CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Machine Learning for generative painting informed by visual arts

Investigating painting techniques for designing Machine Learning pipelines of generative painting

Master's thesis in Computer science and engineering

CHAOMING WANG

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2023

Master's thesis 2023

Machine Learning for generative painting informed by visual arts

Investigating painting techniques for designing Machine Learning pipelines of generative painting

CHAOMING WANG

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2023

CHAOMING WANG

Supervisor: Kıvanç Tatar, Department of Computer Science and Engineering
Examiner: Palle Dahlstedt, Department of Computer Science and Engineering

Master's Thesis 2023
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 76 397 6889

Typeset in LaTeX
Gothenburg, Sweden 2023

Machine Learning for generative painting informed by visual arts
Investigating painting techniques for designing Machine Learning pipelines of generative painting
CHAOMING WANG
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Visual art practice is a complicated, varied, creative process based on the artist's style and preferences. Although many studies have attempted to apply artificial intelligence techniques to art production and statistical analysis, there is still significant scope for exploring how to incorporate the techniques in visual arts practices into generative painting pipelines using Machine Learning. This thesis applies machine learning to analyzing painting techniques in painting practices with a research-through-design approach. The problem is mainly presented as tasks such as segmentation of artworks (in this thesis, paintings), stroke prediction, and the presentation of painting processes based on different painting techniques through different algorithmic pipelines. The results show that most segmentation models based on photo training are challenging to apply to the segmentation of artwork components directly, and relevant improvement solutions are discussed in Chapter 6. In addition, due to the diverse presentation of painting art, this paper presents different painting techniques based on the foreground and background segmentation and 'blocking-in' techniques based on line detection. It discusses the possibility of transferring these painting processes to other painting processes.

Keywords: painting techniques, visual art, image segmentation, machine learning.

Acknowledgements

Completing this thesis would not have been possible without the help of many people. I extend my sincere gratitude to everyone who has contributed to my thesis, in ways both big and small. I am truly fortunate to have such a supportive network of people around me, and I am grateful for the opportunity to have undertaken this research.

I would like to express my sincere gratitude to my supervisor Kıvanç Tatar, for his patience, guidance, and support. I have gained a lot during the project, both in the task-related topics and the ability of art perception and appreciation. Most importantly, I received professional and constructive guidance on doing research. I am truly grateful for their mentorship and for pushing me to achieve my best work.

I would also like to express my gratitude to the esteemed professors in Chalmers, whose teachings and guidance have shaped my research and contributed significantly to my intellectual growth. In particular, I would like to express my gratitude to Lennart Svensson, the professor of the deep learning course, for his extensive coverage of the field of deep learning and generative art, and I have found ideas to address many of the problems in this research by asking him for advice after class. In addition, I would like to thank other researchers in the generative arts and machine learning community whose related research has inspired my continued exploration in this area.

Finally, I am deeply grateful to my family and friends, who have been a constant source of support and encouragement during this challenging journey. Their unwavering belief in me has given me the motivation to continue working towards my goals, even when the going gets tough. I thank them for always being there for me, for listening to my ideas, and for their unwavering love and support.

Chaoming Wang, Gothenburg, April 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in appearance order:

DALLE	Neural network created by OpenAI, that can generate images from natural language descriptions
CNN	Convolutional neural network
GAN	Generative adversarial network
VQ-VAE	Vector quantized variational autoEncoders
LDMs	Latent diffusion models
JPG	Joint photographic experts group
PNG	Portable network graphic
RGB	Color space in (Red,Green,Blue) format
HSV	Color space in (Hue,Saturation,Value) format
CMYK	Color space in (Cyan,Magenta,Yellow,Key) format
RtD	Research method of "Research-through-Design"
HCI	Human computer interaction
SNIC	Swedish national infrastructure for Computing
DCGAN	Deep convolutional generative adversarial network
DRAM	Diverse realism in Art Movements
ASPP	Atrous spatial pyramid pooling
KDE	Kernel density estimation
GMM	Gaussian mixture model
MSE	Mean squared error
LMSE	Local mean square error
ETF	Edge tangent flow
MOS	Mean opinion score

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

t	Index for time step

Sets

$\mathcal{C}_0$	Empty canvas at initial time step
$\mathcal{C}_t$	Progressively generated canvas at time step t
$\tilde{\mathcal{C}}$	Reference artworks shown in canvas

Parameters

X_t	Vector of parameters estimated in the t iteration
μ	Learning rate for optimization process
S_{map},S_{shape}	Matrix of boolean values
$Color_{map}$	Matrix of optimized pixel values
$Color_{comp}$	Computed matrix of values for pixel color
w_{ts}	Weights for strokes at time step t
v_{tc}	Previous canvas state at time step t
s_t	Weighted sum of generated strokes at time step t
$P(x)$	Probability density of the input image
w_k	Weight for each component in clustering
$N(x\|\mu_k, \theta_k)$	each Gaussian component with mean μ_k and covariance θ_k
$G_x(x, y)$,$G_y(x, y)$	Horizontal and vertical gradients

N	Number of local units of differences
R_f,G_f	Features extracted from referenced images and generated images respectively.
$H(x, y)$	The 2-order differential value at pixel $[x, y]$
$M(x, y)$	Magnitude of 2-order differential value at pixel $[x, y]$

Contents

List of Acronyms ix

Nomenclature xi

List of Figures xvii

List of Tables xix

1 Introduction 1
- 1.1 Background . . . 2
- 1.2 Motivations . . . 3
- 1.3 Research Questions . . . 4
- 1.4 Scope and Limitations . . . 5
- 1.5 Clarification of the issue . . . 5
  - 1.5.1 Painting techniques . . . 5
  - 1.5.2 Strokes input . . . 6
  - 1.5.3 Baseline models . . . 6
- 1.6 Research Methodology . . . 6
- 1.7 Sustainability and Ethics . . . 7
- 1.8 Contributions . . . 7
- 1.9 Structure of the thesis . . . 8

2 Theory 9
- 2.1 Image Segmentation . . . 9
  - 2.1.1 Art painting segmentation . . . 9
  - 2.1.2 Using domain adaption in segmentation . . . 10
- 2.2 Stroke-based Painting . . . 11
- 2.3 Art Perception in Painting Process . . . 11
- 2.4 Color Space of Images . . . 12

3 Methodology 13
- 3.1 Research-through-design method . . . 13
  - 3.1.1 Stroke Estimation Using Machine Learning . . . 14
  - 3.1.2 Searching Strategy for Brush Size . . . 15
- 3.2 Platforms and Tools . . . 15
- 3.3 Dataset . . . 16

4 Experiments **19**
4.1 Ablation Study: Segmentation techniques on art paintings 19
4.1.1 K-means clustering on HSV/RGB images 20
4.1.2 DeepLabV3 22
4.1.3 DeepLabV3 with domain adaption 23
4.2 Brush Stroke Models 25
4.2.1 Circle Brush Strokes 26
4.2.2 Template-based Brush Strokes 28
4.3 Hyper-parameters for the training process 28
4.4 Generate Brush Stroke Centroids 30
4.4.1 GMM-based Density Estimation 30
4.4.2 Gradient-based Density Map 31
4.5 Estimate Parameters of Brush Strokes 32
4.5.1 Rotation Angle (θ) Estimation 32
4.5.2 Lengths ($L1, L2, W1, W2$) Estimation 33
4.6 Render Strokes 33
4.7 Experiments I: "Details first" vs. "Background First" in painting process 34
4.7.1 Configurated pipelines 34
4.7.2 Complete "voids" 35
4.8 Experiments II: Iterative design for improvements 36
4.8.1 Application of neural networks 36
4.8.2 Explore on abstract paintings 36

5 Results and Analysis **39**
5.1 Evaluations 39
5.1.1 Quantative Evaluation 39
5.1.2 Visual Analysis 40
5.2 Segmentation Results 40
5.2.1 K-means clustering 40
5.2.2 DeepLabV3 42
5.2.3 Using Domain Adaption 43
5.3 Density Map Estimations 44
5.4 Sampling results given density maps 46
5.5 Stroke-based Painting Results 46
5.5.1 Procedural Painting Processes 47
5.5.2 Test results on abstract paintings 47

6 Conclusion **51**
6.1 Discussions on Sub-tasks 51
6.1.1 Segmentation techniques for art painting 51
6.1.2 Impacts of brush strokes 52
6.2 Limitations 52
6.2.1 Segmentation on art painting 52
6.2.2 Art perception 52
6.2.3 Absence of procedural information 53
6.3 Future work 54
6.3.1 Separate perception and rendering 54

6.3.2 Modifications for generating abstract paintings 54

Bibliography **57**

List of Figures

1.1 Painting process with the blocking-in first approach. 1
1.2 Painting process with Alla Prima approach. 2

3.1 Proposed painting pipeline. 14
3.2 Examples of art paintings in DRAM dataset. 18

4.1 Input RGB images for K-means clustering algorithms. 20
4.2 The visualization of value distributions for input images in RGB and HSV color space. 21
4.3 Segmentation results using RGB and HSV color space of images. . . . 21
4.4 The examples of input images for DeepLabV3 models. 23
4.5 Sample results for DeepLabV3 segmentation. 24
4.6 Illustrations of Stoke models. 26
4.7 Example of computing and applying brush strokes in pixel-wise canvas. 27
4.8 Generated sample stroke. 28

5.1 K-means clustering results of art paintings of animals ("a fox in the wild") in Realism, K=2. 43
5.2 Segmentation results of DeepLabV3 (ResNet_101) on paintings of multi-portaits. 45
5.3 Segmentation results of DeepLabV3 (ResNet 101) on paintings of landscapes. 46
5.4 Results of sampling stroke centroids with different values of p_max. 49
5.5 The painting process of a portrait example using rectangular brush strokes. 49
5.6 Painting process on an abstract painting. 50
5.7 Illustration of painting process of a portrait with *paint_mode* = "f-b". 50

List of Tables

3.1 The content details of DRAM dataset. 16

4.1 Hyperparameters for initial system design of painting process. 34

5.1 Sample test images for segmentation. 41
5.2 Extracted foreground by using K-means clustering ($K = 2$) method. . 42
5.3 Extracted foreground by using DeepLabV3(ResNet 101). 44
5.4 Extracted foreground by using DeepLabV3(ResNet101) with domain adaption. 47
5.5 Test results of GMM-based density estimation and gradient-based density estimation on 100 art paintings. 48

1 Introduction

Visual art practices involve all art forms that convey the ideas and emotions of artists in visual presentations. They are diverse depending on painting styles and the techniques preference of artists. Artists can employ various painting techniques while creating visual artworks, whether trained or out of nature. Unfortunately, the mainstream recognition of painting techniques varies greatly depending on factors such as categorization and art genres. Using stroke-based paintings, for instance, one existing definition of painting techniques is to divide them into seven categories based on the order in which the regions of interest move during the art-creating process[1]. However, other definitions of painting techniques are possible. For instance, painting techniques can involve different brush stroke appearances, including shapes and textures.

The two painting processes shown below demonstrate the different pipelines of creating artworks with blocking-in and Alla Prima techniques in **Figure 1.1** [2] and **Figure 1.2** [3], respectively. With the blocking-in approach, artists choose dominant colors and loosely paint them first on the whole canvas, so the painting process is segmented given the color and tone blocks. Usually, the background and foreground items are separated. While in Alla Prima, which is a quick painting technique, artists paint the background and foreground all at once, and they do not contour the lines and edges as blocking-in but lay every stroke in the same colors.

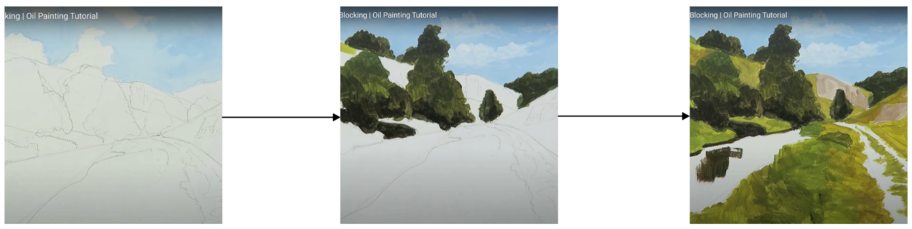

king | Oil Painting Tutorial
Blocking | Oil Painting Tutorial
Blocking | Oil Painting Tutorial

Figure 1.1: Painting process with the blocking-in first approach.

Generative art [1][2] [3] has been prominent in various popular culture industries such as film-making, music, games, robotics, and the Net Art [4]. As an ever-diversifying art practice that combines creative ideas with autonomous systems,

[1]The description of these painting techniques from www.liveabout.com
[2]Figures were captured from Youtube video by Michael James Smith
[3]Figures were captured from Youtube video by Florent Farges - arts

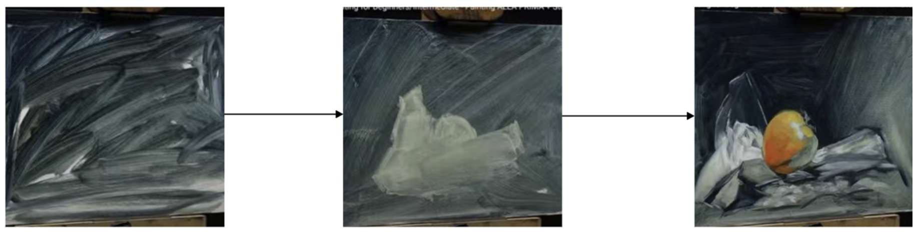


Figure 1.2: Painting process with Alla Prima approach.

generative art fosters discussions in modern aesthetics and art theory. The prosperity of AI technologies has specifically shed light on the ideas that use machine learning to create digital artworks. For example, this year, an image made with Midjourney an artificial intelligence (AI) system that can produce detailed images when fed written prompts based on DALLE, won first prize at the Colorado State Fair Fine Arts Competition [5]. More explorations of AI art systems inspire research on combining generative art with natural visual art practices using machine learning, including AARON [6] by Harold Cohen early in 1973, and recent initiatives such as DeepDreams [7] by engineers in Google, StyleGAN [8] by OpenAI and Stable Diffusion.

1.1 Background

Among various forms of artistic presentation, visual arts have become a mainstream with their wide range of application scenarios in industry, including the paradigmatic forms such as painting and other forms related to mass media such as film-making,dance,photography, etc. Differences in aesthetic perceptions and technical preferences among artists contribute to the diversity of visual artworks. In traditional paintings, for example, the preferences of materials such as canvas,wood,porcelain and tools ranging from pigments,watercolor to charcoal,etc.,which can be seen as 'techniques' in creation of art,can produce unique aesthetics. Apart from that, the artist's actions also have an impact in the aesthetics of resulting artworks, such as the orders and directions of lines in sketching, the distribution and blending of pigments in watercolors.

Recently, visual art practices have more flexibility of combining artists' creativity with computer technologies. Whereas comprehension and appreciation of art are still considered to be exclusively human capability, recent advances in computer technologies in terms of machine learning, computer graphics and digital image processing enabled computational analysis of artworks using AI technologies. An ncreasing number of large-scale digitalized artwork collections gives opportunities to analyse the oeuvres and history of visual art. Particularly, convolutional neural networks(CNNs) boosted the machine intelligence in large-scale classifying and syntheically understanding artworks. On the other hand,these online available datasets of artworks leverage more applications that produce creative practices using AI technologies. In the past decades, research on generative models especially generative adversial networks(GANs) inspired various explorations of making machines generate images given a set of visual art examples.

Painting as a visual art form predominately presented in two-dimensional space clearly serves as a breakthrough in an attempt to combine AI technologies and artistic creation. And because of better computers that allow computing on large datasets of images in super resolutions, a lot of researchers proposed to make intelligent machines create artworks. A remarkable milestone among all these efforts is style transfer using CNN frameworks, which was first introduced in 2016 by Leon A Gatys et al. [9], aiming to generate stylized images by separating the content and style of images. This initiative triggered growing interest in style transfer models and stylized painting methods. Subsequent discussions in this field stand in three categories——stroke-based rendering, image analogy, and image filtering, in terms of the technological frameworks.

Another field of creating novel visual content using AI technologies is to generate images from text, in which emerged not only the GAN-based architectures such as StackGAN [10],but also multi-modal applications based on transformer frameworks, which was first introduced as natural language models solving tasks such as text classification and machine translation [11],has been successfully used for text-to-image generation [12, 13].One state-of-the-art transformer-based model that outperforms previous GAN-based models is CogView [14],which combines transformers with the framework of Vector Quantized Variational AutoEncoders(VQ-VAE) [15] to improve large-scale image generations from text in terms of reducing fidelity loses [14]. In 2022,another most remarkable AI inventions on image generations is Stable-Diffusion [16] based on Latent Diffusion Models(LDMs)that achieved competitive performances on image generation and text-to-image synthesis by applying diffusion models in a latent space of powerful pretrained autoencoders [17].

Although the growing attempts of combining AI technologies and visual arts have enriched approaches to understand and appreciate visual artworks [18], and created efficient ways to generate artworks given textual or audio description, the details of creative process in terms of how painting techniques play roles in creating novel artworks using AI techniques are still enigmatic. Obviously we can appreciate AI-generated visual artworks by evaluating the semantic synthesis between input information and generated results, and analyse the history and context of artworks in a computational manner. Painting techniques used in art creative processes however is necessary for analyse the dynamics of AI-generated visual arts,especially in context of explaining the novelty of AI-generated artworks.

1.2 Motivations

As initiatives of using AI art in industry grows, such as online exhibitions and interactive visual arts, AI technologies especially deep learning with domain-specified neural networks have empowered artists to create artworks in novel forms. However, most attention of researchers,artists and viewers was centered on the resulting visual presentation of AI art, it is far from sufficient to explore how the knowledge in visual arts practices can be integrated into artistic creation processes using AI technologies [19]. Therefore, this project attempts to explore how the knowledge in visual arts practices can be integrated into artistic creation processes using AI technologies. A promising outlook is to practically enrich the context of image-to-text research. For example, recent work in image-to-text synthesis mostly concern on generating semantic description of the scenes, our work can foster following research in generating aesthetic remarks for artworks. Once AI machines gain the capability of aesthetically analysing visual artworks, it is possible to make full use of the advances of machines in terms of large-scale computing to create novel techniques for visual art creation.

The project explores how painting techniques can a play role in the architecture design of AI art technologies. Specifically, we propose generative painting systems that are customized for different kinds of painting techniques and their final artworks given some content images. Since the painting techniques are usually presented in textual way, the system should be constructed on a multi-modal basis that can tackle problems such as extracting painting techniques from text, generating stroke-based images given a set of content images.

The desired outcome is a multi-modal architecture that takes textual description of painting techniques and a set of referenced context images as input and generates stroke-based artworks,which can illustrate the impact of different painting techniques on artistic creation. The system are expected to be applicable to different painting techniques and bring insights for visual arts creation.

1.3 Research Questions

The project aims to inform the generative system design with the knowledge of the artistic process and machine learning technology. The research question is therefore how to design a procedural painting pipeline that can integrate painting techniques from artists. Sub-tasks related to evaluations and generalization issues should be solved in the final stage of designing a generative painting pipeline. One sub-task in this research question is how we evaluate the quality of the generated paintings. Specifically, we decide on the way of quantitative analysis of the generated paintings as a complement to visual analysis. For generalization purposes, the generative painting pipeline was designed for specific painting practices. However, it aims to characterize any typical style of artwork and different painting techniques. So the research involves a discussion of possible improvements for generalization.

Specific principles and requirements were clarified within the research question. Since the procedural information of creating art paintings by artists is unavailable from a dataset of images, the way of translating procedural painting techniques into computative modules in the design of the generative painting pipeline is diverse. Thus it should be clarified with fundamental assumptions in the design. During designing the painting pipeline, specific models and techniques in machine learning that are effective for modular functions should be investigated, and the motivations for selecting these machine learning techniques should be explained. For instance, We compare the performance of the generative system with approaches of neural networks and parameter optimizations.

1.4 Scope and Limitations

Recent advances in machine learning have led to interdisciplinary artists and researchers employing various machine learning techniques in the artistic creation of visual arts, especially in style transfer [20, 21, 22]. However, the problem of integrating the painting techniques of an artist with the machine learning models is less addressed [23]. Therefore, one delimitation is that few principles of designing a generative painting pipeline involve the natural art-creating processes.

Another delimitation is defining the painting techniques that can be used to integrate the artistic ideas and methods in easel painting processes with machine learning models. Painting techniques, as manual and procedural elements in painting processes, are usually diversely defined regarding material choices, stroke textures, and the proceeding orders of items in paintings. The painting techniques in use mainly contained procedural techniques involved in creating processes of art paintings since the generative process is highly modular and flow-oriented [24]. We assumed that all painting processes focus on one canvas region at a time slot so that they can be simulated by a sequence of foreground patches extracted from image segmentations. For example, the orders of object placements in a painting expresses artists' idea of procedural painting techniques such as "blocking in", in which artists place the main edges of colors and shapes [25], and "detail first "(to start with the details on the foreground objects) or "underpainting"(to overlay the colors from background to foreground items) [26]. Moreover, various brush stroke templates were used to represent other painting techniques regarding the material selection and crossing out the underlying colors.

The research scope was limited to designing a procedural generative pipeline focusing on stroke-based paintings. Unlike previous approaches in style transfer, which were deployed with the complete picture of artworks and aimed to achieve good resulting artworks [27], our system is designed in a stroke-based method with segments taken from the translation of painting techniques. By estimating the position, shape, and color of each stroke sequentially, it presented the resulting art paintings and the processes that imitate the actual painting practices.

1.5 Clarification of the issue

1.5.1 Painting techniques

To implement painting techniques in a generative process, we translate procedural painting techniques and represent them as different painting workflows. Specifically, these workflows contain three groups of parameters that affect the semantic segmentation module:(1) "blocking in":in which the image segmentation module first captures edges between foreground and background color shapes and then the overlay different strokes;(2) "Detail first": in which the segmentation module aims to extract foreground items iteratively;(3)"Detail last": the segmentation module output background first given a set of digitalized artworks.

1.5.2 Strokes input

To learn the painting techniques in art creation, pre-defined vector graphics of strokes need to be used as input to the training process. Strokes are stored in vector graphics rather than raster images because the predictions of strokes are based on changing the colors and rescaling the shapes, vector graphics with the property of storing pixel information as points in coordinate system, win over raster images such as JPG and PNG on the convenience of rescaling to any resolution without pixelation [28, 29].

1.5.3 Baseline models

The workflow of conducting experiments in this research contains three main phases:

- Construct flow-oriented representation for different procedural painting techniques;
- Implement semantic segmentation networks on digitalized artworks;
- Generate strokes based on optimizing the shapes and colors of pre-defined strokes.

Baseline models for segmentation and generative processes in the second and third phases were needed. For semantic segmentation, a competitive pre-trained model that extracts the foreground objects can be implemented first. Some state-of-the-art models were executed at this stage, such as K-means clustering and DeepLabV3 [30].

Many machine learning architectures can estimate the colors and shapes of future strokes based on the output of the segmentation module and stroke input. Brush stroke estimations were translated into parameter optimization problems. For the positions of brush strokes, we used sampling methods such as gradient-based sampling. And we used backbones in style transfer architecture such as DCGAN to estimate the shape parameters of brush strokes.

1.6 Research Methodology

This thesis employs the research-through-design(RtD) method, one first-person method that uses personal experience within the design process. Implementing experiments and system design with personal experience can present our relevance for designing and gaining in-depth knowledge on interacting with systems within the related fields [31].

The theory, methods, experiments, and evaluations are based on a proposed translation from visual art practices to the tasks within the field of machine learning.

Following the research-through-design methodology, the thesis work selects exemplary painting techniques and pipelines in actual art-creating processes and designs painting pipelines that use machine learning techniques to understand art-creating practices. The detailed design and motivations are presented in Chapter 4.

Experiments on different machine learning techniques, such as segmentation models, follow empirical research methodology. In Chapter 5, the thesis work analyses the result from each sub-module by comparing the outputs in different categories, such as "landscape" and "portrait" paintings or paintings in realism and abstractionism. In different phases of this thesis work, quantitative and observative evaluations are performed to help analyze the proposed generative painting pipeline on specific tasks.

1.7 Sustainability and Ethics

The main focus of this study is to investigate a process of generating art based on machine learning techniques that incorporate the ideas of human artists in their paintings. The process does not involve human subjects, so this study has no ethical approval issue.

However, it should be noted that the artworks used in the experiment are from publicly available digital art collections, and the use of these paintings does not require and is difficult to obtain permission from the creators. At the same time, the research in this paper focuses only on a representative selection of artworks, most typical paintings from Western art movements. It may lead bias since the research was finished on limited categories of widely available art styles and visual art forms, which were chosen based on the experiment's feasibility and data availability. A specific description of the data set is presented in Section 3.3.

Finally, there are a number of potential stakeholders for this study, including the creators of the artwork, the researchers who analyzed and designed the painting process, and the audience for the visual art done by the painting system. A widely discussed AI ethics issue is the copyright of AI-generated art; however, this is not discussed in this study. Rather, it acknowledges the subjectivity of the participants in the art perception process and attempts to mitigate it in the evaluation.

1.8 Contributions

The contributions of this thesis work are summarized below:

- The work proposes a machine learning pipeline which is built through analyzing painting techniques in conventional visual arts practices.
- The work compares different image segmentation methods for painting segmentation task, and discusses the results.
- The work provides know-how on how painting techniques can inform brush-based autonomous painting processes.

- The work provides an in-depth analysis of the proposed method, while discussing the limitations and proposing future research directions.

1.9 Structure of the thesis

The following content of the thesis is organized as follows:

Chapter 2 presents theories about image segmentation methods, visual art creation, painting techniques, and art perception. Chapter 3 presents the methodology of designing the experiments and algorithms related to the sub-tasks and evaluation methods. Chapter 4 presents the implementing details of experiments and related tools and framework upon which the experiments of this thesis work depend. Chapter 5 presents the results of the experiments. Validity analysis on each result and comparative analysis are presented in this chapter. The conclusions drawn from this thesis are presented in Chapter 6, and limitations, improvements, and future works are proposed.

2 Theory

2.1 Image Segmentation

Segmentation is often used as an image analysis method to analyze an image's local characteristics. The basic principle is to assign a single class to the pixels of an image and eventually combine the localities of the same class. Based on the different domains of the task, these segmented images are often derived from subject-specific datasets. The datasets used for image segmentation tasks tend to be real-world photographs, as such datasets facilitate data annotation for use in supervised learning-based image segmentation schemes. In addition, items in these photographs are rarely deformed, so segmentation models trained on previous datasets can be applied to similar task scenarios. In contrast, the segmentation of art paintings is richly varied depending on the art style because the art style tends to produce deformations on the target so that even the same semantic target will show different local shapes in different art paintings. Therefore, image segmentation based on art paintings is a non-trivial segmentation task.

2.1.1 Art painting segmentation

Art painting segmentation is the task of dividing an image of a painting into different regions or segments based on their visual characteristics or semantic meanings. This task aims to enable automated analysis and understanding of art paintings. This project uses art painting segmentation to analyze and represent different painting techniques. However, art painting segmentation is challenging due to the complexity and diversity of painting styles and techniques and the subjective and context-dependent nature of art interpretation. Different artists may use different painting styles, materials, and colors to convey different meanings and emotions, making it challenging to develop a general-purpose segmentation algorithm that works well for all paintings.

Computer vision techniques for art painting segmentation mainly use features such as gradient, intensity, and neighboring similarity of images. These techniques include classical image analysis, such as K-means clustering, Canny edge detection, and GrabCut, and even hybrid methods. Computer vision techniques theoretically suffer from limitations in art painting segmentation tasks because they ignore higher-level semantic information of images. In addition, these methods have a limitation on super-pixel and large-scale images. Especially the Grabcut algorithm requires that the user first marks out the foreground and background areas with rectangular boxes or strokes, etc., as an initial estimate for the algorithm. Therefore this is extremely inefficient in a large number of image segmentation tasks.

AI approaches can theoretically perform better than classical computer vision approaches because they combine various features such as pixel color, texture, and shape and edges. Specifically some machine learning methods such as K-means clustering and neural networks may have better performance in segmentation tasks. They have more accurate segmentation results, especially in distinguishing the foreground and background of the image. These machine learning methods mainly learn to map the input art paintings with segmentation masks. The mapping strategy is modeled as deep neural networks and trained on a large dataset of annotated paintings. It advances computer vision methods for dealing with large paintings and powerful computation capability for large datasets. However, it generally lacks annotated art paintings, especially for semantic segmentation of abstract paintings.

2.1.2 Using domain adaption in segmentation

Despite proposing possible segmentation methods in the art painting domain, segmentation using domain adaption from photographic images to art paintings is still a priority. Directly transferring the segmentation models trained in the photographic domain can hardly solve the problems in the art painting domain. The differences in colors, textures, and geometric features of content between photographs and art paintings lead to significant accuracy losses when directly using a semantic segmentation model trained on photographs and implemented in paintings that usually contain unrealistic motifs and geometric distortions.

Existing image segmentation methods are rarely based on art paintings but on photographs. A natural way to solve this problem is to propose an architecture that solves segmentation tasks in the art painting domain. However, the need for ground truth makes training a segmentation model based on artwork challenging. Even though art painting datasets are available for training machine learning models, they are rarely concerned with segmentation but classification. Therefore, valuable future work is to make art painting datasets for segmentation tasks. These datasets should contain ground truth for distinct artistic styles since different styles may cause specific transformations of the same realistic object in the painting. However, this project prioritizes using domain adaption in segmentation for time efficiency.

Style transfer and Gram matrices are key techniques that make domain adaption from the photograph-based segmentation models to art paintings possible. By style transfer, we can generate "pseudo paintings" given content images and style images. There are some feature mappings between the generated images and the content images, which are usually realistic photographys with pixel labels for segmentation tasks.

Along with the desire to use ground truth labels for content images, we want to be able to produce ground truth for each style of art painting for training, which depends on the mapping of that style. Gram matrices make this possible because they can be used to measure the mapping relationships for any of the art style transitions.

2.2 Stroke-based Painting

In digital painting, stroke-based painting involves applying brushstrokes or marks to an image to mimic traditional painting techniques. Stroke-based painting can create various artistic effects, from realistic simulations of traditional painting styles to more abstract and expressive images. It is particularly effective in creating textures or layering effects.

In applications that mimic the brush stroke patterns of human artists, stroke-based painting first requires analysis of the reference art paintings. Because brush strokes may vary in shape, size, and color and overlap or blend in complex ways, analysis using traditional computer vision techniques is challenging [32, 33]. However, recent advances in deep learning and image analysis techniques are beginning to make it possible to automatically segment and analyze stroke-based paintings, allowing for new insights into the artistic process and the underlying structures and patterns within the artwork. Existing research commonly employs a parametric approach to digital strokes to render their artistic effects in digital images, with these parameters often necessitating stroke shape, color, and position. This approach effectively makes digital painting controllable and automated, especially in recent studies that use deep learning techniques to analyze the textures, edges, and other features of brush strokes to provide new approaches to digital art creation, such as style transfer[27] based on stroke analysis.

2.3 Art Perception in Painting Process

Art perception is a complex process that involves both artists and viewers. Artists use various techniques, materials, and concepts to create visual art that conveys a specific message or emotion [34]. Viewers then interpret the artwork based on their experience, knowledge and cultural background [35]. In the painting process, the artist's artistic perception is often present in the initial stages of the painting, i.e., thinking about the ideas and concepts they want to express, the techniques of artistic presentation, etc. Artists express their ideas of artistic perception through a series of decisions, such as how to represent the subject matter, which materials to use, and how to apply these materials to the surface.

In this project, art perception is also presented in the process of designing the generative painting architecture and paralleling the viewer's perception. Because we work with information about the artist's complete work, rather than directly observing the actual painting process, our perception of art as viewers are subjective, including our art history and technical knowledge. Moreover, it suggests that diverse generative painting is possible.

2.4 Color Space of Images

The color space of an image refers to a mathematical model used to describe the color information of each pixel in an image. The representation of image color information and the number of components differs in different color spaces. The commonly used color spaces include RGB, HSV, Lab[36], CMYK[37, 38]. The commonly used color spaces in digital image processing include RGB, HSV, and Lab. For example, in image analysis, the digital input image is often in RGB space. In the subsequent processing, such as conversion to a grayscale map, the digital conversion is based on the three-channel values of RGB. Besides, in the HSV color space, the color of each pixel consists of the values of the three components of hue, saturation, and luminance. This color space can effectively solve tasks such as image processing and color analysis in computer vision. Since color analysis for artistic painting in this project is an essential step in the design generation of the painting process, both RGB and HSV color spaces will be used.

3 Methodology

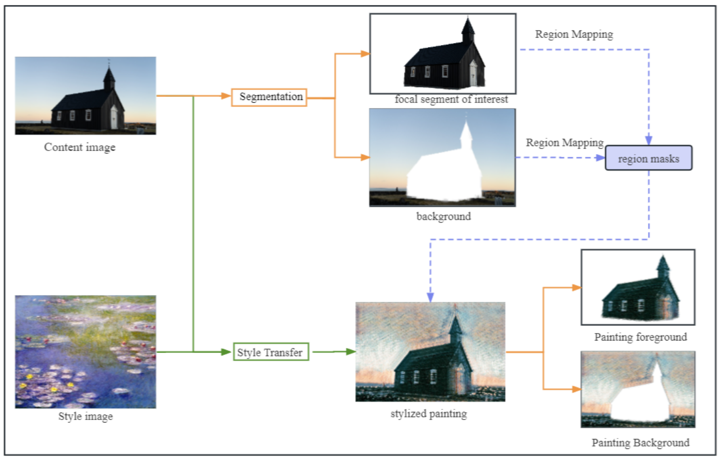
This thesis employs a research-through-design method and first-person method that uses personal experience within the design process. Implementing experiments and system design with personal experience can present our relevance for designing and gaining in-depth knowledge on interacting with systems within the related fields [31].

3.1 Research-through-design method

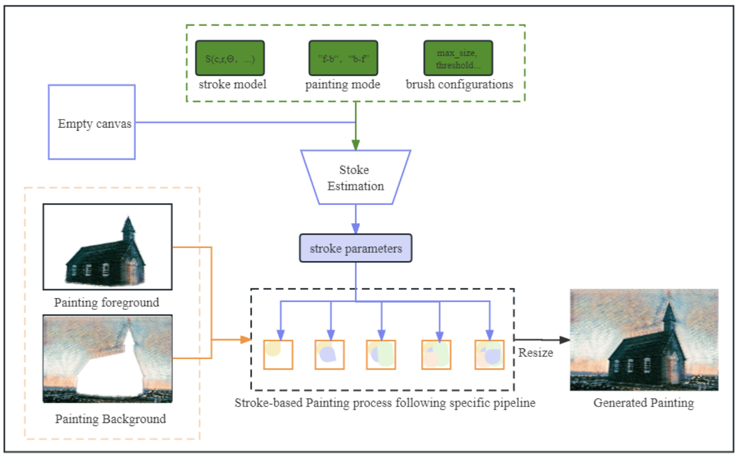
Research-through-design is a research methodology that combines design and research activities to create new knowledge, concepts, and artifacts [39, 40]. This approach is advantageous in fields such as architecture, industrial design, and human-computer interaction (HCI), where designers often encounter complex problems that require innovative solutions. Its operational process involves a cyclical process of design, testing, and evaluation to explore and optimize feasible solutions [39]. The specific process are concluded as the following four main steps:

- **Define the research question or problem**: The first step is to identify the research question or problem, which reviews existing research to find differences or gain insights.
- **Design**: The design phase involves the creation of a new artifact, concept, or prototype. Designers may use various design tools, techniques, and methods, such as sketching, modeling, or rapid prototyping.
- **Test**: After the design is completed, the design results are tested to identify any design flaws or areas for improvement and to provide feedback for design improvement.
- **Evaluate**: The final step is to evaluate the results of the RtD process, including analyzing data from user studies, soliciting feedback from stakeholders, or conducting a comparative analysis of the previous design with existing solutions.

The main tasks in this project are categorized into three modules: the translation of painting techniques from actual painting practice to structured inputs, the generative models and the optimization methods, which can be represented as the structure in **Figure 3.1**. Despite various approaches to translating painting techniques into structured inputs, such as simply encoding a set of painting techniques into different classes or using text summarization [41] to extract tags from a text description of painting techniques, the practical idea is to implement semantic segmentation to analyze the procedural painting techniques in creating artworks. The segmentation module in **Figure 1(a)** extracts foreground objects and background separately, which can be used in estimating stroke sequences. However, exploring other painting techniques related to materials and tools is possible to make the generated images stylized. In the generative painting module, a GAN-based architecture will be implemented to generate artistic images from low resolutions to super resolutions. Although the basic GANs can learn how to paint with real artworks taken as inputs, generating fine art in super-resolution is always challenging. So this module can be implemented with low resolution from scratch. For the optimization method, we can take pixel l_1 or l_2 loss between the canvas with several strokes and the target images.



(a) Stage 1: Art perception process



(b) Stage 2: procedural painting using painting techniques

Figure 3.1: Proposed painting pipeline.

3.1.1 Stroke Estimation Using Machine Learning

Once the image segmentation module extracts patterns that affect the stroke optimization, the generative module G will generate strokes from an empty canvas C_0 step by step. So position estimations of each stroke are defined as an optimizing process that minimizes the difference between the progressively generated paintings C_t and the reference artworks $\tilde{C}$. The update of canvas state follows a weighted sum of generated strokes s_t at time step t and previous canvas state C_{t-1}.

$$C_t = w_{ts} * s_t + w_{tc} * C_{t-1}, t \in 1, 2, 3, ... \tag{3.1}$$

$$w_{ts} + w_{tc} = 1 \tag{3.2}$$

for any $t \geq 1$, where w_{ts} and w_{tc} are weights for strokes and previous canvas state.

The loss function measures the similarity between the rendered and the reference images. Previous research on style transfer and stroke-based paintings proposed some ideas in the structure of loss function. The inspiration is that we should construct a loss function with consideration of pixel loss and stroke loss[42]. In this project, the loss in segmentation was also considered as a complement. We can do this in a gradient descent manner that updates the canvas as follows:

$$X_{t+1} = X_t - \mu \frac{\partial L(\tilde{C}, C_t)}{\partial X} \tag{3.3}$$

where X_t is the vector of parameters estimated in the t^{th} iteration, μ is the learning rate.

3.1.2 Searching Strategy for Brush Size

One goal of the painting pipeline is to find the optimal solution for the brush stroke placement, including the location and size of each brush stroke. A basic principle is that a larger brush size contributes to a faster painting process while it excludes more details. Oppositely, smaller brush sizes can be used to emphasize details, but the painting process can be extremely long. This principle is in line with the painting processes of human artists it therefore can be used in pipeline design.

The brush stroke can be transformed based on different shapes. In this project, the basic shape of a stroke is defined as a circle with a variable radius R. The transformation between the basic shape and stroke shape is based on the spline curve or straight line.

Regardless of the stroke model, the search strategy for stroke size can be elaborated as an exploration-implementation approach. we assume that the initial stroke size is the size of the reference image and guide the stroke size by iteratively shrinking the strokes and comparing the differences in their generated images; after the stroke size is reduced to a reasonable threshold, we assume that smaller stroke sizes will all be applied to the drawing process. Therefore, in the implementation phase, this project can use decreased brush sizes incrementally.

3.2 Platforms and Tools

The system is programmed in Python and based on standard frameworks such as PyTorch and OpenCV. Cloud computing resources and GPU platforms on Swedish

National Infrastructure for Computing(SNIC) were used for training models on large datasets.

For tasks in each module, previous research has provided a wide range of approaches to present excellent performance, so starting with pre-trained models and implementing transfer learning can be effective strategies for this project. In the segmentation module, although machine learning approaches based on pixel classification can be used, methods based on neural networks such as U-Net [43], ResNet [44, 45] and DeepLab [30], can be better options since the datasets used in this project consist of a large number of artworks that can be difficult to classify the content. Therefore, we selected segmentation models based on K-means clustering and DeepLabV3 and compared them with domain adaption approaches. And for the generative painting processes in this system, GAN-based architectures such as DCGANs [46] were used as an improvement of rendering modules.

3.3 Dataset

The research requires three datesets: digitalized artworks for training semantic segmentation networks, stroke samples, and reference paintings for generative models. Previous research on creating art with Ahas provided various artwork datasets. Ideally, the dataset used for this study should cover as many art painting genres and styles as possible to obtain generalizability. In addition, the dataset should be applied to the subtasks of image segmentation and brushstroke rendering. However, existing datasets of art paintings, such as WikiArt dataset, are mainly designed for tasks such as classifications by artists, styles, or genres [18]. They may not be designed explicitly for segmentation tasks due to lack of pixel-level annotations. Therefore, additional manual labeling or annotations may be required for specific segmentation tasks.

The research takes *Diverse Realism in Art Movements* (DRAM) dataset as the preliminary dataset of art paintings used in each module since it can be used to solve challenges in semantic segmentation on art paintings[47]. The DRAM dataset comprises four leading art movements: Realism, Impressionism, Post-Impressionism, and Expressionism. As a target domain of art paintings, it has an unlabeled training set and a fully annotated test set. The test annotations follow the guidelines of PASCAL VOC2012, which serves as the source dataset in domain adaption. Therefore, DRAM is an appropriate dataset for this project.

Table 3.1: The content details of DRAM dataset.

Art Movement	Expressionism	Impressionism	Post_impressionism	Realism
Number of Images	1603	1538	1462	1074
Average Size*	[398,500]	[500, 376]	[500, 425]	[500,354]

* The average size of each subset is the pixel number in the [width, height] format and each number represents the pixel numbers.

A standard set of characteristics, such as a particular style, content matter, or painting technique, often defines art movements. They can therefore serve as a description of a wide range of artistic styles and techniques. **Table 3.1** shows that the DRAM dataset contains about 5500 digital art paintings, with a maximum size of 500*500 pixels. **Figure 3.2** shows examples of works from four different art movements to show their differences. Since these original art paintings were not annotated with images for the segmentation task, they were used as templates for different art styles in the segmentation task. PASCAL VOC12 [48] is a dataset commonly used for semantic segmentation tasks and has 20 classes, including person, bird, bottle, dog, etc. The processed DRAM dataset contains model-created art paintings following the four art movements, in which the contents are from the photographic images in PASCAL VOC12. Therefore, the created art paintings are mapped with the corresponding content images, which are annotated for training segmentation models.

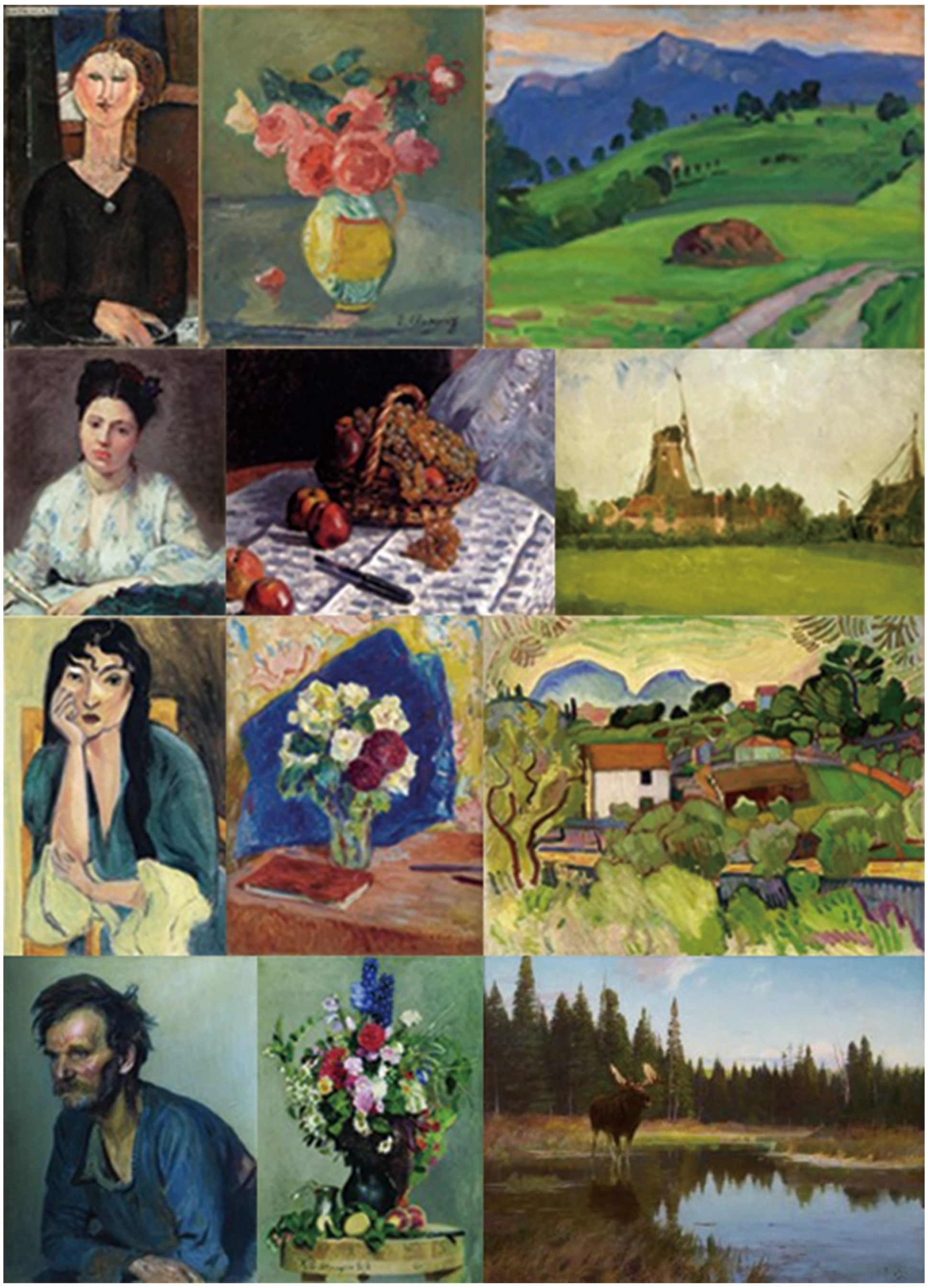


Figure 3.2: Examples of art paintings in DRAM dataset.

4
Experiments

Experimentation in this thesis work is divided into four phases following the workflow of finishing a design of a generative painting pipeline with painting techniques. The four phases in this section involve:

- Ablation study on different segmentation methods in Section 4.1;
- Experiment design for sub-tasks in brush stroke estimations in Section 4.2 - 4.6;
- Experiment on the first painting pipeline in Section 4.7;
- Iterative design for improvements in Section 4.8.

4.1 Ablation Study: Segmentation techniques on art paintings

In this section, we conducted an ablation study on different segmentation methods for art paintings using the DRAM dataset. Generally, image segmentation methods are based on conventional image processing algorithms and semantic segmentation models. Training a semantic segmentation model is a supervised learning process. Therefore, it requires a large number of pixel labels for each class. However, previous research on art paintings and machine learning dominantly focuses on tasks such as image classifications and style transfer, so existing datasets of art paintings with pixel labels for segmentation tasks are limited. However, conventional image processing techniques for segmentation are primarily based on paintings' colors and texture features, and they are usually trained without any label of pixels. There are different ways to present segmentation results in Python using standard libraries such as OpenCV and Pillow.

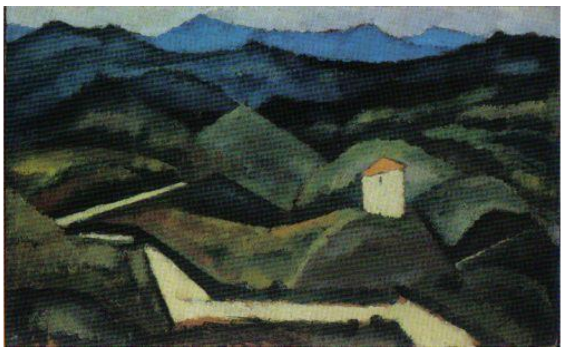
We experiment with the effect of three different segmentation methods based on unsupervised learning, supervised learning, and domain adaptation in the segmentation of art paintings, respectively. In machine learning, k-means clustering is the typical unsupervised learning that can be used for segmentation tasks. Moreover, we can choose DeepLabV3 as the deep learning model for segmentation tasks. At last, we investigated the domain adaption using DeepLabV3 for the same task, which is similar to the research of Nadav Cohen et al.[47]. The detailed implementations are described in Sections 4.1.1,4.1.2 and 4.1.3.

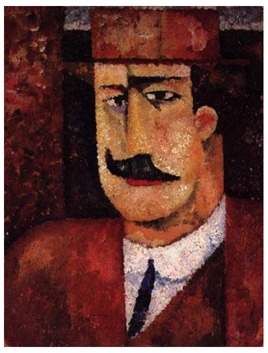
4.1.1 K-means clustering on HSV/RGB images

In this experiment, two problems were explored:

- Is K-Means clustering feasible for the front and back segmentation of art paintings?
- Which is more reasonable, K-means clustering based on RGB or HSV image space?

Since in this experiment, we are mainly concerned with extracting the region of interest ('foreground') and the remaining part ("background") in art paintings, the number of clusters k is set to 2 when applying k_Means clustering. Moreover, the RGB color space of images considers different visual information from HSV color space when we implement K-Means clustering. Thus they impact the complexity of this problem given the value distributions.


(a) landscape


(b) portrait

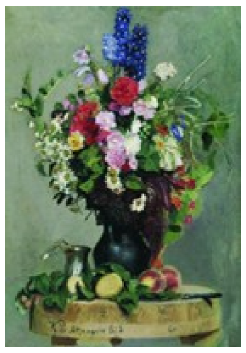
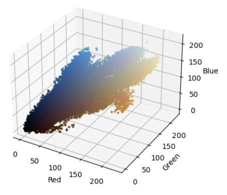

(c) potted flowers

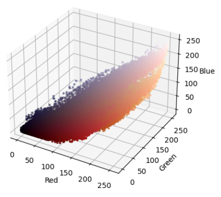
Figure 4.1: Input RGB images for K-means clustering algorithms.

This experimental process consists of three main steps: image pre-processing and feature analysis, foreground segmentation based on K-means clustering, and visualization of the results. In the image pre-processing step, whether or not the input image is converted to an HSV image is critical in comparing the RGB image with the HSV image. The image pre-processing stage also involves vectorizing the image, i.e., converting the two-dimensional data into a vector of pixel values. Using the images shown in Figure **4.1** as examples, we first visualize the distribution of values in the RGB and HSV image spaces, as shown in Figure **4.2**.

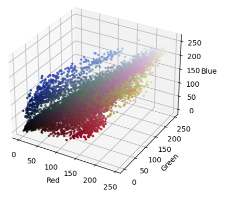
Figure **4.2** shows that there is coupling in the distribution of RGB values of the above three input images, while there is a noticeable aggregation feature in the distribution of HSV. For example, **Figure 4.2 (a),(b),(c)** distributions show obvious aggregation points. In addition, the distribution characteristics of HSV images are correlated with the color distributions of the input images. Since the colors of these scatter plots are derived from the color statistics of the input image, we can see from the visualization that the peak distribution of the HSV image has an obvious correlation with the color distribution. For example, in **Figure 4.1(b)**, the orange color is distributed in the facial part of the original image, while the red color is mainly distributed in the background region. The different distributions of these two color



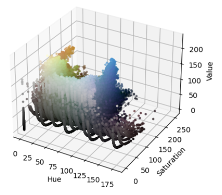
(a) landscape-RGB



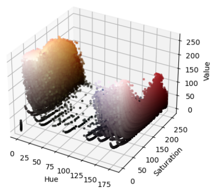
(b) portrait-RGB



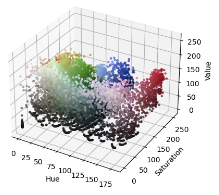
(c) potted flowers-RGB



(d) landscape-HSV

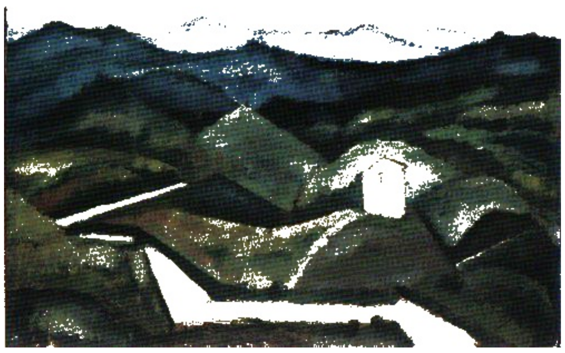


(e) portrait-HSV

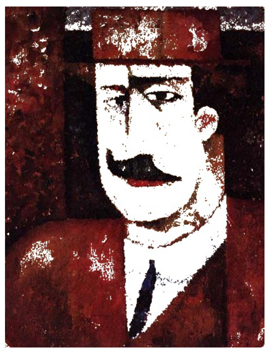


(f) potted flowers-HSV

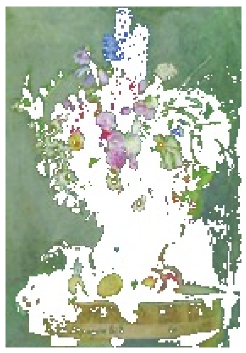
Figure 4.2: The visualization of value distributions for input images in RGB and HSV color space.



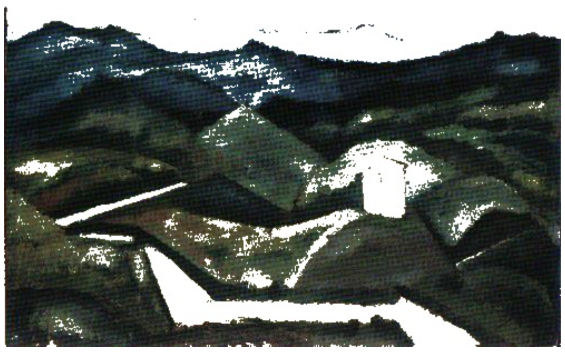
(a) landscape-RGB



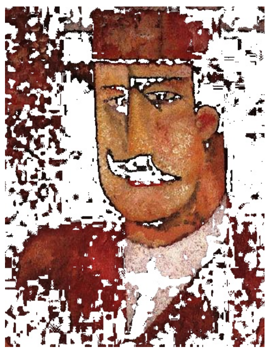
(b) portrait-RGB



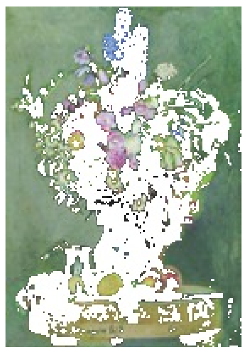
(c) flowers-RGB



(d) landscape-HSV



(e) portrait-HSV



(f) flowers-HSV

Figure 4.3: Segmentation results using RGB and HSV color space of images.

regions also correspond to the **Figure 4.2(e)** distribution in the HSV distribution. In contrast, in the RGB distribution, the distributions of red, green, and blue colors are often stacked to form new colors. Therefore, they do not have a clear correlation with the color distribution in the visual presentation. This also leads to the conclusion that the HSV color space is more suitable for the visual representation of images.

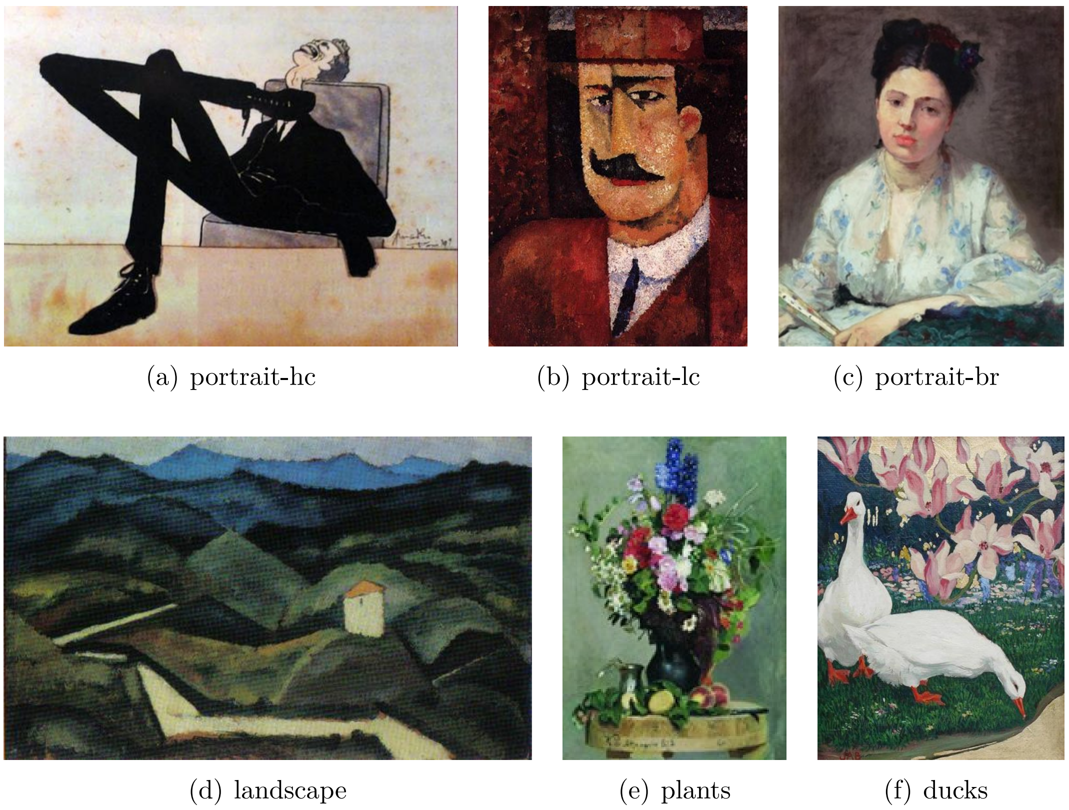
To implement K-means clustering, weused 'KMeans' module from "sklearn.cluster" in the Python environment with the parameter $K = 2$. This module generates the foreground mask by predicting binary labels for each pixel. This step grouped pixels with the same clustering characteristics into a feature class, foreground, or background. The resulting mask is defined as an array of the same size as the input image, with each element value corresponding to a pixel coordinate of the classification result (in the foreground/background segmentation, the foreground is marked as 1 and the background as 0). Finally, the generated foreground masks were placed onto the input images to illustrate results, which can be seen in **Figure 4.3**, where **(a)(b)(c)** is the segmentation result based on RGB color space and **(d)(e)(f)** is the segmentation result based on HSV color space.

4.1.2 DeepLabV3

DeepLabv3 is a Convolutional Neural Network (CNN) model created to address the semantic segmentation issue. It is an upgraded version of DeepLabv1 and v2 and is more effective than its predecessors. It employs the Atrous Convolution architecture, which is also known as "dilated convolutions", and an enhanced Atrous Spatial Pyramid Pooling (ASPP) module from an architectural perspective [30].
This experiment explores the possibility of using pre-trained DeepLabV3 models in PyTorch for inferencing segmentation on art paintings. PyTorch provides three pre-trained DeepLabv3 variants with three different backbone models, which are "deeplabv3_resnet50", "deeplabv3_resnet101" and "deeplabv3_mobilenet_v3_large". Specifically, DeepLabv3 models with ResNet backbone ('ResNet50' and 'ResNet101') have output_stride = 8, whereas DeeLabv3 models with mobilenet_v3_large backbone have output_stride = 16. The pre-trained models can be loaded by " torch.hub.load()" in PyTroch.

In order to compare the segmentation effect with the K-means clustering model in section 4.1.1, the input images in this experiment contain the images shown in **Figure 4.1**. We also introduced other input images to analyze the segmentation results in paintings with different artistic features and contents, as shown in Figure **4.4**. All three DeepLabV3 variants were implemented for segmentation. After inferencing, the predicted pixel labels can be used to generate the foreground and background masks, which were overlayed on the input images. The results are shown in **Figure 4.5**.

As shown in Figure **4.5**, the DeepLabV3 models with three different backbones can achieve foreground segmentation of art paintings to different degrees. In the segmentation results based on ResNet50, the main content information of the art image


(a) portrait-hc (b) portrait-lc (c) portrait-br

(d) landscape (e) plants (f) ducks

Figure 4.4: The examples of input images for DeepLabV3 models.

can be included in the foreground. In contrast, in the results of ResNet101, this information will be included less in the foreground. The model with mobileNet_v3 as the backbone tends to include less content in the foreground, but it can maximize the inclusion of local information other than the main content, as shown in Figure **4.5(d2),(d3),(d4)**.

In addition, for the input images shown in Figures **4.4(d)** and **4.4(e)**, the DeepLabV3 model shows significant limitations in the foreground segmentation task. Since the pre-training of this model is based on the PASCAL VOC dataset, which contains only 20 classes of photographic images, and **Figure 4.4(d)** does not contain any class of semantic subjects, it is wholly classified as background. For **Figure 4.4(e)**, the semantic content is potted plants, and this one class exists in the PASCAL VOC dataset, so this model can extract local features and segment the foreground even though in the domain of artistic paintings, the artistic style causes differences between the images and the photos of natural potted plants. However, the ResNet50-based model classifies this image completely as a background, which is related to the structure of the model and will be discussed in Section 5.2.

4.1.3 DeepLabV3 with domain adaption

Section 2.3 describes the theory of applying domain adaption to segmenting artworks. In summary, this approach attempts to use existing segmentation methods

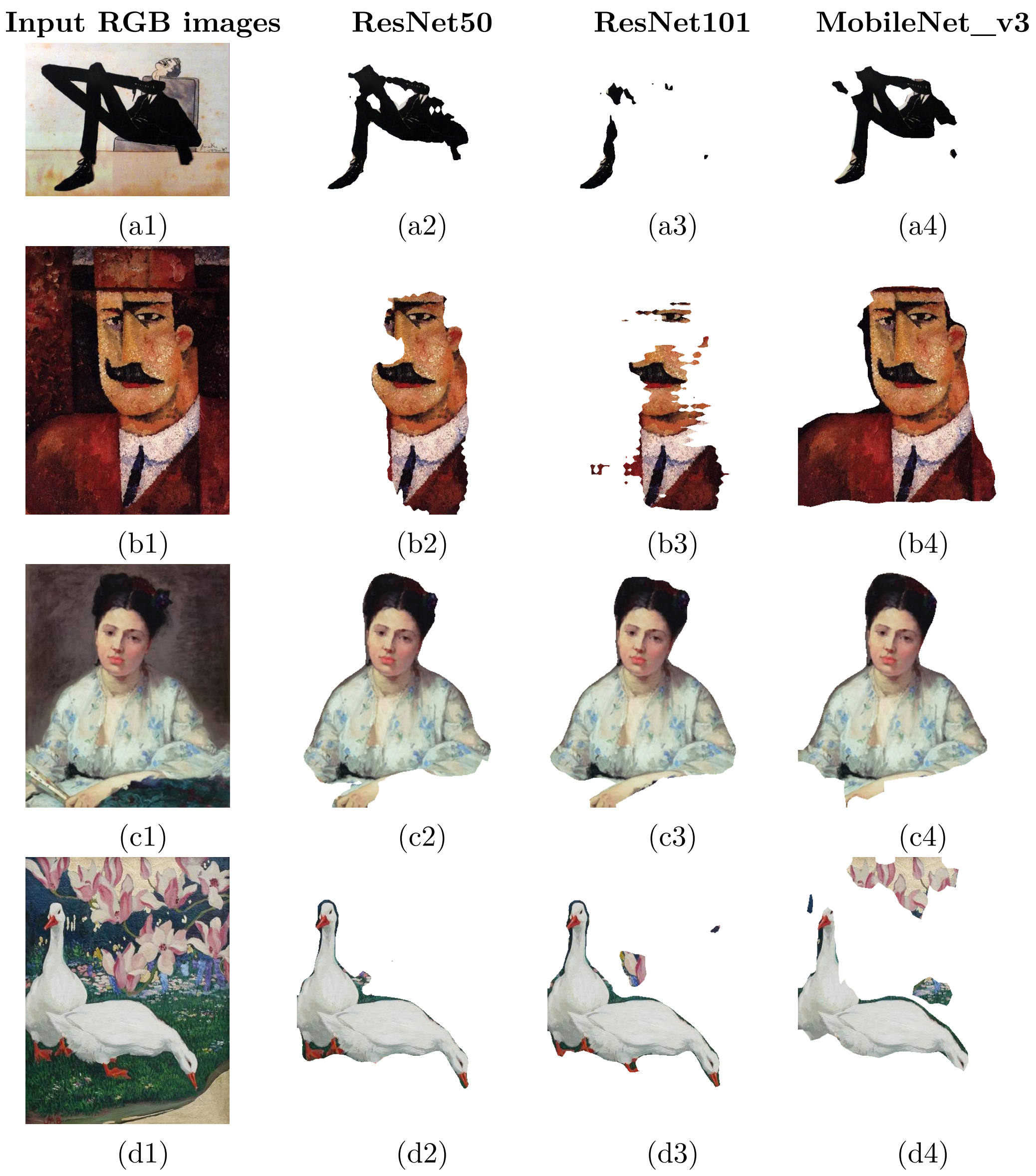


Figure 4.5: Sample results for DeepLabV3 segmentation.

for photographic images and existing augmented datasets to solve the problem of missing augmentation in the artwork domain. This section describes the details of implementing this approach in the Python environment.

The implementation of this method is divided into three main stages [47]:

- Use style transfer to generate pseudo-paintings with ground truth: In this step, an AdaIN model is trained to take the photographic images from PASCAL VOC and the art. The photographic images are content, and the paintings in DRAM are used as style images.
- Different styles of pseudo paintings are used to train the segmentation model: DeepLabV2 is used as the backbone of the segmentation model. Different styles of pseudo-paintings are used individually as input images to train the segmentation model under the corresponding style images. In this stage, the

ground truth of the pseudo painting is used as the ground truth of its corresponding content image.

- Mixing images from two sources to optimize the model: The model is finally trained to directly segment the art paintings in the target domain using photographic images from PASCAL VOC 12 as the input dataset and DRAM subdomains as the target.

4.2 Brush Stroke Models

Brush stroke patterns in art paintings refer to how the artist applies the brush to the canvas. Brush strokes vary in thickness, direction, texture, and color, and they can be used to create different effects, such as conveying movement, depth, or emotion. For example, the Impressionist movement is characterized by loose, spontaneous brushstrokes that capture the effects of changing light and atmosphere. In contrast, the Realist movement is known for precise, detailed brushstrokes that seek to represent the world accurately. In terms of individual brush strokes, the characteristics of the brush show marked differences depending on the artistic style and the techniques employed by the artist in the painting. For example, most of the brushes used by human artists in the artistic process are used for oil painting, where the consistency of the ink varies according to the strength of the artist's brushstrokes, thus creating a unique artistic expression early on. Moreover, different painting techniques such as rubbing, outlining, thwarting, tapping, dabbing, and brushing can manipulate the colors on the brush to present a unique distribution on the panel, resulting in light and dark colors and curvilinear effects of textures. Thus, brushstroke models can also be studied to understand different artists and art movements' techniques and styles.

Brush stroke models are the basis for generative artwork in conveying the regional patterns of art paintings. In generative painting, brushes are the small units of color and shape composition in a single-step painting, displayed in the final artistic effect. Such a brush can therefore be defined as an element containing information about color, shape, position, etc. Multiple brush elements with different property values form an artistic painting. On the other hand, a brush is also an essential element in the artistic analysis of a painting. Accordingly, in a generative painting, local features of the art painting can be extracted as a sequence of brushes with interrelationships. Therefore, in different generative painting models, the sequence of brushes that the same art painting has is not unique due to the different local feature extraction. Therefore, the definition of a brush model adapted to a specific painting system can be specific to a painting model. In this project, the colors of the brushes can be easily translated due to the reference to human painters' painting techniques, while the variable brush shapes need to be specified as a fixed pattern. Therefore, we summarize the basic shapes of brushes in the physical world in a generic way; they can be classified as round, linear, rectangular, and triangular, as shown in **Figure 4.6**.

In **Figure 4.6 (a)** , the circular brush stroke is the most common type in creating

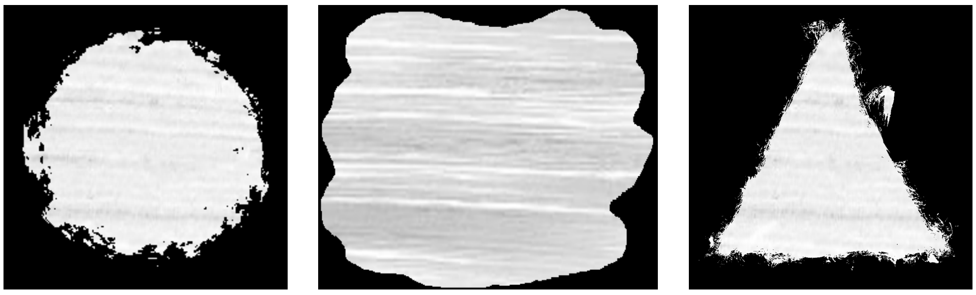

(a) circular brush stroke (b) rectangular brush stroke (c) triangular brush stroke

Figure 4.6: Illustrations of Stoke models.

paintings, widely found in Impressionist, Post-Impressionist, and Abstract Expressionist paintings. On the one hand, the circular brush can represent scenes and contents in paintings. For example, in Claude Monet's water lily paintings, circular brush strokes create the impression of flowers floating on the water's surface. On the other hand, it can also represent distinctive stylistic textures and the physical behavior of the painting process, as in Post-Impressionism, where artists such as Vincent van Gogh used circular brush strokes to create texture, movement, and depth in their paintings. Moreover, in Abstract Expressionism, artists like Jackson Pollock used circular brushstrokes to create complex and layered visual effects of dripping and splashing. Rectangular brush strokes (**Figure 4.6 (b)**) can be used to construct strokes with linear textures, which are often produced by the linear movement of the brush, especially when brushing or painting with techniques such as brushing and painting that often present a distinctly square brush. The triangular stroke (**Figure 4.6 (c)**) is less frequently used than the two strokes mentioned earlier. It appears in art paintings to describe natural forms, especially those containing a pointed tip, or to construct the movement characteristics of a picture, adding tension to it. For example, French painter Paul Cézanne used triangular brush strokes to suggest the form of a mountain peak or rocky landscape. Cézanne created the illusion of the rough, jagged surfaces of these natural forms by using a series of small, sharp, triangular brush strokes.

In some specific oil painting techniques such as thwarting, artists press the brush down and then apply a slight force, thwarting and lifting it, as in the calligraphic counter blade stroke. The difference in color dipping between the nib and the root of the brush and the different directions of lightness and weight of pressing the brush can often produce a variety of changes based on the triangle.

4.2.1 Circle Brush Strokes

A circular brush stroke can be defined with circle centers, radius, and colors. The center point of a circular stroke is the point within the pixel range of the reference image. For simplicity, when taken as an integer, the radius value ensures a quick calculation of the pixel points in the circular coverage and changes the color of those pixel points.

To implement circular brush strokes in the code, this project uses the function 'cv2::circle' for drawing circles in OpenCV-python. Given the documentation this method requires over 5 parameters, including referenced images, the center, the radius of the circle, color, and thickness. Significantly, the circle is filled when the thickness is negative and unfilled when the thickness is positive. Since the brush strokes are mainly filled with colors, the thickness is set as a negative float number such as '-1.0'.

The circle brush strokes are applied with the mapping of stroke shapes and colors. In pixel-wise images, the referenced art paintings are composed of pixels of colors. At any pixel in an art painting, the color can be changed when the stroke shape covers that pixel. Therefore, the stroke shape can be defined as a matrix of boolean values, which is of the same size as the referenced art painting. That is, for any pixel-wise referenced image ('canvas'),

$$C_{new} := C_{now} * (1 - S_{map}) + Color_{map} \tag{4.1}$$

Where C_{new} is the computed new canvas, C_{now} is the current canvas, S_{map} is the matrix of boolean values which represent to change the pixel value or not, and the $Color_{map}$ is the matrix of optimized pixel values. C_{new}, C_{now}, S_{map}, and $Color_{map}$ are of the same size as the referenced images. Therefore, the operation is an entirely element-wise operation.

Since the computed color $Color_{comp}$ is a matrix of the same size as the referenced image, it is necessary to set the colors as 0 at pixels that are not covered by stroke shape. That is,

$$Color_{map} := S_{map} * Color_{comp} \tag{4.2}$$

Figure 4.7 showcases an example of computing in updating a pixel-wise canvas. The canvas size was assumed as (4,4) shown in **Figure 4.7(a)**, and the central pixels are presented at (b) as current canvas C_{now}. Assuming the computed stroke map S_{map} and the color map $Color_{map}$ are 2*2 matrix shown in **Figure 4.7(b)**, the updated canvas C_{new} is the result of element-wise operations.

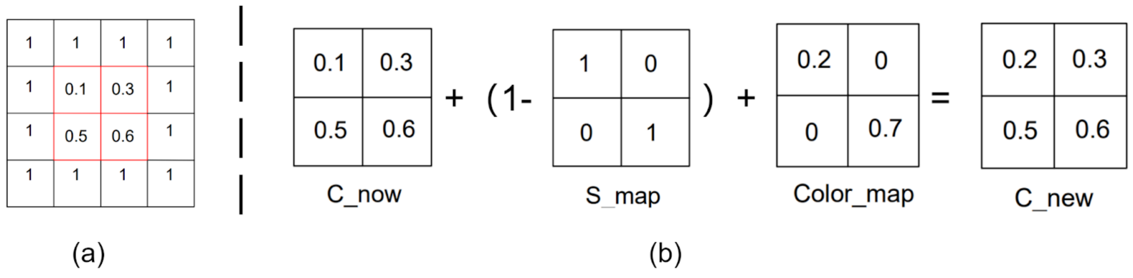


Figure 4.7: Example of computing and applying brush strokes in pixel-wise canvas.

4.2.2 Template-based Brush Strokes

The main brush shapes in the human artistic process are approximately rectangular or solid curved. Although the OpenCV module provides functions to generate circles and rectangles, these brush shapes, which consist of perfect baselines, need to fit better with the rich and varied brush shapes of the actual painting and have limitations in rendering brush color variations. Therefore, in this section, we explore a way to geometrically deform accurate stroke templates using the geometric transformation parameters of rectangular images. For the color processing, the brush colors are estimated with reference to the input image and filled into the transformed brush shapes. Unlike the previous section where the color is filled uniformly by setting the thickness, in this section, we obtain the color coverage of the brush by converting the reference image to HSV(Hue, Saturation, Value) color space. That is, the shape of the brush stroke

$$S_{shape} = f(center(x, y), width(w), length(l), rotation(\theta)) \tag{4.3}$$

The illustration of this brush stroke models are presented in **Figure 4.8**, where (a) is the brush template containing texture features; (b) is the rectangular stroke model; (c) is the generated stroke with the transformed shape and texture.

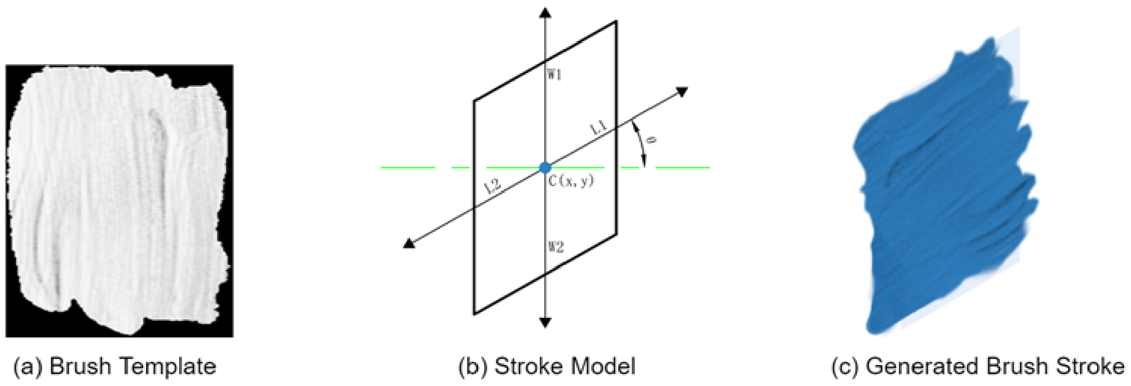


Figure 4.8: Generated sample stroke.

Centroids of brush strokes are generated from a sampling of pixels from the referenced image; the width w represents the distance between the centroid and the vertical edges of a rectangular brush stroke, while the length l represents that between the centroid and the horizontal edges.

4.3 Hyper-parameters for the training process

Hyperparameters are parameters that specify the external requirements of the painting process. Since in the procedural painting process (shown in **Figure 3.1 (b)**), only the resulting partial images are taken as inputs for the second stage of generating paintings, parameters in the first stage, which focus on the art perception, do not affect the painting process. So the hyperparameters in this section clarify external parameters that affect the painting process, including the brush stroke model, maximum brush size, brush sampling number, and paint mode. Except for maximum brush size and brush sampling number, which take positive integer values, the brush stroke model and paint mode take descriptive values, usually set in code as strings or pre-stored interpretable values. Specifically, the brush stoke model and maximum brush size cannot be decided independently because the brush sizes have dependencies on the brush stroke model. These dependencies are generally translated as the computation of areas of brush strokes given specific brush stroke models. For example, the area of a circular brush stroke is computed as πr^2, where r denotes the radius of the circular brush stroke. The brush stroke size has a positive monotonicity with the radius of the circular brush stroke. So the maximum brush stroke be used to define the thresholds for specific parameters of the brush stroke models.

On the other hand, the shape and size of the brushstrokes are closely related to the artistic effect formed and the speed of the painting process. The larger the brush stroke size, the more pixel points are manipulated by a single stroke, thus enabling a complete painting to be formed quickly. However, since parametric strokes are less diverse in their local representation than human-drawn strokes, inevitably, large strokes will lose their ability to emphasize details. Such strokes are more common in some abstract artworks. Conversely, if enough strokes are used, for example, by including only a one-pixel point in each stroke, we can make the canvas end up with a very close result to the reference image. However, such a painting strategy would take time and effort. An original reference image typically contains over 480,000 (800 * 600) to millions of pixels. Although we can cut or scale the image to make it smaller, we still need to gain reference information. On the other hand, such a painting strategy differs from how a human artist handles the painting process. Therefore, it is necessary to use a reasonable brush stroke size. Moreover, the stroke size should be controllable. In this experiment, we assume that the maximum stroke size is close to the size of the reference image. Since the stroke size is calculated from the shape parameters of the stroke, we need to consider the range of parameters when setting these parameters. The exploration of the shape parameters of the strokes will be elaborated in Section 4.5.

The maximum sampling number defines the number of brush strokes in each iteration. The sampling decides the possible brush stroke positions (called "centroids ") N_s. After generating the strokes, a simple placement strategy is to place them on the canvas in the order of the stored stroke sequence. However, to better show the painting process, the strokes can be placed in hierarchical order according to the stroke size, e.g., from largest to smallest stroke size; in addition, we can also iteratively place the strokes in batches, e.g., when $N_s = 5392$ (less than the total number of pixels), assuming the number of iterations $N_{iter} = 15$. Then, in each iteration, 340 strokes are randomly selected from N_s strokes to be painted until all strokes have been placed, completing the iteration.

4.4 Generate Brush Stroke Centroids

One of the primary roles of the reference image as an input to the painting system is to generate stroke centroids randomly during the drawing process based on the perception of their features with the same feature distribution. Therefore, it is necessary to use a suitable method to analyze the feature distribution of the image during the perception stage of the reference image. In image processing theory, a density map can be used as a representation of the image feature distribution. It is typically created by mapping the number of pixels or objects in a given area to a continuous grayscale or color value. Converting the image into a density map makes it easier to analyze and identify objects and patterns within the image. Density maps can be created using various techniques, such as histograms, heat maps, and kernel density estimation. The choice of technique depends on the specific needs of the image analysis task and the type of image data being processed.

This experiment compares the performance of kernel density estimation (KDE) and gradient-based density estimations in the context of images. KDE estimates the probability density map of images by computing the weighted contributions of pixels using kernel functions. Especially Using Gaussian Mixture Model(GMM) to estimate the probability density of images is a clique of the KDE method, as GMM can represent multi-modal distribution and estimate the density of high-dimensional data. GMM-based estimation is non-parametric because the mean and covariance of each Gaussian are estimated from the images, and the weighted sum of each Gaussian contributes to the overall probability density of the image. However, gradient-based density estimation for images can be more intuitive because it uses the heat kernel smoothing method, which involves solving the heat equation on the image data to estimate the density. The underlying intuition is that gradients of images can represent the intensity variations or edges. Techniques such as Sober operation or Canny edge detection can compute the gradients of images in programming.

The comparison is based on calculating the differences between the mean squared error (MSE) of input imagea and that of the estimated density map using the two methods. The MSE measures the average squared difference between the pixel values of the input image in grayscale and the density maps, to quantitatively estimate how much the images differ from each other. Lower MSE values correspond to better density estimation.

4.4.1 GMM-based Density Estimation

GMM-based density estimation for feature extraction maps intensity values to sampling probabilities. It is reasonable since the goal is t sample stroke anchors in the same distribution as the reference images. The stroke anchors cluster in higher-intensity regions on the images. This method assumes that the probability distribution can be modeled as the integration of a finite number of Gaussian models. That is,

$$P(x) = \sum_{k=1}^{K} w_k * N(x|\mu_k, \sigma_k) \tag{4.4}$$

where $P(x)$ is the probability density of the input image, K is the number of components, w_k is the weight for each component, and $N(x|\mu_k, \sigma_k)$ represents each Gaussian component with mean μ_k and covariance σ_k.

The parameters learning, namely the weights, means, and covariances of the Gaussian distributions, is typically estimated from the data using the maximum likelihood estimation (MLE) or maximum a posteriori (MAP) estimation methods. Once the parameters are estimated, the GMM can estimate the density of new data points or generate new samples from the same distribution.

In the technical implementation, we used the GMM module from scikit-learn with two hyperparameters: the number of Gaussian distributions ("n_components") and the kernel size for image blurring ("k_size"). For simplicity, we set $n\_components = 3$ and $k\_size = 3$. The main steps consist of 3 modules. The first step is to convert input images into grayscale since the probability density represents single values in pixels while the color images contain three channel values. The grayscale values are normalized into the range [0,1] for simplicity. The second step is to fit the given number of Gaussian models into image data, which estimates the mean and covariance of each Gaussian. Using the "score_samples" method of the GMM object can generate a grid of points covering the image to illustrate the density map of the input image.

4.4.2 Gradient-based Density Map

The gradient-based method can extract features of the reference image by computing the gradient magnitudes. Because gradients can represent the intensity differences in pixel levels, and larger gradients imply noticeable changes in intensity, it may be associated with the probability density of the centroid. Pixels with large gradients are more likely to compose edges, so gradient magnitudes are reasonable to compute the probability density map of images.

In digital image processing, the gradient magnitude of an image is approximated as a two-dimensional differential approximation of the neighboring pixel values. That is,

$$G_x(x, y) = H(x + 1, y) - H(x - 1, y) \tag{4.5}$$

$$G_y(xy) = H(x, y + 1) - H(x, y - 1) \tag{4.6}$$

where $G_x(x, y)$,$G_y(x, y)$ are the horizontal and vertical gradients respectively, and $H(x, y)$ denotes the 2-order differential value at pixel $[x, y]$. Thus the magnitude

$$M(x, y) = \sqrt{G_x(x, y)^2 + G_y(x, y)^2} \tag{4.7}$$

We normalized the gradient magnitudes to represent the probability density distribution. That is, for each pixel $[x, y]$

$$M(x, y) = \frac{M(x, y) - M_{min}}{M_{max} - M_{min}} \tag{4.8}$$

In programming, the pre-processing steps are similar to that in GMM-based density estimation, which is to load color images and change them into grayscale and rescale pixel values into the range [0,1]. The Sober operator from the OpenCV-python module can compute the gradients in both horizontal and vertical directions. So The code first loads the input image and converts it to grayscale. It then uses the Sobel operator to compute the gradient in the horizontal and vertical directions (G_x and G_y, respectively). It computes the magnitude of the gradient as the square root of the sum of the squared horizontal and vertical gradients. The last step is to present the resulting density maps as grayscale images.

Compared with GMM-based density estimation, this method ignores the data's statistical features but values the input image's local intensity, which is another way to present features (especially edges) of images using intensity.

4.5 Estimate Parameters of Brush Strokes

Since the shape of the brush is defined as an object consisting of four parameters: stroke centroids C(x,y), the length of the rotation angle along the rotation angle direction (L1, L2), and the length of the vertical rotation angle direction(W1,W2), the estimation of each stroke is equivalent to parameter estimations for each stroke centroids. The centroids are generated from a sampling given the estimated density maps of the referenced images, as described in Section 4.5. In this section, we described the parameter estimation given the reference images.

4.5.1 Rotation Angle (θ) Estimation

As for the rotation angles of each brush stroke, researchers has described a method using Edge Tangent Flow (ETF) vector field of images [49]. The underlying idea is that The ETF is calculated based on the gradient field of the image, which includes distinct edge features produced by stacking textures of brush strokes in artistic paintings. Thus at the pixel point, the gradients can be used to guide the estimation of the direction of the brush strokes. Thus for each stroke patch that includes multiple pixels, ETF vector field offers a method to generate an overall direction given gradients of neighboring pixels in the same stroke. The computation of ETF consists of three steps. Compute the gradient vector field (modulus and direction at each pixel); rotate the direction by 90 degrees, roughly pointing to the edges' direction. Approximate the edges by changing the direction of the pixels with a small modulus to that of its nearby pixels with a large modulus while the modulus keeps the same [49].

4.5.2 Lengths ($L1, L2, W1, W2$) Estimation

For each stroke, the vertical and horizontal length parameters (L1, L2, W1, W2) are valued based on the regional approximation of the input image using HSV values. HSV (Hue, Saturation, Value) color space is a better image format than RGB (Red, Green, Blue) format because it separates color and brightness information, containing the perceptual qualities of colors at each pixel. Additionally, the HSV color space is more intuitive for humans to understand and work with, as it corresponds more closely to how we perceive and describe colors in the world around us [50, 51].

In length estimation, since colors at each pixel can be valued in HSV image format, the estimation is a process of detecting possible lengths, comparing the differences between the HSV values of pixels at that length and the starting centroids, and deciding to accept such length. The numerical comparison can be defined as comparing hues and brightness values of starting point (the stroke centroid) and the candidate pixel$P(x, y)$ [49]. Assume the starting stroke centroid is$C(x_0, y_0)$, the length L should be a value such that

$$L = (x_0, y_0) + r * (cos\beta + sin\beta) \tag{4.9}$$

$$|H(x, y) - H(x_0, y_0)| \leq t_H \tag{4.10}$$

$$|V(x, y) - V(x_0, y_0)| \leq t_V \tag{4.11}$$

where r is the searching step length (generally set as integers), and β is the rotation angle for different length estimations. Namely, for L1, $\beta = \theta$, and for L2,$\beta = \theta + 2\pi$, and $\beta = \theta + \frac{\pi}{2}$,$\beta = \theta - \frac{\pi}{2}$ for W1 and W2 respectively. And the t_H,t_V are thresholds for comparison.

Since the final output is the maximal length that satisfies formula(4.9)(4.10)(4.11), in programming the searching process starts with $L = 0$ and updates the starting pixel (x_0, y_0) as (x, y). It repeats till the formula (4.10) (4.11) is not satisfied. Every move in the searching process should be at least one pixel, but too large searching steps may yield zero lengths. For simplicity, we set r as one pixel.

4.6 Render Strokes

After generating a sequence of brush strokes with parameters, the next step is to place every stroke onto the canvas. Generally, we can place the strokes one by one, but in the experiment, we sorted the stroke sequence by size to illustrate different painting pipelines. In this experiment, the painting pipelines, which imply different painting techniques used in human painting processes, refer to the different orders of placing strokes onto the canvas. The painting pipelines such as "full painting", "detail first" and "background first" are encoded as an argument "paint_mode" in programming. Particularly, we shifted the order of foreground and background drawing to illustrate the painting techniques such as "detail first" and "background first".

Since in the painting process using a circular stroke model and template-based painting model, the strokes are parameter sequences, they share the same rendering strategies but different parameters.

4.7 Experiments I: "Details first" vs. "Background First" in painting process

Since the segmentation phases have been explored in the previous work, the following experiments focus on the painting design part, the second phase shown in **Figure 3.1**. These experiments optimized the painting process based on a research-through-design approach iteratively. The design starts with a simple drawing process that measures the system performance with a combination of visual evaluation and quantitative metrics, focusing on exploring whether different drawing models can be applied to painting processes. We consider a global optimization-based approach for estimating stroke parameters in the initial stage. Since the complexity of estimation is related to the number of strokes and the number of parameters in a single stroke, among the three brush models shown in **Figure 4.8**, the circular brush is the model with the lowest number of parameters and is therefore used as the initial stage in the experiments. Moreover, the maximum number of brushes is adapted to the optimized threshold value. **Table 4.1** showcases the hyperparameter settings.

Table 4.1: Hyperparameters for initial system design of painting process.

Parameter Name	values
brush stroke model (S)	"circular" S(p,r,c)
painting mode (P)	"f-b" or "b-f"
max brush size	$0.25 \times min(h, w)$
approximation threshold (T)	100
line model	bazier curve
max iterations	500

In **Table 4.1**, the circular brush model can be parametrically represented as S(p,r,c), where p represents the two-dimensional coordinates of the pixel point, r is the brush radius, and c is the color of the brush, using the color value at p. The drawing modes are defined as "f-b" and "b-f" to indicate the drawing order of different image areas. In the maximum brush size, h and w denote the height and width of the reference image, respectively, and the maximum brush size, measured as the brush radius, does not exceed $\frac{1}{4}$ of the reference image size (measured as the minimal edge). The termination threshold of the optimization process is set to 100.0 pixels to indicate the expected error between the generated image and the reference image.

4.7.1 Configurated pipelines

The painting mode is configurated to showcase procedural painting techniques such as "detail first" and "background first" or even start at every content part. This section first shows the impact of a painting technique known as "detail first" on the placement of color blocks and areas of focus in creating artwork during the implementation of image segmentation. Unlike block painting techniques that emphasize segmenting the color of artwork, detail-first focuses on first capturing an area of interest in the image and then gradually painting around that item until the entire foreground is completed, eventually extending to the entire background. This painting technique is commonly found in realist portraits, where the figure is used as the foreground content, and the background is the remainder. Therefore, this section is designed with a painting process based on detail-first and circular brushes. In this design, the input image is a person's portrait from the DRAM dataset, and the painting process is divided into two modules: segmentation and rendering. The purpose of segmentation is to separate the painting's foreground ("portrait"), and the rendering process will prioritize rendering this part.

Contrary to the above, the second part explores working on the background part, applying large brush strokes to complete the background painting quickly, and then applying smaller brushes in the foreground to retain good painting details. In the process, the same test images are applied for comparison, and the effects of the application of landscape art painting are also explored.

4.7.2 Complete "voids"

Based on the investigations in the previous two sections, this project achieved a basic analysis-painting process, but there needed to be more in the presentation of details. Very typically, the random sampling and estimation of the painting system may cause local "voids" in the final painting, which rarely occur in the paintings of human artists. Hence, a preliminary conclusion is that the previous painting system left this problem behind. Therefore, this section updates the system design of the painting part by adding a detail filler module in the final stage to detect and fill the "voids" in the generated paintings.

Filling "voids" can be designed as a sub-iterative process. This process can filter out local voids by comparing the generated drawing with the original reference image to generate a void mask; after setting the error threshold, the process is stopped by using the local area covered by the void mask as a reference and painting further on the image generated in the previous sequence until the error threshold is reached. Since the local area where the voids appear is minimal compared to the overall reference image, the error threshold can be taken in equal proportions when setting the error threshold. For example, given a (500,400) reference image, if the approximation threshold T is 100 using the formula from Table 4.1; and assume that the voids cover 10% pixels of the reference image, then the approximation threshold for the sub-iteration should be 10, which is proportional to the overall threshold.

4.8 Experiments II: Iterative design for improvements

This experiment is to investigate how to improve the above drawing flow design. Based on the previous testing phase results, the improvements are mainly aimed at the prediction module of strokes and the generalization of the drawing process. For the prediction module of brush strokes, this experiment mainly explores the possibility of applying the neural network model as the backbone to improve the prediction effect. In addition, this project tries to generalize the drawing process by using the analysis of abstract paintings and drawing results as an example.

4.8.1 Application of neural networks

In the research-through-design approach, the completion of a complete system design is only the initial step in the design process, while iterations allow for the exploration and experimentation of different design options to discover new and innovative solutions. The descriptions in Sections 4.7 complete the design of a complete painting system by testing and evaluating it under different parameter conditions. However, the computation time of the optimization error-based brush estimation is long, and it is difficult to capture the features between the local coordinates of the image.

Deep learning models can theoretically win over the method of parameter optimizations in terms of estimation efficiency and feature extraction. Because brush estimation based on optimization errors involves solving an optimization problem to estimate the brush strokes in an image, this optimization process is computationally expensive, especially for reference images of large size or more complex brush models. On the other hand, deep learning models can automatically learn texture features between local pixel points by processing large amounts of training data. The model learns to recognize the patterns and features associated with strokes in the input image and generates the corresponding strokes that best represent these features. Therefore, further, this experiment attempts to update the brush prediction part to a neural network model as the backbone based on the preliminary design. Regarding model selection, all the neural network models used for AI-generated paintings can be used as backbones. In this experiment we mainly improved the approach based on the previous stroke prediction methods [27]. Its operation in practice will be presented in Section 5.5.

4.8.2 Explore on abstract paintings

Testing the painting pipeline on abstract paintings is necessary to ensure that the pipeline can handle the complexity and creativity required by this art form and produce quality results that meet the standards of the art world. On the one hand, abstract paintings are often more complex than representational paintings, making them a more challenging test case for the painting pipeline. The use of color, shape, texture, and other elements in abstract art can be more diverse and less predictable than in representational art. Therefore, testing the pipeline on abstract paintings ensures that the pipeline can handle the complexity of such paintings. Moreover, testing against abstract paintings is essentially a test of the effectiveness of the perceived artistic process that has been designed. It is a highly creative and subjective art form that requires a high degree of artistic interpretation and decision-making. However, in the previous design phase, the art perception methods based mainly on image segmentation and brush analysis may have limitations. Therefore, this test helps us evaluate this phase and address the challenges of achieving unified art perception and generation.

As a highly creative and appreciative subjective artistic presentation, abstract painting is extremely challenging in unified artistic perception and generation. This experiment attempts to analyze the feasibility of system design by applying the previous system design to the practice of painting with abstract painting as a reference.

5 Results and Analysis

This section presents the significant results of each experimental session and the corresponding analysis. Depending on the different questions explored in the experiments, these results mainly contain tests of different segmentation methods on art paintings, the design of the painting process under the application of different settings, and the corresponding evaluation metrics. In Section 5.1, we first describe the evaluation methods used in the analysis. Section 5.2 shows the test examples and analysis of different segmentation methods; Section 5.3 presents the results of comparing sampling methods, and the procedural painting results are showcased in Sections 5.4 and 5.5.

5.1 Evaluations

5.1.1 Quantative Evaluation

In this study, since the input image is the same as the target image, only the decisions of the painting techniques including the brush stroke type and procedural pipelines are considered,the evaluation measures the influence by using the local mean square error (MSE) between the generated images and the referenced images . That is

$$L_{MSE} = \frac{1}{N}\sum_{i=0}^{N}(R_i - G_i)^2 \tag{5.1}$$

where N is the number of local units of differences, and R_i, G_i represent the true values and generated values from referenced images and generated images,respectively. In equation (5.1), local differences can be directly expressed as errors between pixel points. For example, given an referenced image of size (500,400), the number of local difference units N is 200,000, and R_i,G_i are pixel values in images. However, using pixel loss measures a large number of pixel differences, and it treats each pixel as an independent unit. According to the inspiration of the neural network model, there are associations between local pixels in the image. These associations can be extracted as features, so another idea is to define the evaluation metric as the mean square error between image features of different layers. That is,

$$L_{MSE} = \frac{1}{N}\sum_{i=0}^{N}(R_f - G_f)^2 \tag{5.2}$$

where R_f, G_f denote the features extracted from referenced images and generated images by same models, which can be typical deep machine learning models such as

VGG Networks.

One advantage of using local features instead of pixel values is that it can decrease the number of computation units N since the number of output nodes in VGG networks decides the number of feature units that can be customized. Moreover, using deep machine learning models to extract local features includes spatial correlations between neighboring pixels, which more intuitively decides the visual differences between the generated and the reference images. Therefore, the quantitative analysis of the generated art paintings in this project combines the above two loss computations.

5.1.2 Visual Analysis

Direct observation of the visual effects of different generated art paintings is also a valid method in addition to quantitative evaluation. This method is mainly used to qualitatively analyze the effect on different content paintings under the same painting process and the rendering effect of different painting processes and parameter settings for the same art painting. Of course, this method inevitably includes the subjective consciousness of the observer, and the use of a large-scale mean opinion score (MOS) is a way to avoid this problem. Specifically, the number of test images or the number of interviewed users in the same condition can be expanded.

5.2 Segmentation Results

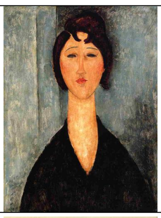
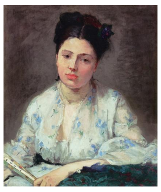
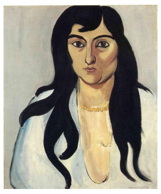
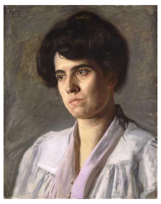
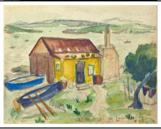
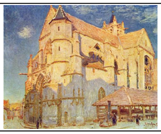
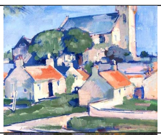
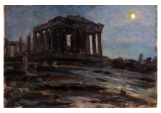
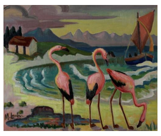
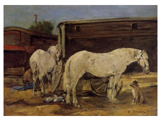
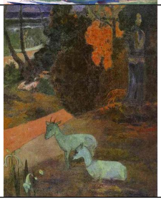
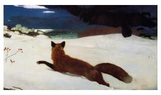
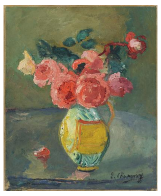
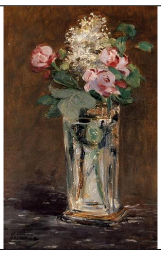
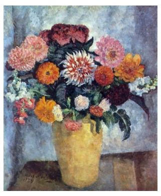
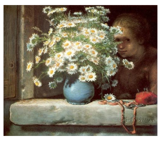
This section compares traditional computer vision-based methods (i.e., K-means clustering), deep learning networks (i.e., DeepLabV3), and domain adaptation-based methods for before and after view analysis of art paintings. All three methods were tested on the same dataset of art paintings. Also, to compare as much as possible the performance of these methods on different artworks, we used 4 different styles defined by the Art Movements and 4 primary contents of images for testing, especially for the segmentation effect of abstract paintings.

The samples of art paintings are presented in **Table 5.1**. Since these three methods contain different parameters that can affect the results, we explore the parameters for each method respectively and exploit the parameters that yield optimal results in comparison.

5.2.1 K-means clustering

From section 4.1, we have learned that the K-means clustering method is feasible for segmenting art paintings. Those art paintings using RGB color space are more likely to have good segmentation results in connecting neighboring pixels with the same colors than those using HSV color space. Therefore this section shows the results of foreground segmentation using this method in paintings with different themes and artistic movements. The results are shown in **Table 5.2**, in which the white masks indicate the excluded contents (i.e.,"background") and the other parts in colors are

Table 5.1: Sample test images for segmentation.

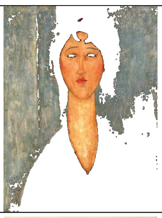
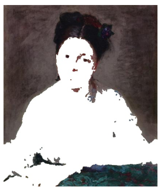
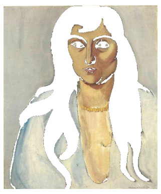
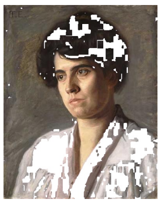
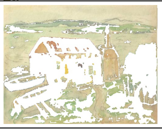
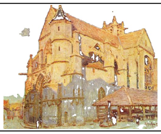
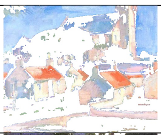
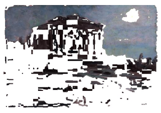
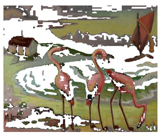
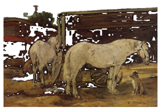
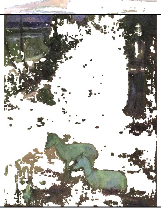
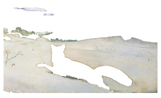
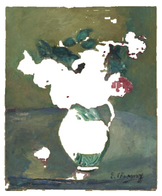
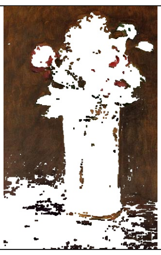
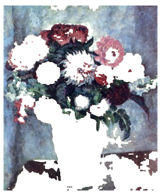
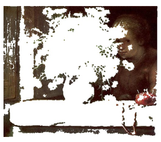
Content \ Style	Expressionism	Impressionism	Post_impressionism	Realism
Portrait				
Buildings				
Animals				
Potted flowers				

extracted foreground.

In **Table 5.2**, all the art paintings were segmented with color differences. The first row shows that every facial part in the portraits was extracted, while the hair in all portraits and the shoulder of the woman in black (portrait in expressionism) were excluded. For art paintings of "buildings" shown in the second row, most of the buildings were inluded in foreground along with some theoretically background parts, and the walls of houses in the first column were missing. The art paintings containing animals and potted flowers are partially segmented given the spatial distributions of colors.

One conclusion is that using K-means clustering to segment the foreground is a color-based method. Thus its efficiency depends on the statistical features of colors in the art paintings. Specifically, this method looks for K clusters that have different dominant colors. For example, in the art painting of a fox in the wild shown in **Figure 5.1 (a)**, the color of the fox and the night sky in the background is similar. Both are distinctive to the snowy ground, so when K=2, this algorithm clusters the fox and the sky together while separating the white ground, shown in **Figure 5.1(b)**. This conclusion also explains why the algorithm separates the black hair from the facial part of the portraits in **Table 5.2**. The black hair part has a distinctive color from the other part, and the facial part has a more significant similarity with the

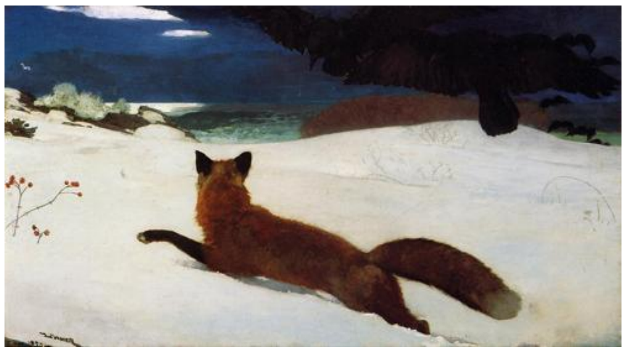
Table 5.2: Extracted foreground by using K-means clustering ($K = 2$) method.

Style / Content	Expressionism	Impressionism	Post_impressionism	Realism
Portrait				
Buildings				
Animals				
Potted flowers				

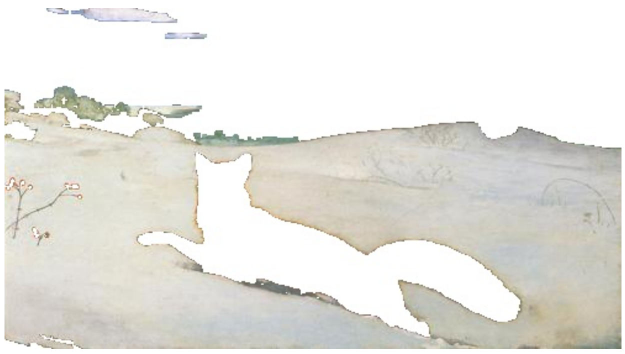
background part in colors. Therefore, the algorithm separates all the black parts into other clusters. Despite missing some semantic parts, using K-means clustering to segment art paintings is reasonable because it parallels the idea of color blocks in creating art paintings, where every color block can be considered a segment and can be created simultaneously.

5.2.2 DeepLabV3

The segmentation results can be seen in **Tables 5.3**. It showcases that this method failed to extract foreground from all 4 art paintings that contain "buildings" as their preliminary contents, neither from the art paintings contain "potted flowers" in impressionism art movement. It implies that this method fail in extracting foreground from art paintings contain some specific contents or in specific styles. However,the segmentation results on portraits (the first row in Table 5.3) and animals (the third row in Table 5.3) gathered more semantic elements in the foreground compared with thoes in Table 5.2. Specifically, the foreground of portraits contained hair and body parts, and the animals were separated from the background. In terms of the potted flowers (shown in the last row in Table 5.3), the dominant contents in the example potted flowers in expressionism and post_impressionism were not included in the foreground results. Even though the other two example paintings of potted flowers were segmented, only part of pots and flowers were included.



(a) original



(b) foreground

Figure 5.1: K-means clustering results of art paintings of animals ("a fox in the wild") in Realism, K=2.

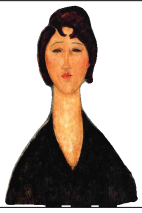
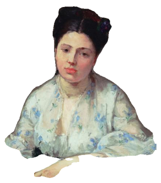
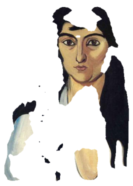
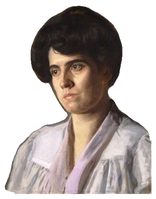
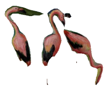
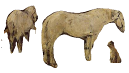
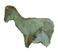
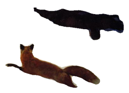
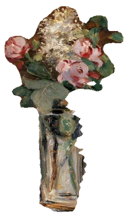
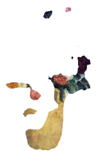
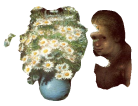
Moreover, we tested foreground/background segmentation on art paintings containing multiple portraits and landscapes. The results shown in **Figure 5.2** indicated that segmentation with the DeepLabV3 method had no problem separating these figures from the background, **Figure 5.3** yielded similar results at the first two columns, which include "persons" and "horses" in the foreground. However, in **Figure 5.3**, it failed to segment anything from the pure scenery shown in the third row, thus every element was included in the background part.

Although DeepLabV3 can achieve semantic segmentation of some art paintings, there are limitations in some specific styles or dominant components. One possible reason is that since the DeepLabV3 network is trained on a dataset of photographs, and the visual differences between photographs and art paintings can be huge, the segmentation of art paintings is limited in foreground/background segmentation.

5.2.3 Using Domain Adaption

From **Table 5.3**, we can infer that the art paintings of buildings cannot be segmented using DeepLabV3 because it was trained on PASCAL dataset, which contains 20 classes of photographic items, and class "buildings" is not included. Therefore, the semantic segmentation model DeepLabV3 and the method using it as backbone

Table 5.3: Extracted foreground by using DeepLabV3(ResNet 101).

Content \ Style	Expressionism	Impressionism	Post_impressionism	Realism
Portrait				
Buildings				
Animals				
Potted flowers				

cannot classify any image of contents in class "buildings". So Table 5.4 omitted the segmentation results for these art paintings.

The segmentation results of using domain adaption and DeepLabV3(ResNet 101) is shown in **Table 5.4**. In **Table 5.4**, all 4 portraits were segmented with the faces,shoulders and hair included in the foreground. The animals in the second row are well-preserved in the foreground. In terms of art paintings of potted flowers, only the one in post_impressionism has improved segmentation result compared with **Tables 5.2** and **5.3**.

5.3 Density Map Estimations

This section presents the result and analysis for searching stroke anchors given the reference images. we implemented the experiment stated in Section 4.5 with 100 color images (50 paintings of portraits and 50 paintings of landscapes). There are 3 hyperparameters in the experiment: the maximum number of iterations for computing the average MSE of KDE method(max_iter), the kernel size for Gaussian blurring(k_size) and the number of components for GMM (n_component). For simplicity, we set max_iter = 10,k_size = 3 and n_component = 3. The results can be seen in **Table 5.4**.

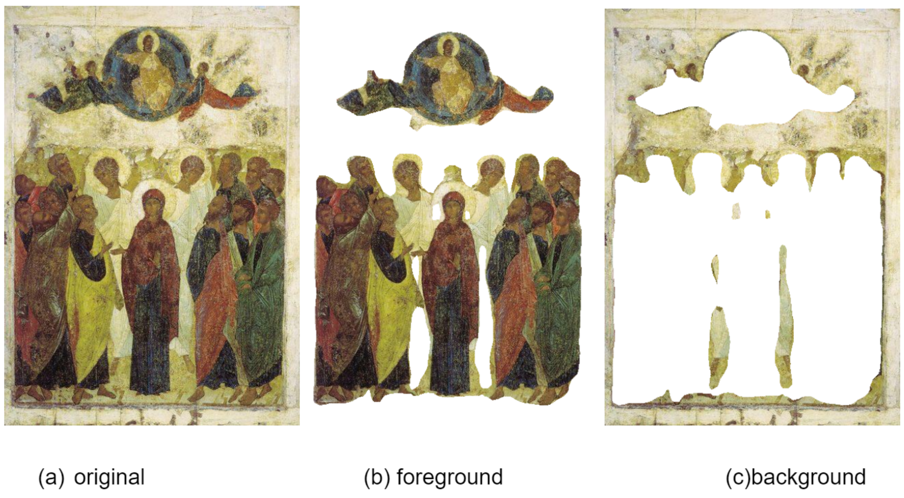


Figure 5.2: Segmentation results of DeepLabV3 (ResNet_101) on paintings of multi-portaits.

Table 5.5 shows that GMM-based computation for probability density maps yields smaller MSE than a gradient-based method, demonstrating that probability density maps of images generated from the GMM-based method are closer to the ground truths of reference images. By statistically analyzing the comparison results on 100 art paintings, the GMM-based density estimation methods have a higher frequency (59%) of smaller MSE. However, in super-pixel paintings, the gradient-based density method may win over GMM-based method with smaller MSE. Since the test images are in the same range of input images used in further experiments, we can conclude that the GMM-based density estimation performs better in this project.

Gaussian blurring can reduce the MSE loss for both methods because it filters potential noises. Since the kernel size of the Gaussian filter decides the blurred images, choosing an appropriate kernel size is essential. However, the GMM-based density map contains nearly 10 times the number of estimated anchors than that of the gradient-based density map. In brush stroke sampling, it is necessary to reduce the number of anchors for the sack of computation complexity.

As stated in Section 4, the maximal values of probability density for sampling stroke centroids p_max can affect the number of centroids. And Section 5.2 implies that using gradient-based density estimation can generate fewer centroids. In practice, we need to find an optimal number of stroke centroids since too many strokes may incredibly increase the computation complexity while too few strokes may lose the capability of presenting fine details of the art paintings.

Obviously, p_max should be in range [0,1]. To decide the value of p_max, I tested candidate values ($\frac{1}{4}$,$\frac{1}{36}$,$\frac{1}{64}$) on art paintings in three different content and style categories: abstract painting, landscape paintings and portrait painting. The results can be seen in **Figure 5.4**.

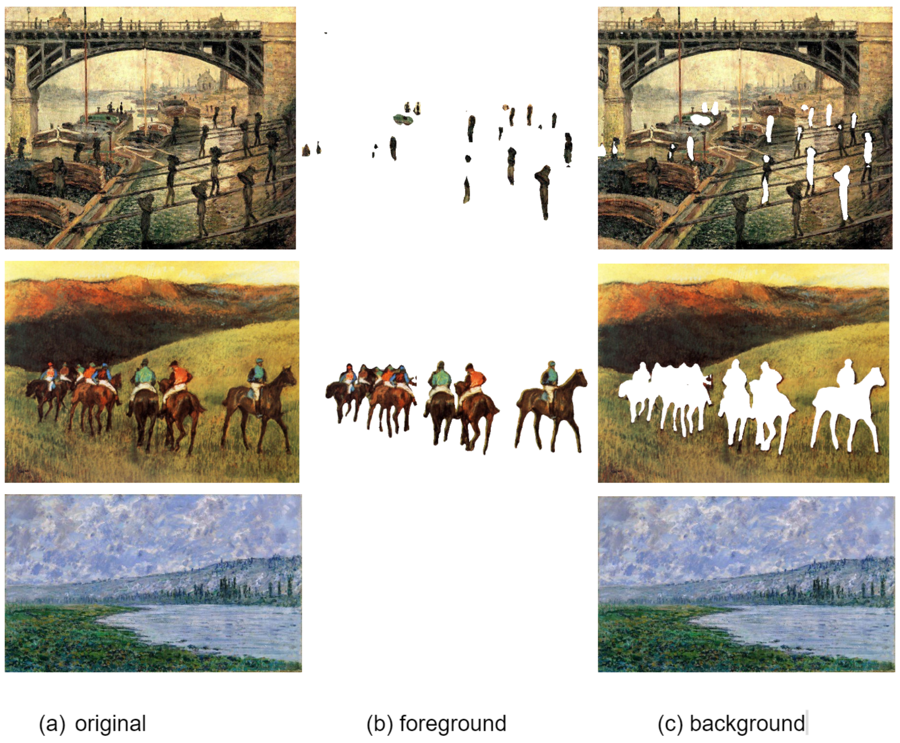


Figure 5.3: Segmentation results of DeepLabV3 (ResNet 101) on paintings of landscapes.

5.4 Sampling results given density maps

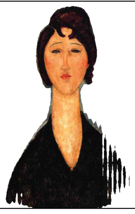
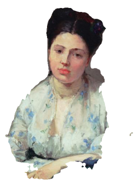
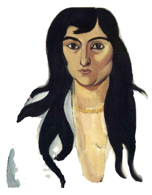
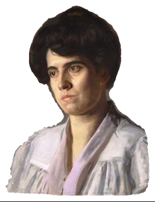
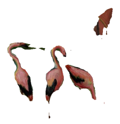
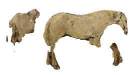
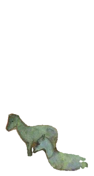
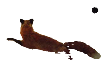
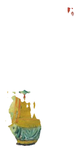
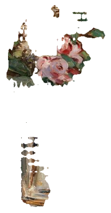
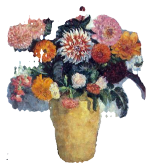
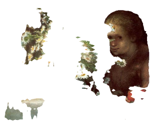
From **Figure 5.4**, we can infer that larger p_max yields a larger number of sampled stroke centroids K. The third column (c(1),c(2),c(3)) illustrates obviously weighted distribution of the stroke centroids, in which we can recognize the sparse region and dense region (mainly edges) of the sampled maps. However, the centroids distribution in the fourth and fifth columns of Figure 5.4 show that too sparse centroids can not represent the actual density distributions, thus it can be very difficult to extract regional features given such stroke centroids.

As for efficiency, obviously the large number of stroke centroids took a longer running time to finish the parameter estimations and final rendering steps. So for simplicity, it is necessary to optimize p_max in order to gain good painting results while avoid large time and space complexity.

5.5 Stroke-based Painting Results

In this section, we present some procedural painting results with different configurations including the brush stroke types, maximal brush stroke size, painting modes and styles of reference images.

Table 5.4: Extracted foreground by using DeepLabV3(ResNet101) with domain adaption.

Style / Content	Expressionism	Impressionism	Post_impressionism	Realism
Portrait				
Animals				
Potted flowers				

5.5.1 Procedural Painting Processes

To present the procedural painting process, we use a strategy of saving resulting canvas per 100 stroke placements. **Figure 5.5** showcases the procedural results of total 289 rectangular brush strokes with the maximum sampling probability $p\_max = \frac{1}{64}$, the reference image is the portrait shown in **Figure 5.4(a3)**. **Figure 5.5** shows that the brush stroke estimations can progressively make the canvas closer to the reference image. As the brush strokes increase, more details are addressed.

5.5.2 Test results on abstract paintings

The test result on abstract paintings using the rendering process is shown in **Figure 5.6**, where the brush stroke model is triangular with the brush size decreases proportionally. The maximum brush size was set as 500 pixels. However, the painting process shown in **Figure 5.5** relies on the random sampling of stroke centroids so that all color regions are covered in each sampling. Thus the rendering process generates brush strokes in all colors simultaneously, which ignores that the painting process of artists usually focuses on one color region or object block at once. Therefore, the segmentation of art paintings is introduced to indicate the underlying idea of selecting object blocks or color segments in creating art paintings. **Figure 5.7** shows the proposed painting process in Section 3 with different painting modes configured.

Table 5.5: Test results of GMM-based density estimation and gradient-based density estimation on 100 art paintings.

image_id	pixels	m1_anchors	m1_MSE	m2_anchors	m2_MSE
1	4633790	3300026	0.115200626	434729	0.258515911
9	3170846	1798592	0.047195653	164956	0.404194104
32	1066500	670514	0.031024918	94137	0.33374438
37	851968	420420	0.033029294	64530	0.196625398
8	408342	248337	0.2480008	23343	0.184701111
93	380904	246212	0.203497653	33118	0.098629718

In the experiment, the painting mode represents the pipelines of shifting active areas to finish an art painting. When it was configured as "f-b", which represents to paint foreground first and the background later, the painting system will take the stored image of foreground elements as the first reference image, then use the generated canvas as the starting canvas with reference to the background images to finish the second stage of painting process. The brush stroke model was set as "circular", and the brush size was defined as the radius, which decreased in half from 16 to 2.

Figure 5.7 indicates that smaller brush sizes can address more details of the art paintings, while the sampling size of brush strokes may be extremely large, which causes the painting process consuming much time.

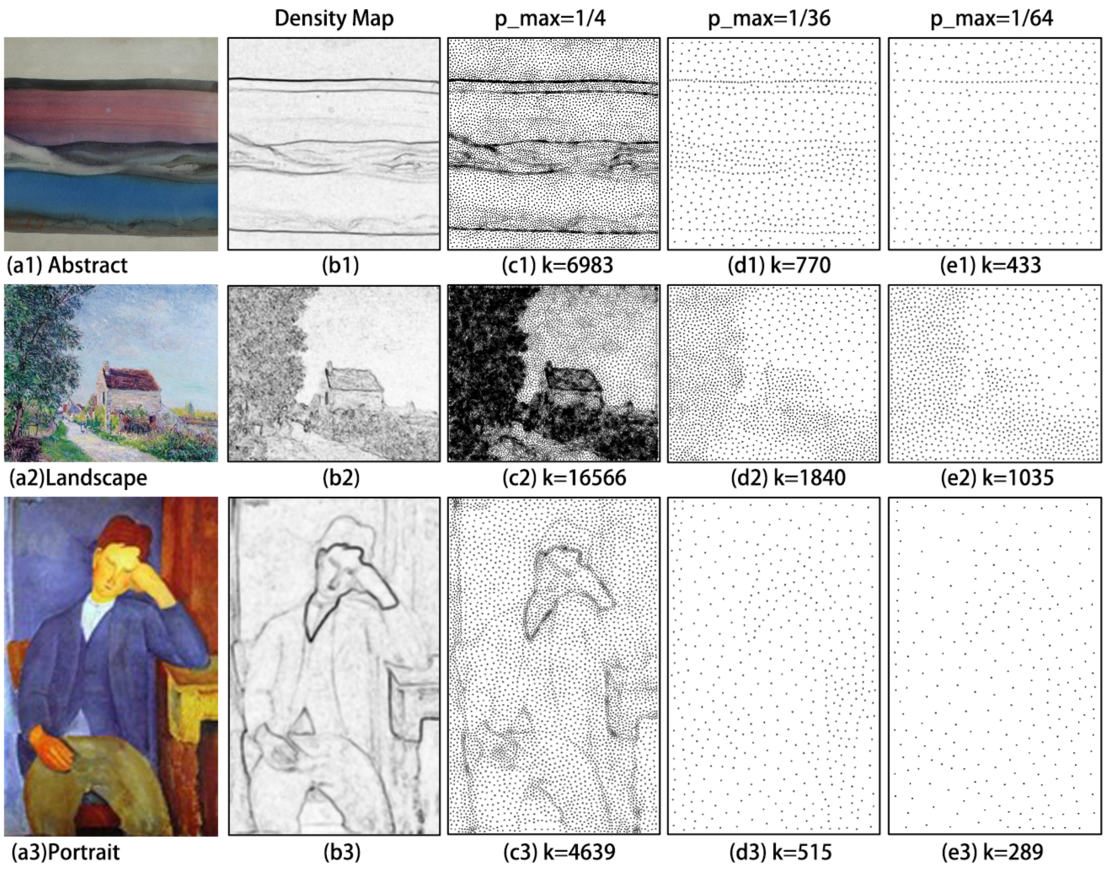
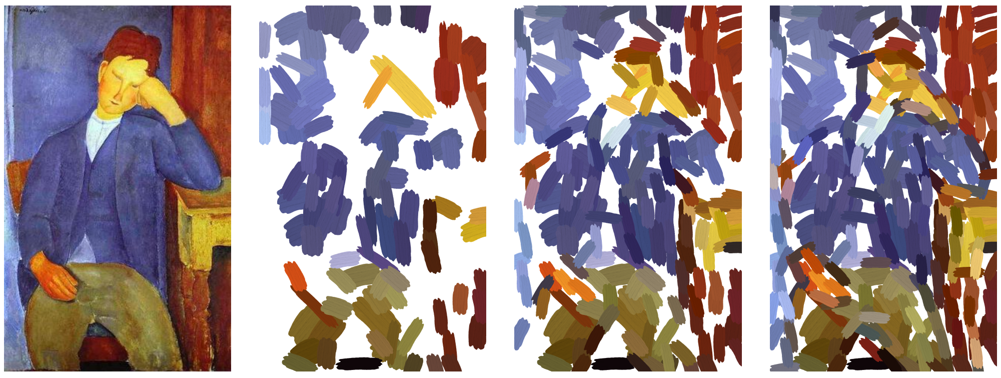


Figure 5.4: Results of sampling stroke centroids with different values of p_max.



reference $N_s = 100$ $N_s = 200$ $N_s = 289$(final)

Figure 5.5: The painting process of a portrait example using rectangular brush strokes.

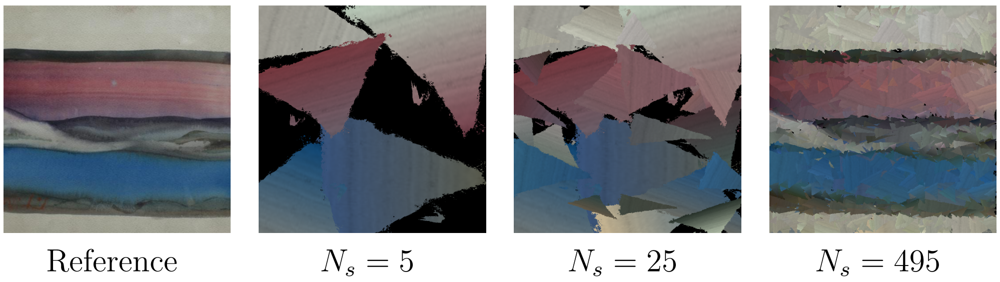


Figure 5.6: Painting process on an abstract painting.

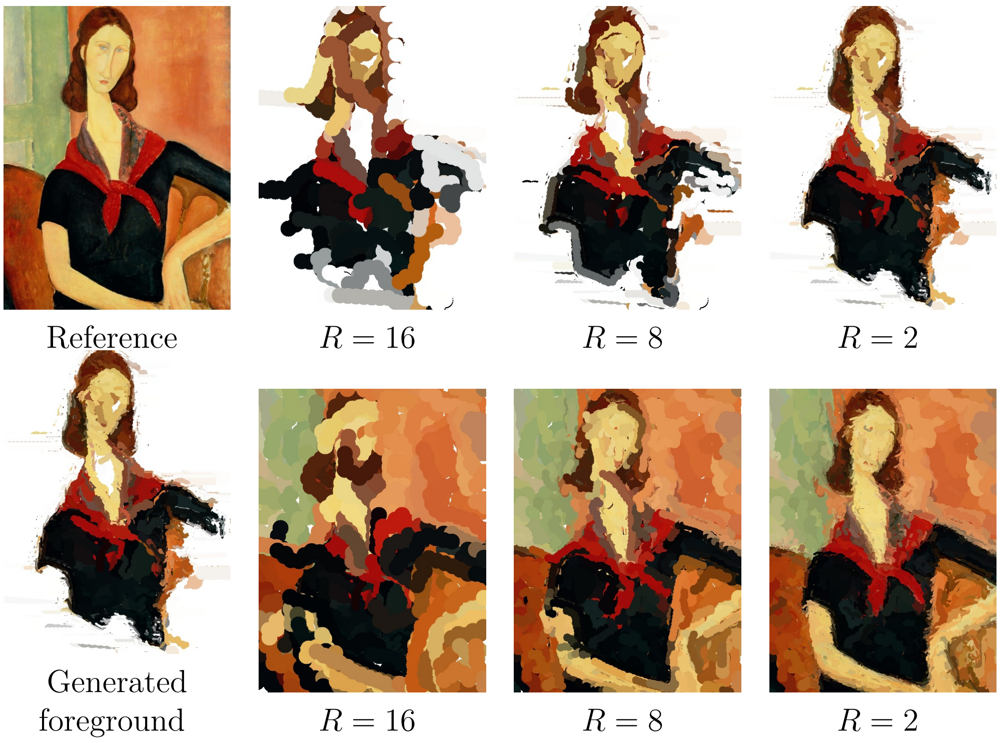


Figure 5.7: Illustration of painting process of a portrait with *paint_mode* = "f-b".

6
Conclusion

This thesis finished designing a generative painting pipeline that uses a series of machine learning techniques to involve painting techniques from natural painting practices. The thesis designed the painting pipeline as a shift between regions of interest in art paintings by taking specific painting techniques, including "detail first" and "detail last" as examples. The generative processes were designed as brush stroke estimations. The generative painting pipeline takes painting techniques from natural painting practices as configurations, which makes the generative painting pipeline controllable.

The resulting art paintings generated from the pipeline create novel aesthetics from machine learning techniques and natural painting practices. With the configuration of the painting pipeline that controls the brush stroke models and the paint mode, we can quickly create novel art paintings with similar content but controllable artistic features. Therefore, this generative painting pipeline can be used to create more art paintings in the future. Moreover, the generative processes may involve human thinking and emotions in generative art, which has already introduced a new method of creating visual art.
Moreover, the research compared different machine learning techniques in module designs described in Section 6.1. The limitations and possible future work related to this research are presented in Sections 6.2 and 6.3, respectively.

6.1 Discussions on Sub-tasks

6.1.1 Segmentation techniques for art painting

By analyzing the results in **Tables 5.1** to **5.4**, we can conclude that although the deep learning model DeepLabV3 can capture different layers of image features compared to the machine learning model K-means clustering, the segmentation based on K-means clustering still has an advantage in this experiment. Because artistic creation gives artistic representation to objects in the physical world, it makes the physical objects referenced by the painting art more different in form and color. This difference is related to the painter's artistic skill and style preferences. However, in semantic segmentation, the morphological characteristics of an object are critical indicators to distinguish a specific type. This makes semantic segmentation particularly difficult in the domain of art paintings.

Moreover, semantic segmentation combined with domain adaptation can effectively avoid object morphological differences due to artistic styles. However, on the other hand, the dataset for training semantic segmentation models often covers only part of the object shapes and thus still shows limitations. For example, since the DeepLabV3 model is a semantic segmentation model trained on the PASCAL dataset, it can only recognize 20 object classes included in the PASCAL dataset, making it difficult to be useful in the field of art painting with a wide range of subject matter.

6.1.2 Impacts of brush strokes

How an artist uses brush strokes can affect the texture, depth, and overall appearance of a painting, resulting in a unique visual effect. The number and size of strokes a painting uses can affect its overall appearance. For the same painting, using a high proportion of strokes can create a sense of detail and complexity, while using fewer strokes can lead to a more superficial, minimalist appearance. On the other hand, using a larger brush size can create bold, expressive strokes, while using a smaller brush size can produce more detailed, complex strokes. In addition, the choice of brush can determine the textural character of a painting. For example, thicker, heavier strokes can produce a rough, textured surface, while using thinner, lighter strokes can produce a smoother, more even texture. In addition, the type of brush used (e.g., hard or soft bristles) can also affect the texture and appearance of a painting.

6.2 Limitations

6.2.1 Segmentation on art painting

The rendering performance using the proposed painting pipeline requires better methods to solve the connection between colors and shapes. Color-based segmentation methods are regarded as a better solution for art paintings, but it suffers from problems such as understanding and presenting the shapes of items in a painting. Even though more advanced deep learning models can 'learn' the features of different paintings to segment figures and shapes in realistic styles, they have difficulty solving problems within understanding and segmenting abstract painting styles.

6.2.2 Art perception

Artistic creation is more than the application of a particular style to a realistic scene, but is a process of manipulating the mental processes of viewers by shaping their sensory experiences and memory recalls. While we can use AI techniques to try to understand existing paintings or use deep generative models to create new works of art as other research has done, it still struggles to capture the complexity and subjectivity of artistic perception. First, visual art is broad and diverse and not widely available, making it challenging to build a comprehensive dataset to train AI algorithms. Moreover, artistic perception is highly subjective, even for the same visual presentation. AI decisions relying on data and rules can hardly involve human emotions, experiences and cultural context. On the other hand, AI algorithms are designed to recognize patterns and make predictions based on available data. They have yet to produce truly original or creative works of art.

6.2.3 Absence of procedural information

The painting process may vary depending on the artist and the style or technique. However, in general terms, it involves first sketching to establish the fundamental values and colors of the painting and secondly using broad brushes and large areas of color to establish the overall composition, i.e., masking. These two processes can establish the basic shapes and color partitions of the painting's ground. Next, the artist often uses refinement techniques to add localized details, textures, and subtle color variations. The information included in the description of the painting process can provide a more detailed and nuanced perception of the painting process and the artwork itself. From artistic imagination to the formation of an artistic painting, this information encompasses the choice of materials and tools to be used, the techniques used in the painting process, and the composition and design of the color layout. Various painting techniques, such as blending, layering, and glazing, can be directly expressed in the final painting as brush marks and textures.

However, information about the process of creating art paintings is difficult to obtain. Many works of art were created in earlier periods when documentation and record keeping were less comprehensive than it is today. As a result, little or no information may be available about the painting process for these works. Even if records existed when the artwork was created, they may have been lost or destroyed over time, making it challenging to piece together information about the painting process. In addition, different artists may have used different recording methods, which makes it difficult to compare and analyze the painting process of different artworks.

Moreover, painting techniques and methods can vary significantly between artists and styles and even within the works of the same artist, which makes it challenging to generalize the painting process based on a small sample of works. Therefore, this is a huge challenge for AI models that rely on such information to be trained. We are limited to making artificial generalizations about a feasible painting process based on existing painting processes. During the system design process, the designer misunderstands or does not fully understand the meaning and significance of the artwork, and therefore the designed process is different from the actual rich and variable painting process.

6.3 Future work

6.3.1 Separate perception and rendering

Inspired by applying artistic style transfer to the segmentation of art paintings, a possible generative painting architecture can be a possible solution. When an artist creates a painting from specific scenes and objects, the artistic imagination process can be regarded as a style transfer between the realistic space to artistic representations. For example, when a human artist observes specific scenes and objects in real life, the precursory process of artistic creation is already underway. In this process, the artist thinks about how to present the artistic effect of the painting, which can be seen as a stylistic mapping between the actual scene in the physical world to the artistic effect imagined by the artist. The artist dominates this style and often presents correlations within the same painting. Thus, for any painting based on the imagination of a natural scene, we can record the difference between the image in the artistic painting and the natural scene in a specific artistic style, which can work well especially in realism paintings.

Thus, the generative painting process that incorporates a stylistic mapping from the natural scene to the painting consists of two stages: the artist's imagination and the rendering. In generative painting, the input is correspondingly photographic images, and the main task of the imagination phase is to find a style mapping of the input image that can be applied after the segmentation of the photographic images, thus improving the difficulties of traditional machine learning methods in the task of image segmentation of artworks.

6.3.2 Modifications for generating abstract paintings

Due to the cryptic nature or even the absence of content semantics in abstract paintings, semantic-based image segmentation is likely to perform mediocrely in analyzing abstract paintings in the design of the painting system shown in **Figure 3.1**. One possible improvement is to use an image segmentation method based on color distribution or texture features. For example, using color distribution-based image segmentation to analyze the color block features of abstract paintings, abstract paintings with similar color block distribution features are used as a class of style templates, and generative adversarial networks (GANs) can be used to generate new abstract paintings with similar visual features to the original paintings.

Another strategy for generating abstract art paintings is to train a convolutional neural networks (CNN) individually on a large dataset of abstract paintings, potentially developing a model that can identify different visual features in the paintings and classify them based on their style or other features. A generative adversarial network is then employed to generate new, abstract art images of the specified visual features. However, such generated abstract paintings do not require any reference images and are random in nature. Therefore, once the CNN model for analyzing the abstracted visual features is trained, the subsequent painting process based on the given reference image can be implemented by style transformation. However, in this approach it would be challenging to include user-configured painting techniques.

Bibliography

[1] Margaret A Boden and Ernest A Edmonds. What is generative art? *Digital Creativity*, 20(1-2):21–46, 2009.

[2] Philip Galanter. Generative art theory. *A Companion to Digital Art*, 1:631, 2016.

[3] Kıvanç Tatar, Mirjana Prpa, and Philippe Pasquier. Respire: Virtual reality art with musical agent guided by respiratory interaction. *Leonardo Music Journal*, 29:19–24, 09 2019.

[4] Alison Colman. Net. art and net. pedagogy: Introducing internet art to the digital art curriculum. *Studies in Art Education*, 46(1):61–73, 2004.

[5] Rachel Metz. Ai won an art contest, and artists are furious | cnn business, Sep 2022.

[6] Harold Cohen. The further exploits of aaron, painter. *Stanford Humanities Review*, 4(2):141–158, 1995.

[7] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. 2015.

[8] 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*, pages 4401–4410, 2019.

[9] 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*, pages 2414–2423, 2016.

[10] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[11] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM Computing Surveys (CSUR)*, 2021.

[12] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR, 2018.

[13] Yang Liu, Yao Zhang, Yixin Wang, Feng Hou, Jin Yuan, Jiang Tian, Yang Zhang, Zhongchao Shi, Jianping Fan, and Zhiqiang He. A survey of visual transformers. *arXiv preprint arXiv:2111.06091*, 2021.

[14] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering

text-to-image generation via transformers. *Advances in Neural Information Processing Systems*, 34:19822–19835, 2021.

[15] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[16] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjrn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[17] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[18] Eva Cetinic and James She. Understanding and creating art with ai: Review and outlook. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 18(2):1–22, 2022.

[19] Rui Xu. Discussing the aesthetic emotion of artworks by ai and human artists with the mediating variable of aesthetic fluency. In *International Conference on Human-Computer Interaction*, pages 84–94. Springer, 2021.

[20] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[21] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

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

[23] Ardavan Bidgoli, Manuel Ladron De Guevara, Cinnie Hsiung, Jean Oh, and Eunsu Kang. 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)*, pages 412–418. IEEE, 2020.

[24] He Feng. Analysis and research on the techniques of digital painting creation. In *International Conference on Frontier Computing*, pages 676–679. Springer, 2022.

[25] Dan Scott and Sahriyar Selim. Blocking in, Jun 2022.

[26] List of art techniques, Aug 2022.

[27] Zhengxia Zou, Tianyang Shi, Shuang Qiu, Yi Yuan, and Zhenwei Shi. Stylized neural painting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15689–15698, 2021.

[28] Raphael Gontijo Lopes, David Ha, Douglas Eck, and Jonathon Shlens. A learned representation for scalable vector graphics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7930–7939, 2019.

[29] Alexandre Carlier, Martin Danelljan, Alexandre Alahi, and Radu Timofte. Deepsvg: A hierarchical generative network for vector graphics animation. *Advances in Neural Information Processing Systems*, 33:16351–16361, 2020.

[30] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[31] Jules Françoise, Sarah Fdili Alaoui, and Yves Candau. Co/da: Live-coding movement-sound interactions for dance improvisation. In *CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2022.

[32] Pascal Barla, Simon Breslav, Joëlle Thollot, François Sillion, and Lee Markosian. Stroke pattern analysis and synthesis. In *Computer Graphics Forum*, volume 25, pages 663–671. Wiley Online Library, 2006.

[33] Aaron Hertzmann. A survey of stroke-based rendering. Institute of Electrical and Electronics Engineers, 2003.

[34] David Freedberg and Vittorio Gallese. Motion, emotion and empathy in esthetic experience. *Trends in cognitive sciences*, 11(5):197–203, 2007.

[35] Helmut Leder, Benno Belke, Andries Oeberst, and Dorothee Augustin. A model of aesthetic appreciation and aesthetic judgments. *British journal of psychology*, 95(4):489–508, 2004.

[36] ARTHUR C HARDY. Minutes of the thirty-first meeting of the board of directors of the optical society of america, incorporated. *Journal of the Optical Society of America*, 38(7), 1948.

[37] James D Foley and Andries Van Dam. *Fundamentals of interactive computer graphics*. Addison-Wesley Longman Publishing Co., Inc., 1982.

[38] International Color Consortium et al. Image technology colour management-architecture, profile format, and data structure. *Specification ICC. 1: 2004-10 (Profile version 4.2. 0.0)*, 2004.

[39] John Zimmerman, Jodi Forlizzi, and Shelley Evenson. Research through design as a method for interaction design research in hci. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 493–502, 2007.

[40] Ilpo Koskinen, John Zimmerman, Thomas Binder, Johan Redstrom, and Stephan Wensveen. *Design research through practice: From the lab, field, and showroom*. Elsevier, 2011.

[41] Korrapati Sindhu and Karthick Seshadri. Text summarization: A technical overview and research perspectives. *Handbook of Intelligent Computing and Optimization for Sustainable Development*, pages 261–286, 2022.

[42] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Ruifeng Deng, Xin Li, Errui Ding, and Hao Wang. Paint transformer: Feed forward neural painting with stroke prediction, 2021.

[43] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[44] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 1451–1460. Ieee, 2018.

[45] Rongyu Zhang, Lixuan Du, Qi Xiao, and Jiaming Liu. Comparison of backbones for semantic segmentation network. In *Journal of Physics: Conference Series*, volume 1544, page 012196. IOP Publishing, 2020.

[46] Guido Salimbeni, Frederic Fol Leymarie, and William Latham. Generative system to assist the artist in the choice of 3d composition for a still life painting. In *Machine Learning for Creativity and Design (NeurIPS 2019 Workshop)*, 2019.

[47] Nadav Cohen, Yael Newman, and Ariel Shamir. Semantic segmentation in art paintings. In *Computer Graphics Forum*, volume 41, pages 261–275. Wiley Online Library, 2022.

[48] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[49] Zhengyan Tong, Xiaohang Wang, Shengchao Yuan, Xuanhong Chen, Junjie Wang, and Xiangzhong Fang. Im2oil: Stroke-based oil painting rendering with linearly controllable fineness via adaptive sampling. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1035–1046, 2022.

[50] Rafael C Gonzalez, Richard E Woods, and P Prentice Hall. Digital image processing third edition pearson international edition prepared by pearson education. *Journal of Biomedical Optics*, 14:029901, 2008.

[51] Anil K Jain. *Fundamentals of digital image processing.* Prentice-Hall, Inc., 1989.