, headwear, mask.

That is, $\hat{a}$ is a 35-dimensional vector of logits. We precompute a set of vectors $\hat{a}_{i=0}^{N}$ corresponding to a bank of 400 hairstyles and accessories that are paired with

photographs. As shown in Figure 4 in the main paper, at inference, and given an image I_i, we first perform a forward pass through our model, that is, $\hat{y}_i = D_t(F(E_s(I_i)))$ to get the predicted avatar parameters. We then perform a forward pass through our attribute predictor $\mathcal{H}(I_i)$ to get a query vector $\hat{a}_i$. We perform K-nearest neighbors to retrieve the most similar $\hat{a}_j$ from our databank. Because these faces are paired with ground truth avatar parameters, we can get the parameters corresponding to hair and accessories ($\hat{y}_j^{hair}, \hat{y}_j^{accs}$), respectively. We mask out the hair and accessory parameters from the predicted parameters $\hat{y}_i$ and paste the retrieved hair and accessory parameters from the hair pipeline.

Background. We can match the average background color by computing a segmentation mask I_s of the input image I. We use a BiseNet [13]-based network [8] to get I_s. We then calculate the mean RGB value in I's background to feed it to the vector graphics engine.

C.6 Ablation Studies

C.6.1 Pretrained Image Encoders Ablations

We are interested in testing the causal relationship between our face encoder and identity preservation capabilities of our model. That is, how well our encoder E_s helps preserve identity, keeping the rest of the networks (F and D_t) fixed. We generate avatars using off-the-shelf pretrained face encoders and compare these encoders with ours. ArcFace [2] is trained on a much larger dataset (1M images), and uses a ResNet100 [3] as a backbone. FaceNet [12] uses an inception model as backbone and is trained on 8M identities. pSp's encoder [11] uses a feature pyramid network [7] to get the representations of the input images, and it is trained on FFHQ dataset [5]. All encoders generate a latent vector $w \in \mathbb{R}^{512}$, except for pSp which projects to $\mathcal{W}+$. Here, we define w to be the average vector.

Figure C.5 shows comparisons of different encoders. Overall, our encoder and pSp

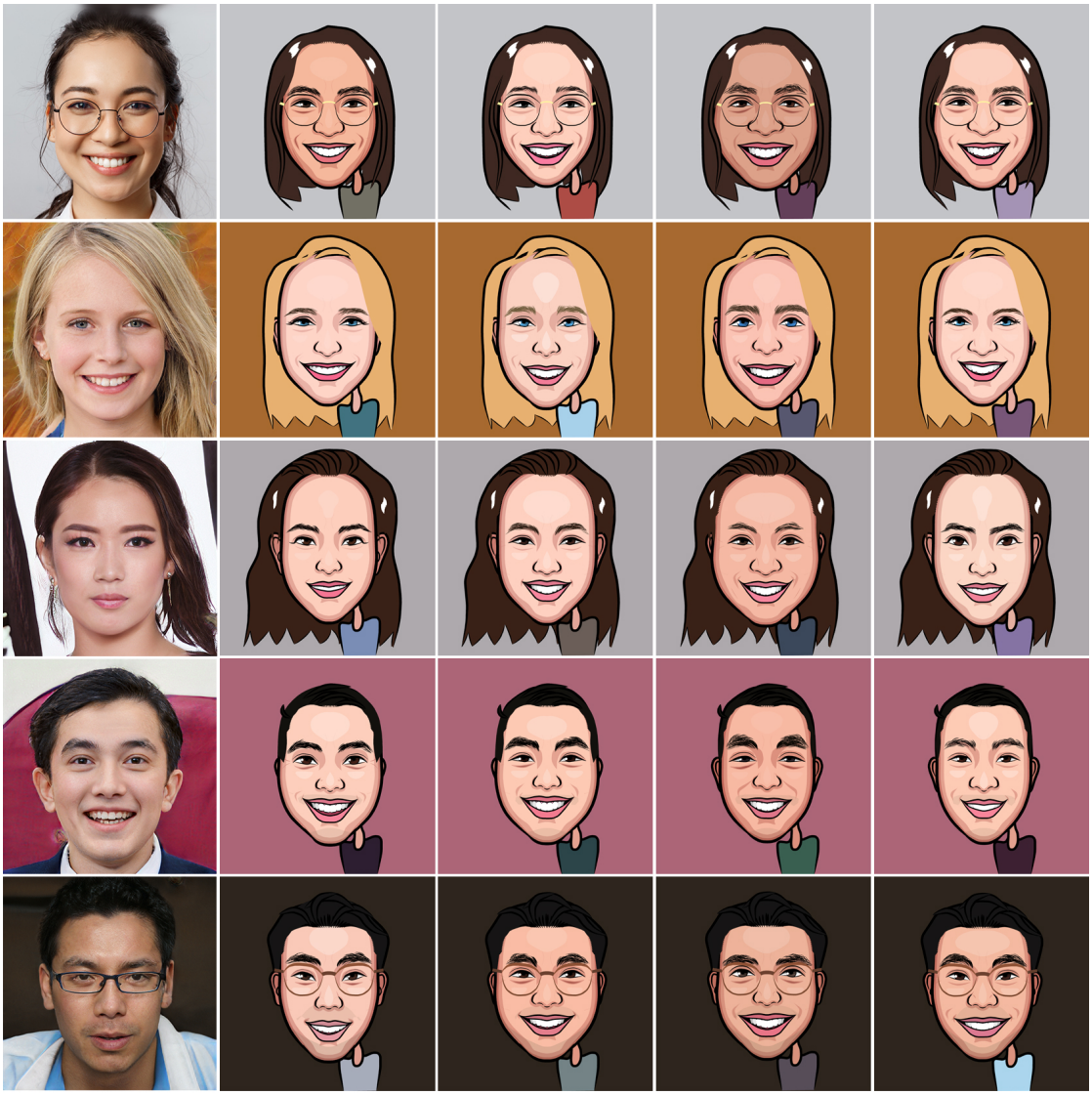
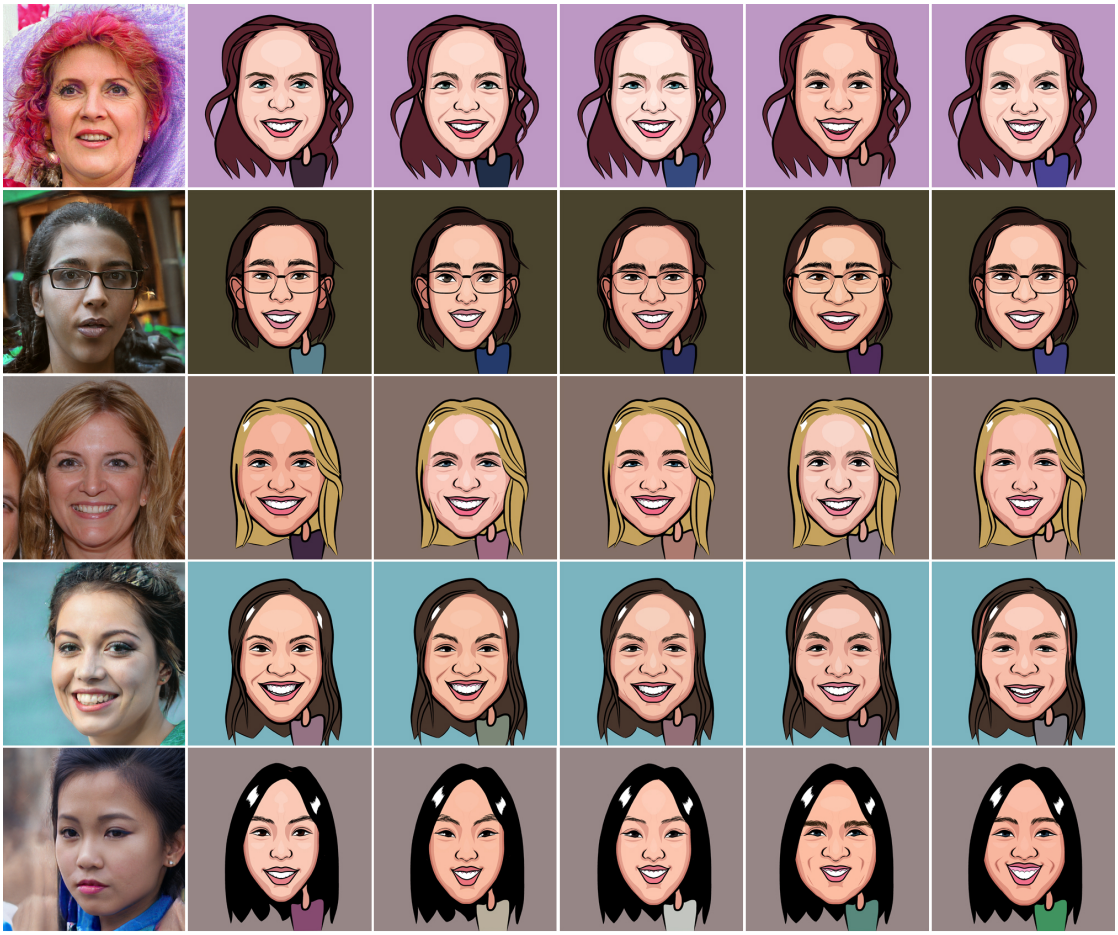


Figure C.5: Avatars generated by our method using different pretrained image encoders.

encoder output more successful results in terms of identity preservation. ArcFace struggles preserving the identity of the input more than the rest of the encoders, and FaceNet (column (c)) preserves the overall identity but fails to capture finer details in the mouth: in the first and second rows, it wrongly leaves a gap between the upper teeth and the bottom lip, and in the fourth row, it wrongly omits this gap. When the network uses our

encoder, the shape of the mouth is generally better than other decoders. Altogether, our pipeline design allows for other pretrained encoders to be plugged-in without significantly degrading the identity preservation of the generated avatars.



(a) Input (b) Ours$_{\sim 10k}$ (c) Ours$_{\sim 4k}$ (d) Ours$_{2k}$ (e) Ours$_{1k}$ (f) Ours$_{500}$

Figure C.6: Effect of amount of paired data. (b) Our model trained on 9970 pairs. (c) Our model trained on 3995 pairs. (d) Our model trained on 2000 pairs. (e) Our model trained on 1000 pairs. (f) Our model trained on 500 pairs

C.6.2 Effect of Amount of Data

We are interested in testing the amount of paired data that is needed so that our model does not break. Figure C.6 shows examples of our model trained on different amounts

of data. Specifically, we train our model on 9970 image-avatar parameters pairs (column (b)), 3995 (column (c)), 2000 (column (d)), 1000 (column (e)), and 500 (column (f)). As noted in Chapter 6, our model starts breaking identity preservation when trained on 2000 paired samples or fewer, as shown in columns d, e, and f. When trained on 1000 samples or fewer, the model produces exaggerated facial features, such as a bigger and narrower mouth (third and last rows), larger head size (first, second, and fourth rows), and nose misplacement (second, third, and last row). When trained on nearly 4000 samples (column (c)), our model maintains the input's identity overall. However, it sometimes misses more nuanced features, making errors in scale and placement. When trained on nearly 10000 paired samples, our model captures much better identity features. There is a visual relation of amount of data and avatar quality, and our method needs at least 2000 paired data samples in order to preserve identity and generate high quality avatars.

C.6.3 Loss Ablations

We provide more visual comparisons between our model using different losses in Figure C.7. This ablation study tests our mapping function F under different objectives. For an easier evaluation, we re-state our objective functions here. Our full objective is composed by a (1) reconstruction loss $\mathcal{L}_{rec} = (\hat{y} - \bar{y})^2$, where $\hat{y}$ is the predicted parameters output by a forward pass through image encoder, mapping network and parameter decoder $D_t(F(E_s(I)))$ given an image I, and $\bar{y}$ is the vector of parameters that is paired to I; and by a (2) alignment loss. Our alignment loss is composed by three terms: $\mathcal{L}_{mse}$, $\mathcal{L}_{cossim}$, and a weight regularization term $\mathcal{L}_w$. The first two measure a predicted latent code given by a forward pass through the image encoder and mapping network $z_f = F(E_s(I))$ and the output of a pretrained parameter encoder E_t that we refer as the "expert" network, $z_t = E_t(\bar{y})$. Our weight regularization term imposes a regularization on the last layer of F's weights, built on the assumption that F's last layer should be as similar as E_t's last layer weights, since the former wants to project a vector into the same latent space as the latter, and both networks are designed to be a 2-layer

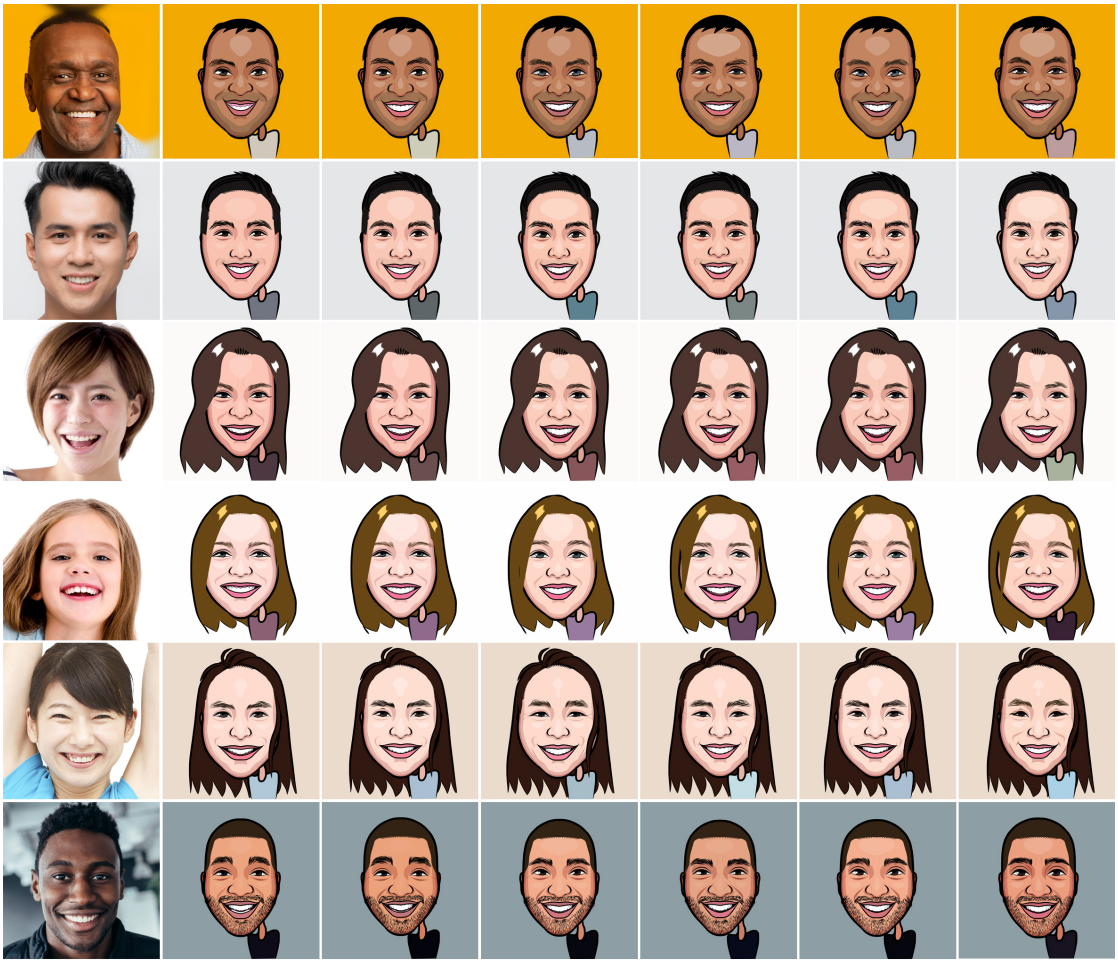


Figure C.7: Model variations without proposed losses. (b) only uses reconstruction loss $\mathcal{L}_{rec}$ between generated and ground truth parameters. (c) only uses weight alignment loss $\mathcal{L}_w$. (d) uses reconstruction $\mathcal{L}_{rec}$ and cross modal loss $\mathcal{L}_{cm}$ but does not use weight alignment loss $\mathcal{L}_w$ (e) uses all except for the MSE loss term in $\mathcal{L}_{cm}$. (f) uses all except for the cosine similarity term in $\mathcal{L}_{cm}$ (g) uses all losses

MLP.

All columns in Figure C.7 keep the reconstruction loss $\mathcal{L}_{rec}$. Column (b) removes the alignment module entirely and only keeps the reconstruction loss. The rest of the columns (c,d,e,f, and g) keep different terms from the alignment module. Specifically, column (c) only maintains the weight regularization term $\mathcal{L}_w$. Column (d) does the

opposite and only keeps this term (besides reconstruction loss). Columns (e) and (f) remove $\mathcal{L}_{mse}$, and $\mathcal{L}_{cossim}$ from our alignment module, respectively. Last column (g) keeps all the losses.

Without our alignment module (column (b)), the model does not capture well the identity of the input person. From here, the rest of the columns do a relatively good job at preserving identity. However, by just using our weight regularization (column (c)), the model fails to capture some facial components that contribute to the entire identity. For instance, in the third row, the eyes and the nose are closer together than the input image. In the fourth row, the model is generating somehow features of an older face. In the fifth row, the generated eyes do not register the smiling and consequently more stretched eyes from the input image. In the bottom row, the eyes are not as big as the person in the input image.

Without our weight regularization term (column (d)), the model does a better job at fixing the problems explained in the previous paragraph, but still generates facial features that are a bit off from the input image. For example, in the bottom row, the nose shape is different from the input image. In the fifth row, the cheeks' wrinkles are not as pronounced as in the input image. In the fourth row, the eyes of the little girl are not accurate, and a similar inaccuracy happens in the first row.

Without the MSE term in our alignment loss (column (e)), the model fails to register the eye shape accurately in the last two rows, as well as the second row. In the first row, the face shape or outline is rounder than the input image. In the second row, the nose is not as accurate as the one generated using all loses (column (g)). However, it seems like this loss term might be the less important.

Without the cosine similarity loss term (column (f)), the model generates less accurate avatars than column (e) and (g). In the first row, the eyes are bigger and have a different color than the input image. In the third row, the face shape is longer and less round than the input image. In the fourth row the eyes do not look similar to the input image, and the mouth does not register the open gesture (see how columns (e) and (g) do present a gap between the upper teeth and the bottom lip). The identity of the last

two rows is not successfully preserved.

By using all proposed losses our model consistently preserve identity better across age, gender and ethnicity.

C.7 Comparison with Previous Methods

Figure C.8 shows more examples of comparisons with previous methods. We include MSGAN-Pix2pix [9] in this figure. We use official implementations of each method and train with the default hyperparameters. Overall, we see our method generates better quality and better identity preserved avatars. The GAN-Adaptation method [10] (column (b)) works better for males than females. Hair color is generally not well preserved (all rows except for the last two), and it wrongly adds eyeglasses on some inputs (first three rows and sixth). It generally preserves well cheek wrinkles. StyleCariGAN [4] (column (d)) generates diverse and good quality avatars, but sometimes it struggles preserving the identity of the input. In the top row the avatar does not register the elongated facial shape of the input. In the second row, the avatar face is not as round as in the photograph. In the fourth row the model is generating eyeglasses, and in the fifth and sixth row, identity is almost completely missing. U-GAT-IT [6] (column (e)) produces smooth colors and generally preserves identity better than other previous methods. However, the generated avatars present some deformations that degrade the quality of the avatar. It struggles with long hair, but in general it preserves well key facial features such as eyebrows, eyes, nose or face shape. StarGANv2 [1] (column (f)) produces higher quality avatars than other previous methods, but identity is not as well preserved. The last three rows have similar faces. The first row does not preserve the elongated face from the input. MSGAN-Pix2pix [9] (column (g)) generates diverse avatars and generally registers well different hairstyles and preserves relatively well facial shape (except for the example in the third row). The quality of the image generated by this method is compromised by the artifacts the model generates.

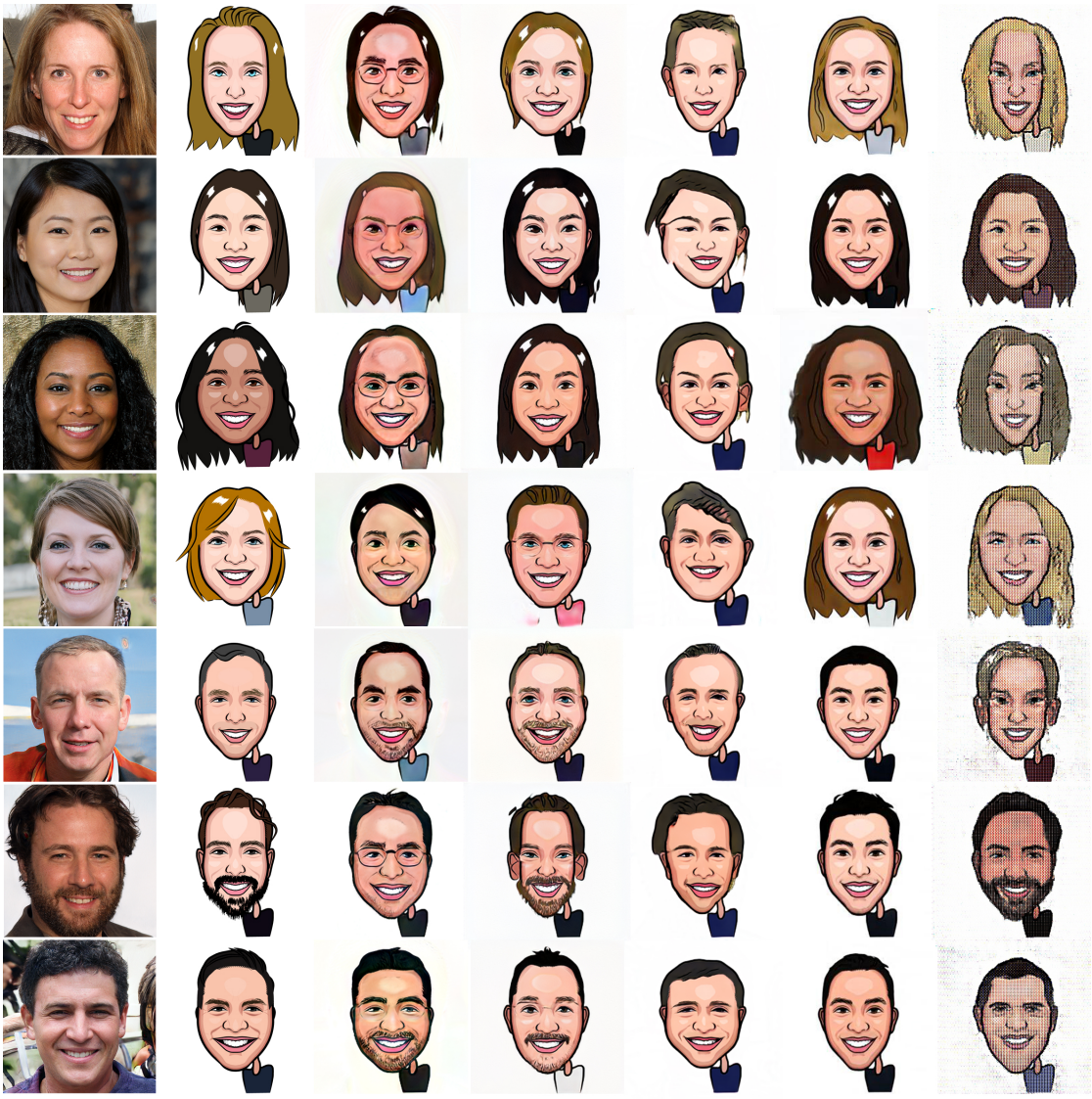


Figure C.8: Comparison of our model with state-of-the-art methods. All methods are trained on the same dataset, comprising 3995 image-avatar pairs.

C.7.1 Inference Runtime

Table C.4 shows the inference runtime of our method in comparison to previous methods. Methods free of GAN-inversion or render engines such as U-GAT-IT [6] and MSGAN-Pix2pix [9] are the fastest to generate avatars, as these models only need a for-

Method	Gen.	Inv.	Hair	Total
U-GAT-IT [6]	0.0108			0.0108
GAN-Adapt. [10]	0.013	60.975		60.988
StarGAN-v2 [1]	1.222			1.222
MSGAN-pix2pix [9]	0.0102			**0.0102**
StyleCariGAN [4]	0.081	77.946		78.027
Ours	1.229*		2.747	3.976

Table C.4: Inference Runtime computed in seconds. GAN-Inversion is needed in StyleGAN2-based methods, where the latent vector corresponding to the input image needs to be found. Inversion is computed over 1000 iterations. *Our forward includes the forward pass through the inference module and the time that of the vector graphics engine to render the avatar in image space.

ward pass through their networks to directly generate the avatar in image space. GAN-Inversion methods such as StyleCariGAN [4] and GAN-Adaptation [10] are slower due to the optimization step needed to compute the StyleGAN latent vector that corresponds to the input image. These methods need around one minute to generate an avatar. Our method generates avatar images in around 4 seconds. This is due to the rendering step that converts the generated vector of parameters to an image ($\sim 1.2s$), and the hair and accessory retrieval pipeline ($\sim 2.7s$).

C.7.2 User Study

The user study was conducted using Amazon Mechanical Turk. To maintain anonymity, we do not collect emails nor ask about any demographic data. Monetary compensation in line with standards for small rating tasks was offered. We obtain a total of 113 responses from different users. The survey is structured in two parts. The first evaluates identity preservation, that is, how well the avatar capture the facial features of the portrait image. The second part evaluates avatar quality, which is how well the avatar itself maintains facial proportions, disregarding image quality such as resolution. The overall user study is estimated to take around 10 minutes. As described in Chapter 6, we compare our method with 5 previous methods. All users in this study see the same pairs of avatars. In other words, we design only one survey, which is fixed for all participants.

Evaluation on Identity Preservation For the first part, evaluation on identity preservation, we present the users with 15 comparisons, each one showing always our method compared to another previous method. Since we are comparing with 5 previous methods, in general we compare our method with each previous method 3 times. Before starting this section of the survey, we provide a clarification example of the questions in this section:

EXAMPLE OF QUESTION IN SECTION 3: Question: Given a photography, select which avatar preserves the person's identity more accurately. You should select left or right. Answer: It is clear from this image that the right avatar is able to preserve much better the identity of the person in the photography. The shape of the nose and the shape of the eyes are very similar. The overall face structure is similar too, including the face outline. Even without considering eyeglasses, the right avatar does a better job. Although the hair is similar in both avatars, they fail to capture the color.

Evaluation on Avatar Quality For the second part of the survey, evaluation on avatar quality, we show 5 two-way comparisons. That is, we compare our method with each of the previous methods only once. This decision is based on: (1) The fact that all methods are consistent in their avatar representations regardless of the avatar instance, (2) avoid redundancy, (3) make the survey less time consuming. We provide the following instructions before starting this section of the survey:

This section evaluates quality and visual appealing of the avatars. That is, how good an avatar looks. You can evaluate them based on their visually appealing, color consistency, relative proportion of facial features, or proportion of the face as a whole, among others factors. PLEASE, DO NOT FACTOR IN THE IMAGE QUALITY (i.e, resolution or crispness) WHEN CHOOSING YOUR ANSWER.

An example question that evaluates identity is shown in Figure C.9, and an example question that evaluates avatar quality is shown in Figure C.10

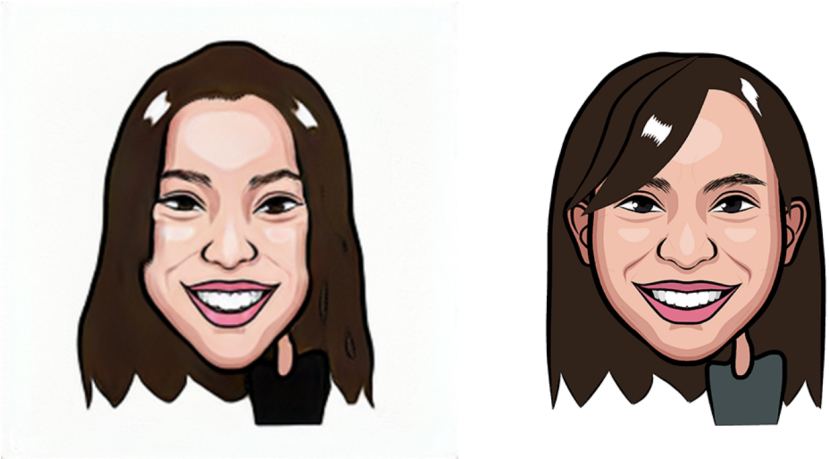
(3.3) Given this photograph:

(3.3) Which avatar preserves the identity of the person above more accurately? *

- ◯ Left Avatar
- ◯ Right Avatar

Figure C.9: Example of question on identity preservation in our user study

(4.1) Which avatar has a better quality? *



- ○ Left Avatar
- ○ Right Avatar

Figure C.10: Example of question on avatar quality in our user study

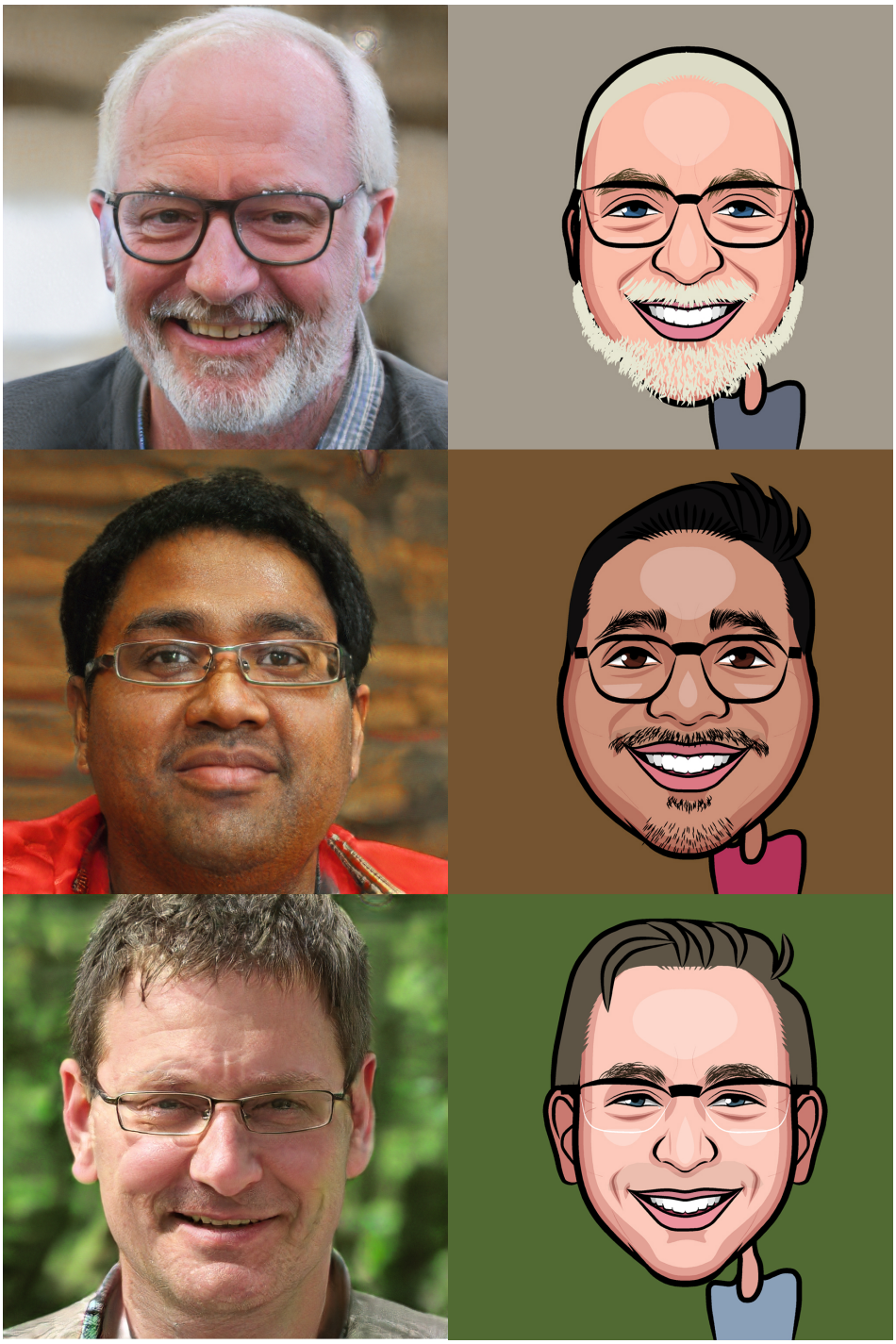


Figure C.11: Samples generated using StyleGAN-2 generated faces.

Figure C.12: Samples generated using StyleGAN-2 generated faces.

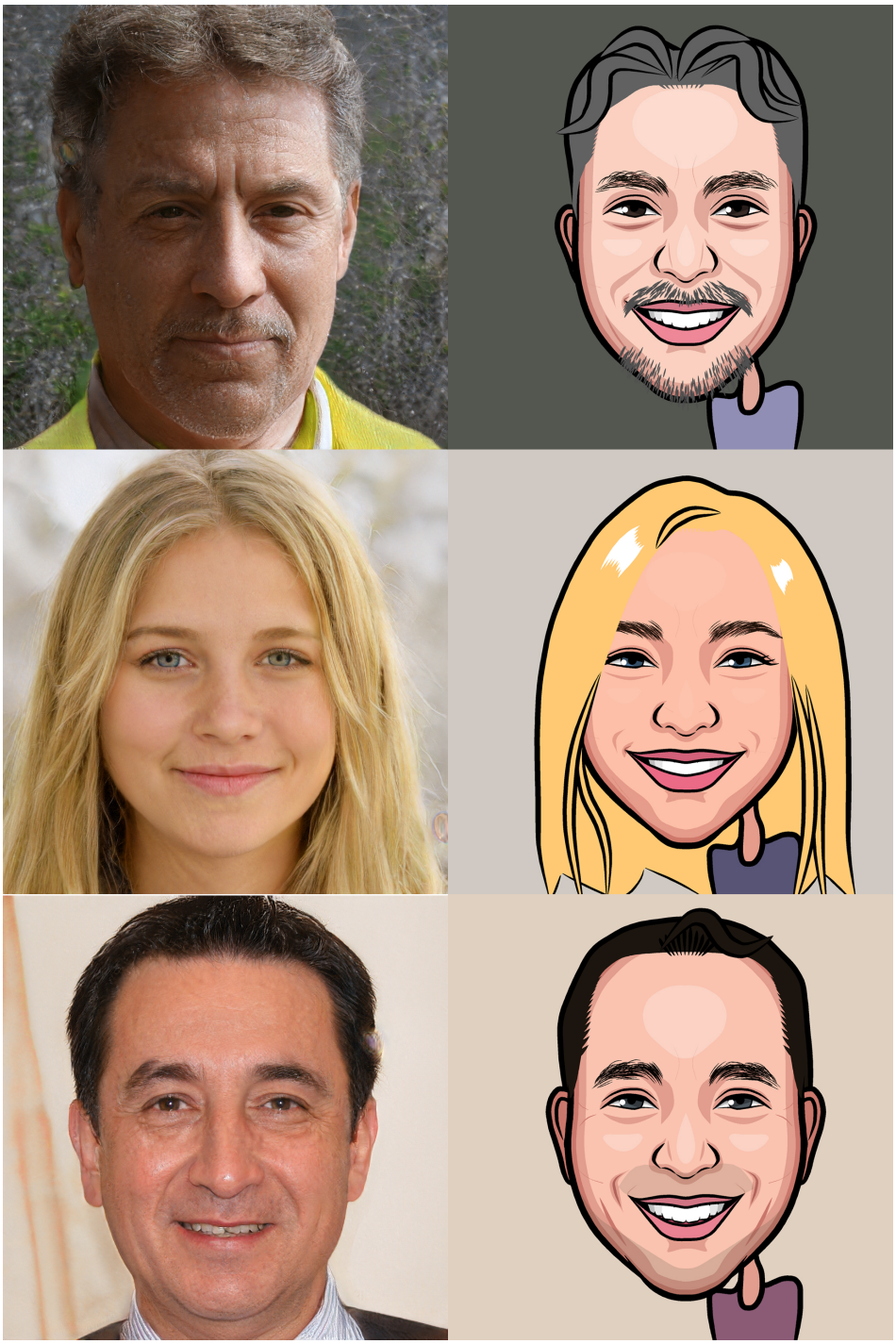


Figure C.13: Samples generated using StyleGAN-2 generated faces.

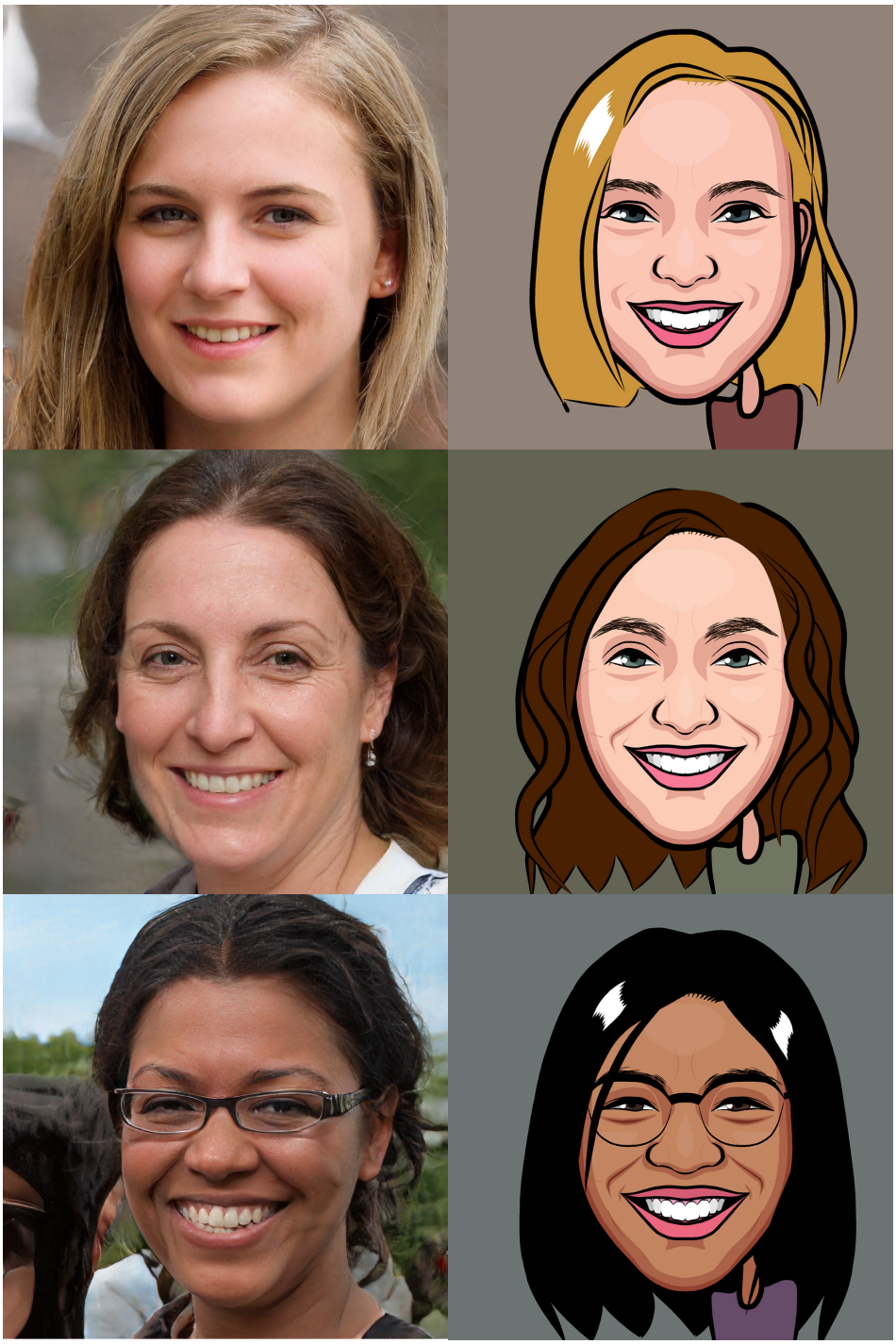


Figure C.14: Samples generated using StyleGAN-2 generated faces.

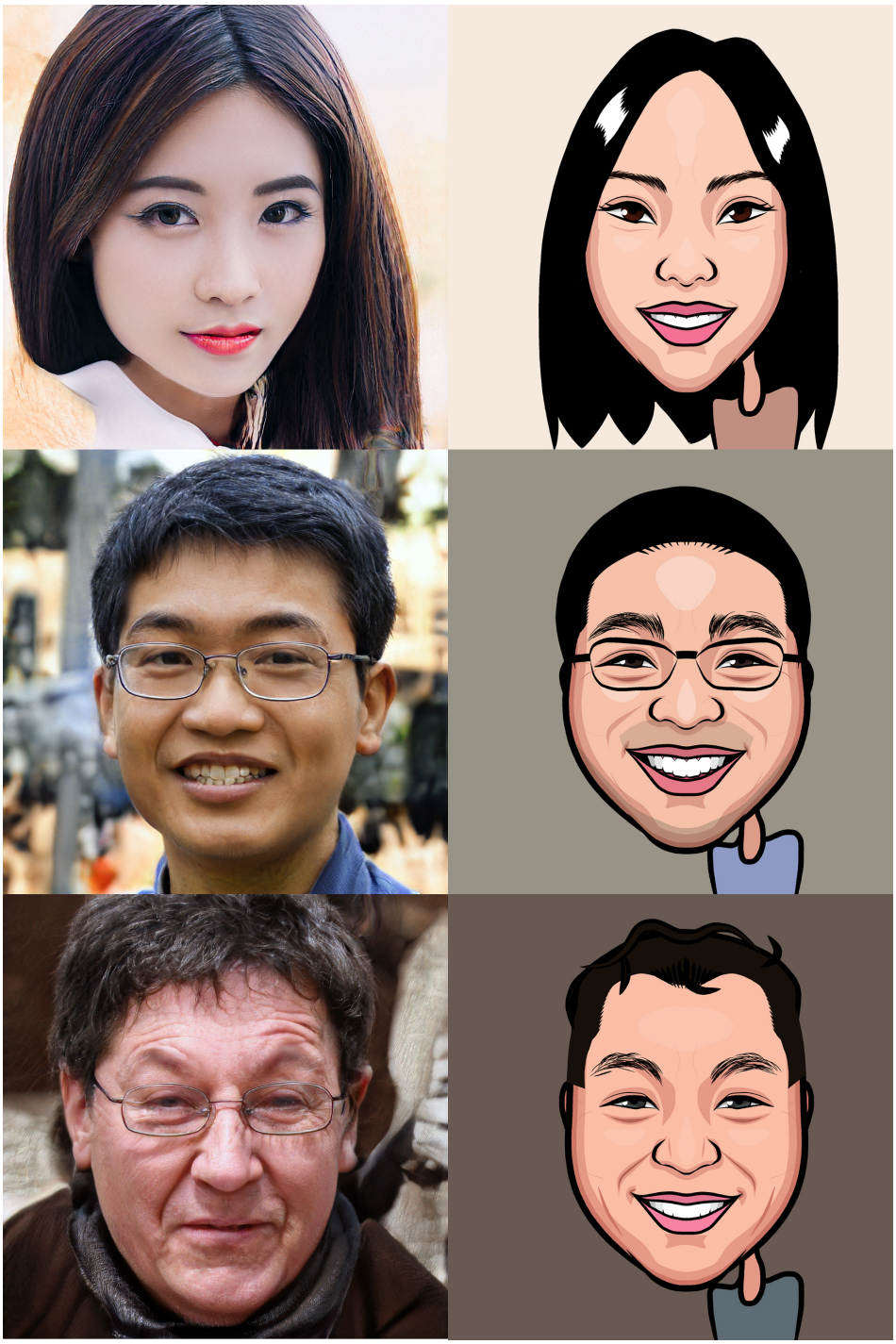


Figure C.15: Samples generated using StyleGAN-2 generated faces.

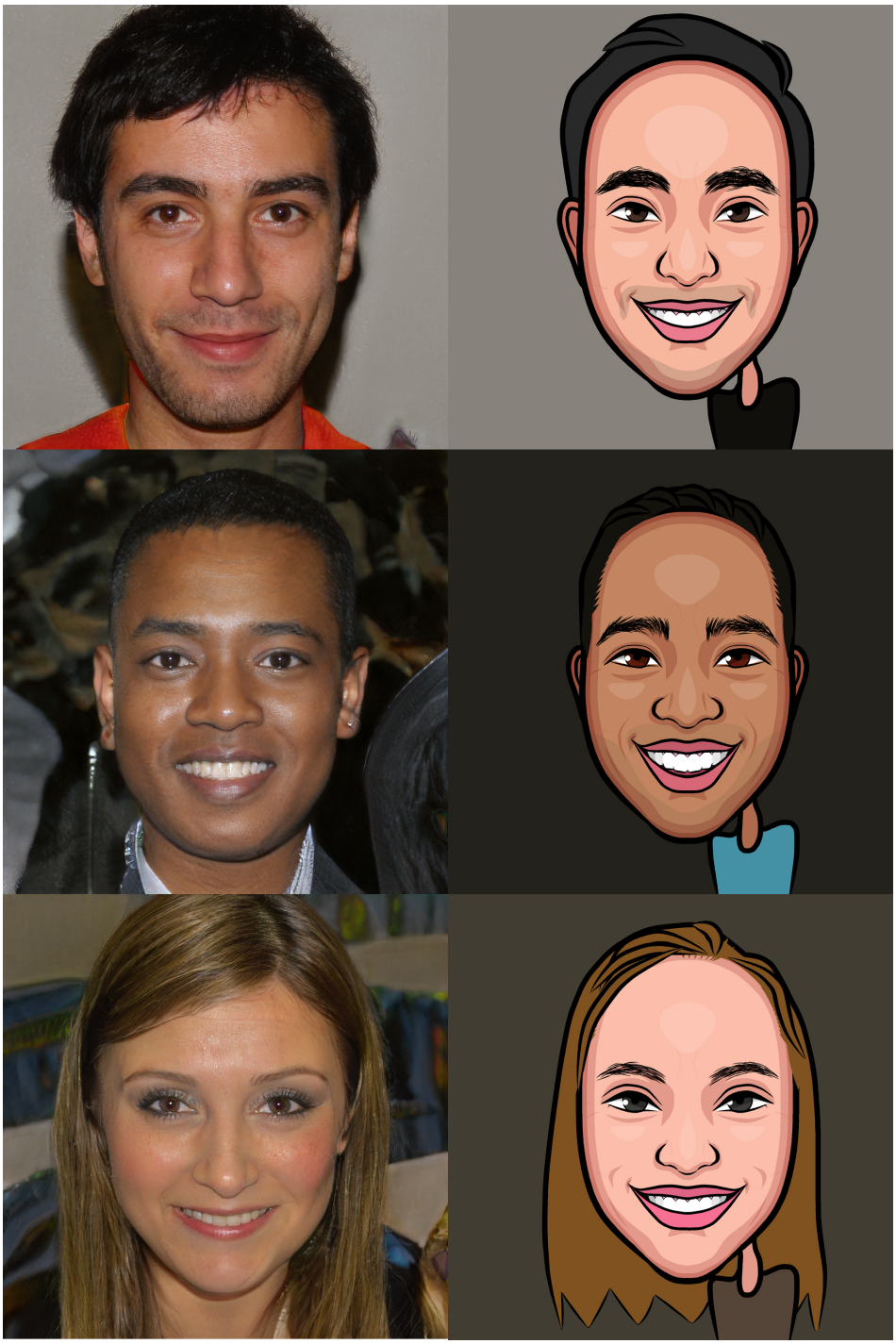


Figure C.16: Samples generated using StyleGAN-2 generated faces.

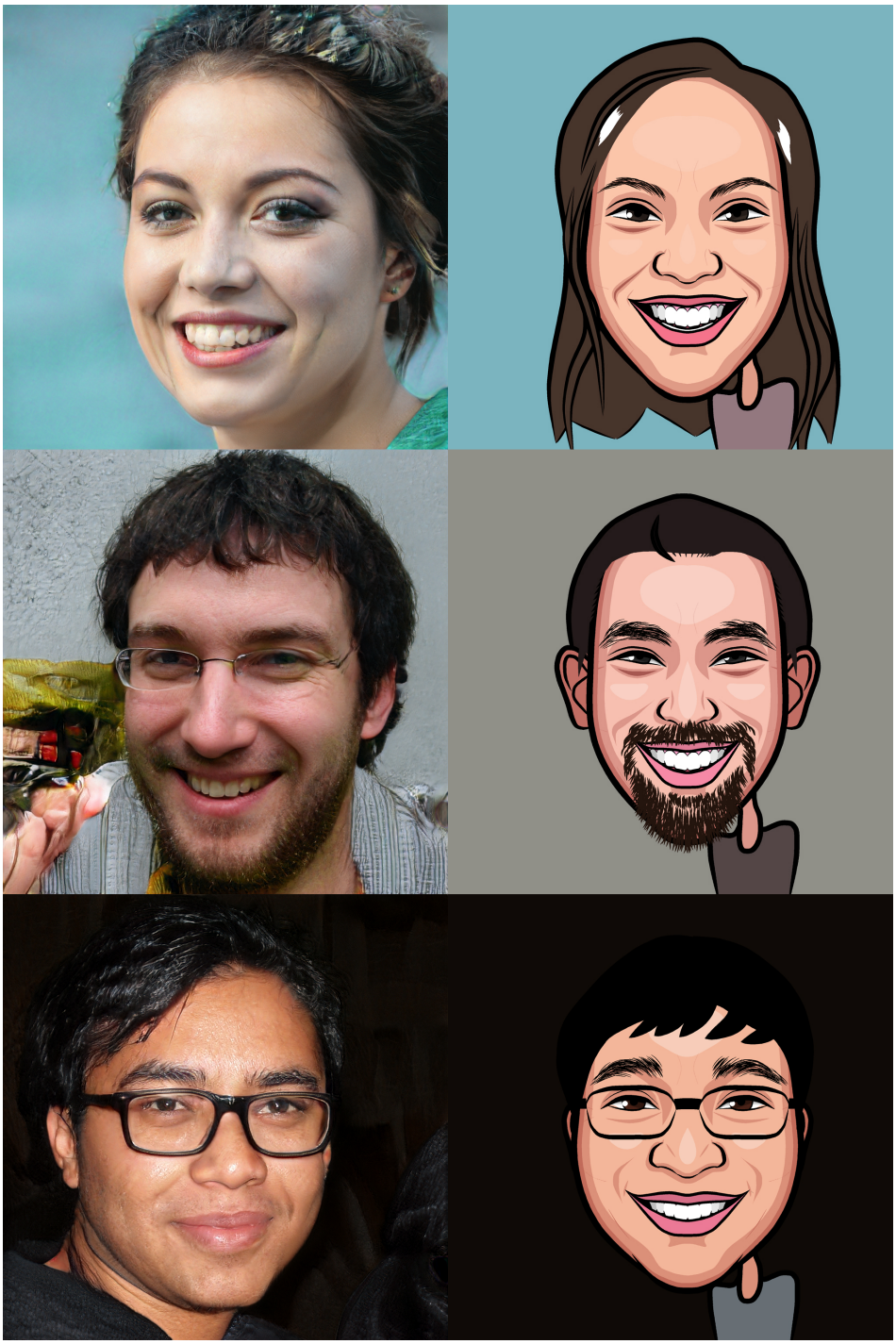


Figure C.17: Samples generated using StyleGAN-2 generated faces.

References

[1] Yunjey Choi et al. “Stargan v2: Diverse image synthesis for multiple domains”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8188–8197.

[2] Jiankang Deng et al. “Arcface: Additive angular margin loss for deep face recognition”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 4690–4699.

[3] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[4] Wonjong Jang et al. “StyleCariGAN: caricature generation via StyleGAN feature map modulation”. In: *ACM Transactions on Graphics (TOG)* 40.4 (2021), pp. 1–16.

[5] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[6] Junho Kim et al. “U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation”. In: *arXiv preprint arXiv:1907.10830* (2019).

[7] Tsung-Yi Lin et al. “Feature pyramid networks for object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[8] Ingo Lütkebohle. *Face Parsing*. https://github.com/zllrunning/face-parsing.PyTorch. 1028.

[9] Qi Mao et al. “Mode seeking generative adversarial networks for diverse image synthesis”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 1429–1437.

[10] Utkarsh Ojha et al. “Few-Shot Image Generation via Cross-Domain Correspondence”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 10743–10752.

[11] Elad Richardson et al. “Encoding in style: a stylegan encoder for image-to-image translation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 2287–2296.

[12] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823.

[13] Changqian Yu et al. “Bisenet: Bilateral segmentation network for real-time semantic segmentation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 325–341.