Sketch Style Recognition, Transfer and Synthesis of Hand-Drawn Sketches

Dissertation by

Sara Mohammed Shaheen

In Partial Fulfillment of the Requirements

For the Degree of

Doctor of Philosophy

King Abdullah University of Science and Technology

Thuwal, Kingdom of Saudi Arabia

June, 2017

EXAMINATION COMMITTEE PAGE

The dissertation of Sara Mohammed Shaheen is approved by the examination committee

Committee Chairperson: Prof. Bernard S. Ghanem
Committee Members: Prof. Perter Wonka, Prof. Wolfgang Heidrich, Prof. Karan Singh

ABSTRACT

Sketch Style Recognition, Transfer and Synthesis of Hand-Drawn Sketches

Sara Mohammed Shaheen

Humans have always used sketches to explain the visual world. It is a simple and straightforward mean to communicate new ideas and designs. Consequently, as in almost every aspect of our modern life, the relatively recent major developments in computer science have highly contributed to enhancing individual sketching experience. The literature of sketch related research has witnessed seminal advancements and a large body of interesting work. Following up with this rich literature, this dissertation provides a holistic study on sketches through three proposed novel models including sketch analysis, transfer, and geometric representation.

The first part of the dissertation targets sketch authorship recognition and analysis of sketches. It provides answers to the following questions: Are simple strokes unique to the artist or designer who renders them? If so, can this idea be used to identify authorship or to classify artistic drawings? The proposed stroke authorship recognition approach is a novel method that distinguishes the authorship of 2D digitized drawings. This method converts a drawing into a histogram of stroke attributes that is discriminative of authorship. Extensive classification experiments on a large variety of datasets are conducted to validate the ability of the proposed techniques to distinguish unique authorship of artists and designers.

The second part of the dissertation is concerned with sketch style transfer from one free-hand drawing to another. The proposed method exploits techniques from multi-disciplinary areas including geometrical modeling and image processing. It consists of two methods of transfer: stroke-style and brush-style transfer. **(1)** Stroke-style transfer aims to transfer the style of the input sketch at the stroke level to the style encountered in other sketches by

other artists. This is done by modifying all the parametric stroke segments in the input, so as to minimize a global stroke-level distance between the input and target styles. **(2)** Brush-style transfer, on the other hand, focuses on transferring a unique brush look of a line drawing to the input sketch. In this transfer stage, we use an automatically constructed input brush dictionary to infer which sparse set of input brush elements are used at each location of the input sketch. Then, a one-to-one mapping between input and target brush elements is learned by sparsely encoding the target sketch with the input brush dictionary.

The last part of the dissertation targets a geometric representation of sketches, which is vital in enabling automatic sketch analysis, synthesis and manipulation. It is based on utilizing the well known convolutional sparse coding (CSC) model. We observe that CSC is closely related to how line sketches are drawn. This process can be approximated as the sparse spatial localization of a number of typical basic strokes, which in turn can be cast as a non-standard CSC model that forms a line drawing from parametric curves. These curves are learned to optimize the fit between the model and a specific set of line drawings.

Each part of the dissertation shows the utility of the proposed methods through a variety of experiments, user studies, and proposed applications.

This dissertation is dedicated to

my parents,

Alia and Mohammed

&

my husband and son,

Ammar and Kenan

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to all the people who were beside me throughout the coursework of this dissertation. This work would not have been accomplished without the support of several great people around me for whom I dedicate this thesis. I would like to deeply thank Prof. Bernard Ghanem who gave me this opportunity in the first place. My deepest thanks go to him for believing in me and for having me as a member of his great IVUL team during the thesis work. It is with his kindness and support I managed to complete this work within an amazing research environment. What he taught me is way beyond what is written in this dissertation. My special thanks also extend to Adel Bibi who has never hesitated to discuss new ideas with me and to offer his mathematical support. I would like to also thank my friends and colleagues Lama Affara and Mohamed Ibrahim for the helpful discussion and excellent suggestions.

I am thankful for having the opportunity to learn from a professional group of faculty members, researchers, and students at the Visual Computing Center (VCC) at KAUST. Many thanks go to Tina Smith for her administrative help and cheerful smiles and for her kind encouragement. I would like to also thank the greatest team I can ever be working with, IVUL. They are a great example of hard working and dedication. Our group meetings were very special and kept me always up to date with new research directions. Our night by day working times before submission deadlines will be truly missed.

An honorable mention and appreciation should go to my friends in KAUST. I was lucky to have such exceptional close friends of mine: Emna Zidini, Imene Boudellioua, Amal Aboulhassan, Sawsan Al-Halawani, Doha Hamza, among many others. I would like to thank them for the sweetest joyful moments we spent together. I sincerely thank them for being there whenever I needed them and for being a valuable source of motivation with their great company.

My utmost gratitude, deep love and appreciation should go to my parents for their dedication and endless patience. I owe it to them for any achievement I have made in this life including this thesis. It is their inspiration and their ever lasting support that gave me all the strength I needed to move forward. I would like to deeply thank them for being such great examples for me to imitate and follow. Those few lines cannot adequately express my feeling of appreciation towards them. Thank you for always pushing me to pursue my dreams and for dedicating your lives in eliminating any obstacles that can threaten them. Thank you for your prayers and lovely hugs and for having faith in me. I also extend my thanks to my sisters: Wafaa, Ghadah, Raghdah and Fatimah and to my brother Abdul Aziz for their big love and support.

I would like to express my deepest appreciation and gratitude to my husband, Ammar. I would like to thank him for all the kind advice, for the great pushing and encouragement during the difficult times, and for being involved with his help and support in every detail of this journey from the beginning till the end. I would like to thank him for bearing with hearing my complains and for always being there when I needed him. You were the source of my strength. I would like to also thank my little one year old son, Kenan. Your arrival to this world in the middle of this journey was the greatest motivation for me. You brightened my life with your cute smiles. Thank you for bearing not having me around for many days during my studies. Words would never be enough to express my gratitude to my small family.

Lastly, I would like to thank all the people at KAUST, my friends and my family members for their help and support. Without their presence in my life, nothing would have been accomplished.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

This dissertation provides a holistic study of style in hand-drawn sketches. Hand-drawn sketches are a vital means to communicate ideas and thoughts and to demonstrate individual talent, ranging from children who initially express their understanding of their surrounding by simple sketches before words to professional artists who showcase their talents with outstanding sketches. Moreover, architects, engineers, and designers place their thought in the form of sketches first. Thus, utilizing modern computing powers and algorithms in designing models for automatic sketch understanding, analysis, and transfer will enable a wide range of applications that can enhance various aspects of an individual's personal and professional life. For example, an application that automatically detects fraudulent sketches will protect the copyrights of valuable sketches. Also, designing sketch training applications will minimize training time and effort and will provide better training capabilities. Moreover, automatic transfer of simple sketches to higher quality ones will result in better quality sketches with minimum effort.

Consequently, all of this has attracted researchers to develop techniques for automatic sketch analysis [1], recognition [2], beautification [3] and synthesis [4]. Figure 1.1 and Figure 1.2 show some example applications. This field has also been enriched by means of a number of commercial products, which were developed to assist artists and designers in generating professional sketches with minimum effort, such as Adobe Illustrator and Photoshop [5]. As the research demand in this field is increasing and to facilitate further research on sketches, a number of sketch datasets were made available to the research community ranging from portrait sketch datasets [4] to mechanical parts sketch datasets [6].

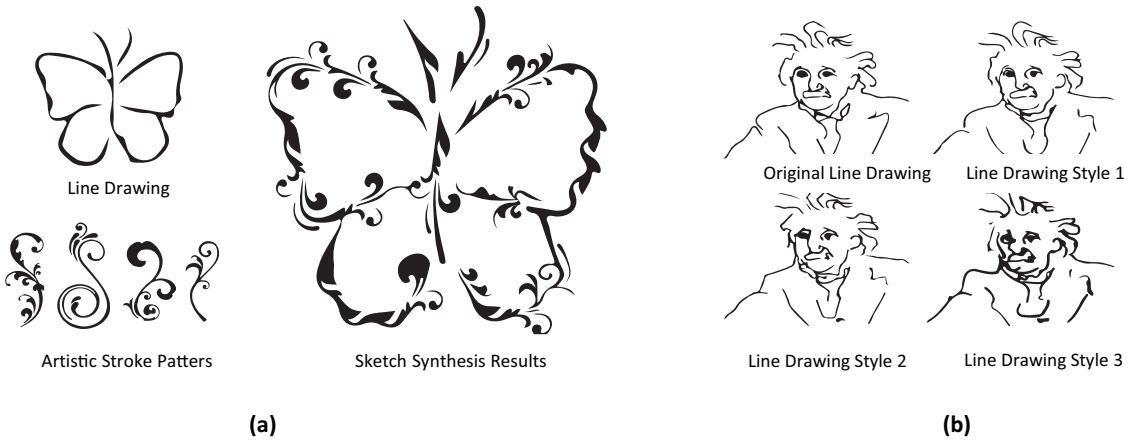


Figure 1.1: Examples of sketch based inspired research work. (a) Sketch beautification and brush synthesis [3]. (b) Sketch Style transfer at the stroke level [7].

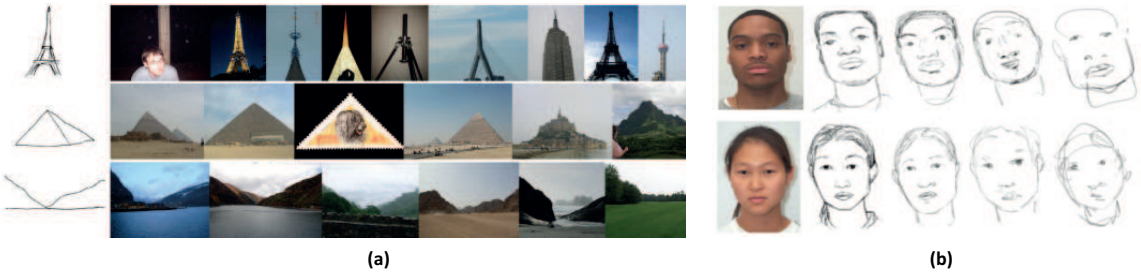


Figure 1.2: Example of sketch based inspired research work. (a) Sketch recognition [8]. (b) Sketch synthesis of portrait sketches with different levels of details [4].

The artistic style used to draw a sketch is a vital visual component of a sketch. Sometimes, sketch styles are recognizable to human observers and other times, when subtle differences exist between styles, they are not. However, in both cases, an artistic style is a difficult concept to rigorously define. To start any sketch manipulation or analysis process, a method to describe a sketch in terms of its stroke segments usually goes first in the development pipeline. For this purpose, sketches need to be represented as a set of parametric curves. This is known as sketch vectorization or digitization. It enables sketch transformations that are invariant to scale and nonrigid body transformations. Consequently, sketches are usually collected digitally using the Wacom device which through using a stylus pen and a pressure sensitive pad enables artists to directly render sketches digitally on their computer screens. As such, accessing stroke information of a sketch becomes a straight forward process. However, digital sketches require artists to go through special training on how to use such a device and many complain that it does not give them the freedom to express their artistic style as when drawing using a regular pen and paper. Alternatively, non-trivial geometric solutions of sketch vectorization were proposed [9], see Figure 1.3 for example.

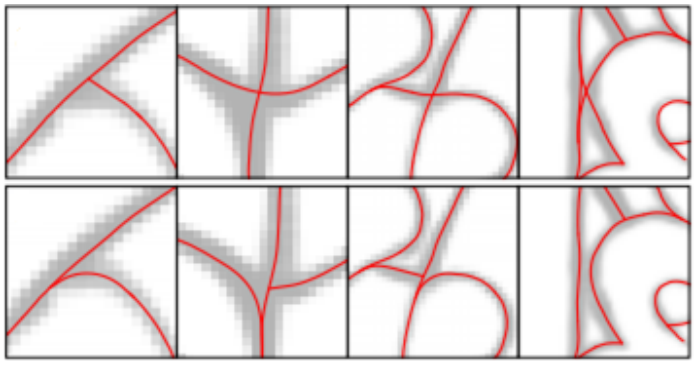


Figure 1.3: Examples of sketch vectorization. Sketch strokes are replace with the best fitting parametric curves [9].

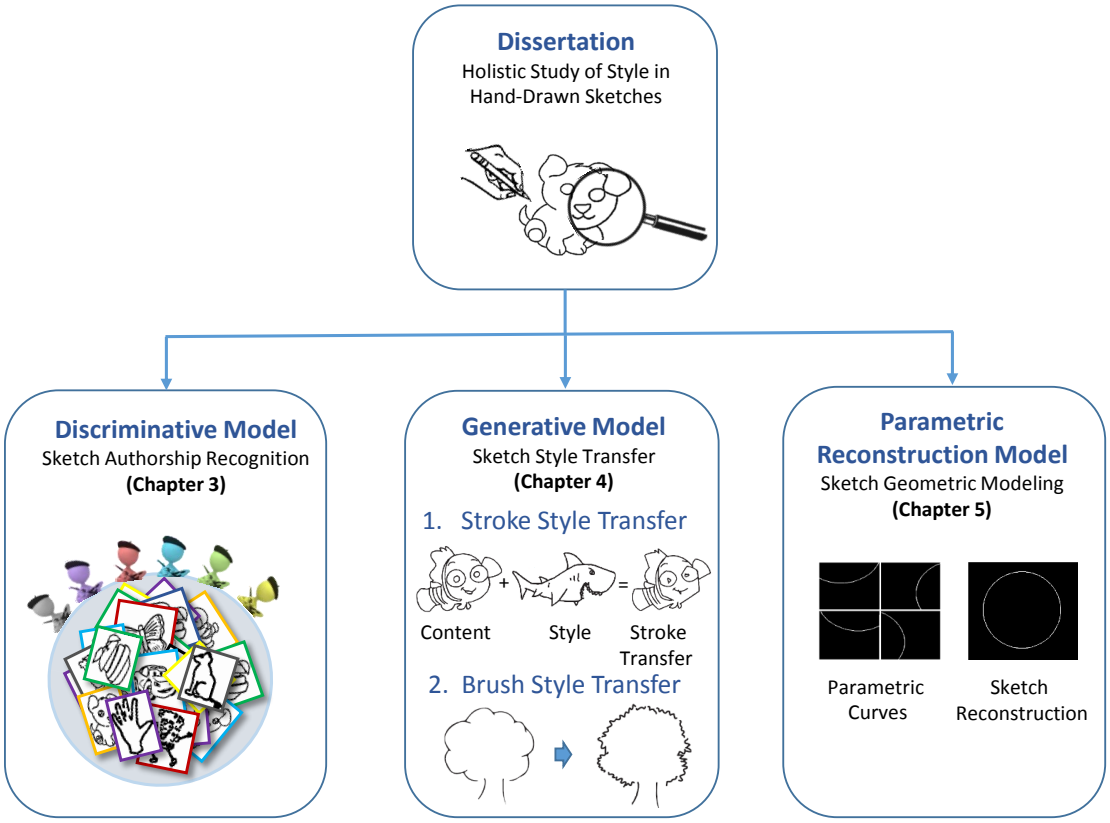


Figure 1.4: Dissertation outline. Sketch style is thoroughly studied in this dissertation by three connected models. The first model is a discriminative model that is concerned with sketch style analysis and authorship recognition of hand-drawn sketches. The second model is a generative model that is based on insights gathered from the discriminative model. It provides two levels of sketch transfer: stroke and brush levels of transfer to target both subtle and major sketch transformations. The third model provides automatic parametric sketch representation for geometric sketch modeling.

1.1 Dissertation Overview

In this dissertation, we aim to provide computational tools that target the analysis and transfer of sketch style in drawings. We show that a discriminative model built using 2D features can be very effective in discriminating between subtly different sketching styles and more effective than human performance on the same task. The discriminative model is entitled stroke authorship recognition. By building on concepts and insights gathered using this discriminative model, we formulate a generative model that enables transfer of a sketching style from one sketch to another. We entitle this generative model as sketch style transfer. It provides two different sketch transformation methods that enables for both subtle and major levels of sketch style transfer. The third model in this dissertation, contributes an automatic method for sketch parametric representation. This parametric reconstruction model is entitled as sketch geometric modeling. This model enables the geometric description of a sketch, which is vital for various applications related to sketch style manipulation, analysis, and synthesis. Together, these three models build a holistic study on the artistic style of hand-drawn sketches. The overall dissertation outline is depicted in Figure 1.4. Below, we introduce each of the aforementioned models and highlight the major technical assumptions, model work flow, applications and contributions to the field of each.

1.1.1 Sketch Authorship Recognition

In the first part of the dissertation (Chapter 3), we investigate whether individuals (specifically artists) are distinguishable by the way they draw, and if so, the extent to which this uniqueness can identify their drawings, so as to help train them or detect sketch fraud. This work is the first to address the problem of authorship recognition from drawings. Sketch authorship recognition (denoted as SAR) assumes that an artist's style can be recognized by the frequency distribution of certain mathematical descriptors, which are deduced from basic sketching strokes. A fundamental assumption is that artist style manifests itself math-

ematically in the artist's basic strokes. For example, we observe that Disney's Mickey Mouse tends to exhibit rounded strokes using many, nearly elliptic curves, whereas Looney Tunes' Daffy Duck consists of a combination of straighter and tightly curved strokes. By extracting enough strokes from a digitized drawing, a characteristic histogram is created. This histogram captures the inherent style of the overall drawing.

The proposed sketch authorship method is useful in a variety of applications such as sketch fraud detection (i.e. discriminating fraudulent from original sketches), where there is a need to examine fine-grained stroke-level features to discriminate sketches that *look* similar. This problem cannot be approached using existing shape matching techniques, which are unable to detect local and fine differences. Also, these methods are heavily dependent on the content of the sketch and not its style. Moreover, this authorship recognition technique can play a major role in artistic style training and reproduction, since it can be used to quantitatively assess how the sketching style of an artist in-training progressively becomes similar to a desired target style throughout the training process. For example, newly recruited artists at Disney are typically required to undergo a six month training procedure to familiarize themselves with the company's sketch techniques. This is done to ensure that new and existing characters can be created with the same Disney *look*. Clearly, there is a need for an automated technique to examine how an artist progresses during his/her sketch style training. Another area that can make use of this proposed work is handwriting verification and signature analysis, where the current state-of-the-art uses features that are centric/specific to handwriting and the focus is on the uniqueness of letters, punctuation formation, as well as, flow and structure. The proposed SAR method, on the other hand, provides a more general context to investigate the uniqueness of drawn strokes. This work is also useful to other areas including design patent litigation and brand marking. As subtle differences exist among sketches of different artists, especially when attempting fraud, we focus on examining low-level stroke features to discriminate between styles that are very similar. These features are empirically validated and biologically inspired.

1.1.2 Sketch Style Transfer

The second part of this dissertation (chapter 4) targets sketch style transfer and synthesis. Based on sketch authorship and style analysis (the discriminative model), this novel generative model is proposed. Transferring an input sketch to match another artistic style is useful in building a number of applications, such as sketch style training, where novice artists or professionals are guided while drawing to transfer their sketches to match another sketching style. Moreover, sketch style transfer can enhance the quality of sketches drawn by digital media to the level where they look as if they were drawn freely by an artist. Cartoon companies can adopt style transfer techniques to preserve the look of their characters by automatically updating how, for example, a newly employed artist is drawing to match the style of other senior artists.

Consequently, this part of the dissertation proposes an efficient technique for automatic sketch style transfer, which transforms a sketch (a query sketch) in such a way that it represents another artistic sketching style (of a target sketch), regardless of the sketch content. It consists of two transfer modules. The first module is called the *stroke-style transfer*, which focuses on transforming the query sketch at the stroke level. It is built based on the assumption that the subtleties of an artist's style can be recognized by analyzing the frequency of some representative geometric stroke-level features. This transfer operation is formulated as an optimization problem that seeks the best transfer parameters while being regularized to avoid degenerate solutions and to stay faithful to the original sketch content. The second transfer module focuses on *brush-style transfer*. It focuses on transferring the brush look of a stylized sketch (a target sketch with a unique brush look) to a line drawing (the query sketch). We find that convolutional sparse coding (CSC) [10] can model/approximate the sketching process of artists and, therefore, it is used in the brush-style transfer stage. This assumption is empirically validated in our work through a user experiment. Interestingly,

we cast brush-style transfer as a mapping problem between stylized brush elements and their corresponding abstractions using simple strokes. CSC is used here to synthesize a simple line-drawing that depicts the content of the target but using simple line strokes.

This transfer technique can be used to apply both subtle and major transformations on hand-drawn sketches. For subtle transformations, artists can use the stroke-style transfer module to get their sketches closer to a target style, which they are trying to mimic. For example, this is essential when a new artist is assigned to draw a well-known cartoon character by following the same style as another artist. Local stroke-level features are hard to mimic in this case, but they are crucial to maintaining the overall sketch characteristics of the cartoon character. On the other hand, major sketch transformations are made possible by our brush-style transfer module. With it, artists are able to automatically apply a unique brush look of a target sketch to their simple line drawings. This automatic transfer saves artists a painstakingly large amount of effort, when compared to creating this transferred brush look manually. The word *style* is interpreted in so many different ways in different contexts. Thus, each definition tackles the area of sketch synthesis and transfer differently leading to different results and serving different applications. Our proposed methodology in this dissertation adds a new perspective to the problem and defines sketching style to include both stroke-level and brush-level aspects of a sketch. With both modules, we aim to present a comprehensive technique for sketch transfer, which is feasible in various artistic sketch applications.

1.1.3 Sketch Geometric Modeling

The third part of the dissertation (Chapter 5) is on sketch vectorization, which is defined as representing a sketch as a number of connected parameterized curves. It is of great demand in the graphics community. It enables sketch transformations that are invariant to scale and non-rigid body transformations. It also fuels various applications related to

sketch style manipulation, analysis and synthesis [11]. Previous work on these topics tends to avoid the non-trivial geometric solutions of sketch vectorization [12] and replaces it with some manual interaction to access strokes of sketches (strokes are the building blocks of a sketch). This involves pre-processing steps such as building large libraries of strokes or digitally collecting sketches using the Wacome device [7, 13]. As such, utilizing existing modern computing algorithms for *automatic* sketch parametric representation is a valuable contribution to the graphics research community.

The proposed work here is an extension to the formulation of the well-known convolutional sparse coding (CSC) model [10], such that it can better model the reconstruction process of line drawings. This can be done by constraining the filters in CSC to a predefined group of parametric curves. This results in learned filters that, in their content, describe the sketch geometrically as a set of curves. Such representation can be utilized for sketch vectorization. We believe that this work is the first to introduce geometry constraints to the standard CSC formulation. Our choice of the CSC model and how it approximates the sketching process is validated through experiments and user studies, which are presented in detail in Chapter 5.

In summary, the work presented in this dissertation targets developing and experimenting with new techniques for efficient hand-drawn sketch manipulation. This includes sketch style analysis, transfer and geometric representation. Below, we present the major contributions of this work.

1.2 Contributions

This section presents a summary of the major contributions in this dissertation:

- Chapter 3 proposes a novel method for sketch authorship recognition based on the hypothesis that the collection of an artist's strokes in a sketch are unique to that artist and their frequency can be used to define his/her style. This can be used for multiple purposes including the detection of sketch fraud. Another contribution provided in

the sketch authorship recognition model is the compilation of two new sketch datasets collected from a number of experienced artists. These datasets are designed to expose the recognition method to different levels of challenges and sketch variations. They are made publicly available, along with their digitized form and source code, to allow for further research in this topic. Using the sketch authorship model two applications are included. The first provides artists in-training with immediate feedback on how close their sketching style has become to a particular target style and monitors their progress throughout the training process. The other provides the first quantitative and automatic measure to evaluate the quality of automatic sketch synthesis tools.

- Chapter 4 presents an automated and data-driven technique for sketch style transfer that enables sketch transformations at both the stroke- and brush-level, thus, allowing for an overall stylization of an input line drawing. Through the first transfer method, i.e.stroke-style transfer module, we show that it is feasible to transfer a sketching style of a line drawing to another (regardless of content) by adopting a global representation of the sketch as a whole. Such representations are usually used for sketch discrimination, as in Chapter 3. To the best of our knowledge, this is the first time it is used for sketch style transformation. The second transfer method, i.e.brush-level transfer module, can *automatically* discover detail-rich brush strokes, which are apt for transfer, as well as, their mappings to simple line strokes.

- Finally, Chapter 5 proposes a method for automatic parametric sketch representation inspired by the well known CSC model. The original CSC function is reformulated by adding a constraint on the learned filters to belong to a defined set of parametric curves. We solve the resulting non-convex optimization and validate its convergence through experiments.

1.3 Scientific Publications

+ While working on this dissertation, the following scientific papers were published: [11] and [14]:

- Sara Shaheen, Alyn Rockwood and Bernard Ghanem. **SAR: Stroke Authorship Recognition**. Computer Graphics Forum, 2016.

- Sara Shaheen and Bernard Ghanem. **Stroke Style Transfer**. Eurographics Association-Short Papers, 2017.

+ Finally, the following work is to be published at ICCV (2017):

- **Constrained Convolutional Sparse Coding for Parametric Based Reconstruction of Line Drawings.** With co-authors: Lama Affara and Bernard Ghanem.

Chapter 2

Related Work

This chapter of the dissertation surveys the work most related to the problem of the proposed research. We group them into seven main categories that are relevant to sketch style analysis, synthesis, recognition and vectorization. We also include work related to shape matching, handwriting and signature analysis and graphical authentication which are partially related to our work. Towards the end of this literature, we survey the well-known convolutional sparse coding (CSC) model which is a technique used in developing a vital part of this research problem.

2.1 Sketch Analysis and Transformation

In this section, we highlight previous work on sketch analysis, style transfer and synthesis that is relevant to our approach.

2.1.1 Sketch Analysis

In the work of Limpaecher *et al*. [15], a total of 13,000 drawings of faces was collected where their focus is on providing auto-correction of strokes for novice artists. Ulike the work prposed in this dissertation, they do not discuss the problem of stroke style authorship. Lu *et al*. [16], on the other hand, mimicked a particular artistic style via matching that was based on filtered velocities and shape context. Also, the approach of Cole *et al*. [6] studied where artists draw lines in sketches of objects such as tools, automobile parts and bones. They concluded that artists tend to draw similar lines in consistent locations. As part of

the experiments presented in chapter 3, we use the dataset of this work to show interesting new results regarding the uniqueness of sketch style despite strict sketching constraints imposed on the artists. More in sketch analysis, the work of Grimm *et al*. [1] provides suggestions for research directions in 3D sketching based on an observational study, where artists are asked to comment on how and why they made the marks on a number of objects they sketched. Also, a semi-automated method was developed Noris *et al*. [9] for semantic segmentation of digital sketch drawings to support further sketch analysis.

2.1.2 Sketch Transfer and Synthesis

The techniques of Hsu *et al*. [17] are among the first stroke synthesis methods in the literature of sketch style transfer. Following this seminal work, several more approaches have since been proposed and we group them into four categories.

Data-Driven Transfer Approaches. Freeman *et al*. [7] apply artistic style transfer by matching the k-nearest artistic drawing styles in a database comprising hundreds of strokes annotated in all possible styles and rotations. A lot of work on style synthesis has focused on human portrait sketches. For example, the method of Berger *et al*. [4] used a Wacom tablet and a stylus pen to collect a database of face sketches, drawn by a number of artists using reference face photographs. Each stroke is parameterized as a directed curve along with pen parameters. These strokes are used to build a stroke library that represents the sketching style of an artist and is then used to synthesize a portrait sketch in different artistic styles. In comparison to this method, our focus is to transfer styles regardless of the sketch content (i.e. to go beyond human portraits). Moreover, we transfer sketches drawn by any medium, either digitally or via scanned versions of paper sketches. Interestingly, we use the sketch synthesis dataset of [4] in this dissertation to demonstrate how our stroke authorship recognition (presented in chapter 3) can quantitatively evaluate sketch synthesis. Our results are on par with those reported by the authors after an online study with human participants. Moreover, the PortraitSketch system [18] allows auto-correction of strokes

for face sketches by novice artists. It automatically adjusts the geometry and stroke parameters (thickness, opacity, etc.). Unlike our work, their corrections are based on features taken from the underlying source image, which the novice artist draws on top of it. More recent work in face sketch synthesis and transfer is found in the work of Zhang *et al*. [19] and Wang *et al*. [20].

Building on the work of Berger *et al*. [4], Ben *et al*. [21] use their stroke library to render a line drawing (not necessarily faces) of the first frame of a video animation. However, their work focuses on promoting temporal coherence (through video tracking) to extrapolate the synthesis across consecutive frames rather than focusing on per-frame transfer results, which is the scope of our work. Moving to sketch paint synthesis, RealBrush [22] is a data-driven real median paint brush replicator, while HelpingHand [13] is a system for stroke synthesis based on filtered velocities and shape context from a library of 6D stylus trajectories. Decobrush [3], on the other hand, is a system that uses structured decorative patterns to allow synthesis of such patterns following user-sketched paths. Their synthesis is restricted to a number of defined styles from an example library, while our transfer method can use the style of any input sketch and transfer its unique strokes to other line drawings of different content (refer to chapter 4).

Example-Based Transfer Approaches. The proposed work of Hertzman *et al*. [23] is among the early methods of example-based sketch synthesis. They learn a statistical model of curves to synthesize new curves through a computationally intensive process. Moreover, in an attempt to investigate and define sketching styles, the work of Li *et al*. [24] transfers curve styles from examples of 2D curve shapes. Interestingly, it discriminates between style-revealing and content-revealing curves features. Their content-style separation analysis is based on a proposed feature-shape association matrix (FSM). Others used graph techniques for efficient stroke synthesis as the work of Kim and Shin [25].

Later work by Kalogerakis *et al*. [26] uses sketches learned by example to transform new drawings by hatching styles. Another work that synthesizes sketches using hatching styles is by Gerl *et al*. [27]. Using probabilistic models in the process of sketch synthesis has also been adapted by a number of recent approaches. Lang *et al*. [28] propose an autoregressive Markov model based on sampling over a series of learned feature distributions of curvature values. They explicitly have to handle non-pleasant scenarios, such as, overshoots and broken lines. More recently, ShipShape [29] is developed as a general sketch beautification assistant based on preserving geometric relations between strokes. Their method is based on evaluating different geometric rules on a new sketched path using the previously drawn and resolved paths. Transfer techniques in this dissertation consider features extracted from all strokes of the sketch and not each stroke independently, without the need for tracking paths during sketching. In Painting by feature [30], part of the work was dedicated for line feature where edges are stylized from an example sketch based on a a one-dimensional feature representing an arbitrary curvilinear structure. Clearly, each sketch synthesis and transfer approach starts by a unique definition of style and builds on this definition to perform transfer/synthesis, thus, including various pros and cons along the way. Our proposed work can be viewed as an addition to this rich literature, which sheds light on a new, systematic, and automated approach for sketch style transfer.

Texture- and Image-based Synthesis. Despite the fact that texture synthesis techniques suffer from scalability due to the textures' prescribed fixed resolution, Northrup *et al*. [31] and more recently Ando and Tsuruno [32] show that texture-mapped strokes are among the simplest and most efficient ways to stylize lines. Our brush-style transfer module discussed in chapter 4 is akin to example-based texture synthesis approaches as in Wei *et al*. [33]. However, to the best of our knowledge, non of the existing approaches utilizes automated methods to learn the brush strokes and how to transfer them to a line drawing.

It is worth mentioning that the seminal work of Kwatra *et al*. [34] is one of the pioneers in proposing texture optimization as an alternative method to pixel-based and patch-based algorithms, while combining the best of both worlds. As it synthesizes an output texture in the unit of pixels, it considers all pixels together when optimizing an energy function. Furthermore, Yang *et al*. [35] have proposed a convolution-based method for pencil rendering by developing a swing bilateral LIC (SBL) filter. Unlike our work where filters are learned from parametric information of stroke segments (as discussed in chapter 4), their filters are derived through image-based features such as pencil stroke flow, noise distribution and shades of grey colors in a sketch. They use multiple stylization approaches that are specific to the artistic nature of pencil-rendered sketches.

Synthesis through Detail Adaptation. Other approaches to sketch synthesis are concerned with depicting various levels of detail in their synthesis results. The level of detail can be on the elaboration side [36] or on the abstraction side [37]. The latter is one application we present in this dissertation, in chapter 4, as a promising use case for the adaptation of our brush-style transfer module. In our work, however, we aim to transfer strokes such that they represent other artistic styles present in other sketches and not to elaborate on or reduce the stroke set of a sketch. Another wide field of research, into which our work can partially fall, is the fairing, or neatening of strokes such as the work presented by McCrae *et al*. [38] and Simhon *et al*. [39].

Automatic stroke synthesis has been researched for a few decades. Thus, it is impossible to cover every work related to these topics. For a more detailed representation of the literature landscape, we refer the reader to two survey papers [40, 41]. They are concerned with artistic stylization techniques using 2D input images or videos of non-photo realistic rendering (NPR), as well as, sketch based image retrieval methods.

2.2 Sketch Recognition and Retrieval

Eitz *et al*. [2] develop an automated data-driven method to explore a large collection of hand-drawn sketches using drawings collected by many non-experts. Their primary goal is to represent sketch content to perform object recognition from a sketch and *not* sketch style. Concurrently, Sun *et al*. [42], propose a system for real-time recognition and retrieval of semantically meaningful attributes of hand-drawn sketches. Their work is not limited to pre-defined object classes. Other sketch retrieval methods are based only on geometric similarity between sketches as the work by Shrivastava *et al*. [43] and Eitz *et al*. [8]. Unlike our work, the field of sketch recognition and retrieval is based on classifying sketch content from a discrete number of semantic categories. They do not provide comparisons among similar sketches or across artistic styles and do not address the problem of authorship recognition.

2.3 Shape Matching

Sketch analysis and authorship determination seem akin to shape matching, but has different requirements as further inspection is needed. Numerous methods have been developed for shape matching and classification in the past as in Mokhtar *et al*. [44], Belongie *et al*. [45], Jin *et al*. [46] and Berg *et al*. [47], as well as, recently the work of Michel *et al*. [48] and Ion *et al*. [49]. Shape matching searches for similar shapes between two images, where one is usually considered the query image. The work in this dissertation differs from shape matching in two main aspects. First, shape matching focuses on global information of contours such as zero crossings of curvature, while the stroke analysis techniques, adopted in this research, segments the contour and studies detailed local features from each curve segment e.g. the eccentricity of a conic fit, as discussed in chapter 3. Second, shape matching is usually applied to low resolution images for computational reasons, where these images tend to be classified by their content, e.g., a set of different

mice silhouettes/shapes from different artists tend to belong to the same object class. Techniques developed in this research can be employed efficiently on images of various sizes, especially those at high resolution. Unlike shape matching, ours is less dependent on sketch content and more focused on the local intricacies of sketch style.

2.4 Forensic Handwriting Analysis

Forensic handwriting analysis is a well studied problem and entails a large body of previous work [50]. The aim here is to develop techniques that identify authorship by comparing hand-written samples of different people. Similarly, signature analysis and more precisely off-line feature based signature verification has also been studied for many years [51]. More recent work is by Kovar *et al*. [52] and Srihari *et al*. [53]. While all this work strives to prove the individuality of only handwriting and signatures [54] [55], we investigate the uniqueness of hand-drawn strokes in general (as discussed in chapter 3). As a result, features extracted in handwriting analysis are only applicable to that specific domain (e.g., height of a loop, pitch, baselines etc.) and not generalizable to sketch style analysis [56].

2.5 Graphical Based User Authentication

A number of graphical based user authentication methods such as doodle sketches were proposed as an alternative to conventional authentication methods. As an example, PassDoodle is a graphical based authentication mechanism, which attempts to identify users by their handwritten designs (doodles). For authentication, the query doodle is represented on a regular grid and matched to training doodles in order to determine user authenticity [57, 58]. The effectiveness of using PassDoodles for user authentication is demonstrated by Renaud *et al*. [59]. Oka *et al*. [60], on the other hand, develop a sketch-based authentication method to find the *closest* sketch in a training database by extracting edge orientation features from a user's query sketch. Unlike our stroke analysis techniques proposed in this dissertation (refer to chapter 3), these authentication methods neither build a discrimnation

model nor provide an analysis of different artistic styles. Moreover, they do not build an intermediate sketch representation of the user input which makes them similar to shape matching techniques discussed above. An exposition of the shortcomings of graphical based authentication methods is given in [61].

2.6 Sketch Vectorization

To start any sketch manipulation process, a method to describe a sketch in terms of its stroke segments is the first stage in the pipeline. An example of an automatic sketch vectorization method is by Noris *et al*. [12]. They propose an automated method to vectorize clean line drawings as connected Bézier curve paths. Their approach is based on extracting strokes centerlines as indicated from the image gradient along a stroke segment. This problem is not trivial to solve geometrically because even when input drawings are comprised of clean and high-contrast lines, inherent ambiguities make vectorization difficult. Consequently, researchers tend to rely on off-the-shelf commercial products, as what is found in Adobe illustrator [5]. Other approaches are based on collecting sketches digitally using the Wacom device, which makes accessing per stroke information and expressing it geometrically a straight forward process [4, 13]. Our proposed work provides a fully automated model to represent a sketch geometrically and to provide an automatic mapping to where strokes should be placed in the sketch utilizing the unique formulation of CSC (as discussed in chapter 5). This will enable more efficient and reliable sketch style analysis, synthesis and manipulation.

2.7 CSC Applications and Advancements

In this section, we highlight different applications and previous computational advances of the convolutional sparse coding (CSC) model. CSC is used in the research work presented in this dissertation for automatic parametric sketch representation, as discussed in chapter

5. Consequently, it is vital to highlight the CSC literature in this part of the dissertation. Research on CSC has taken two main directions. The first direction focuses on applying CSC to a wide range of computer vision problems such as image processing [62–66], designing new deep learning architectures [67], computational imaging [68], tracking [69] and structure for motion [70].

The other CSC research direction focuses on finding an efficient solution to the CSC problem. This direction is driven by the high computational demands of minimizing the non-convex CSC objective. Moreover, CSC sparse dictionary learning algorithms are special as they enable for diverse translation-invariant image patches through using the convolution operator in its image representation. Consequently, the literature witnessed seminal advances in CSC as found in [10, 71, 72]. Due to the diagonalization property of circulant matrices in the Fourier domain, speeding up the CSC solution has seen much progress. In fact, Bristow *et al*. [10] propose to model the optimization as two convex subproblems that are solved iteratively in a fixed point strategy. Each subproblem is solved in the Fourier domain using the Alternating Direction Method of Multipliers (ADMM) [73]. Following that, Bristow *et al*. [74] provide a thorough discussion on a number of optimization methods for solving convolution problems and their applications. To further improve efficiency, Kong *et al*. [71] describe an optimization approach that exploits the separability of convolutions across bands in the Fourier domain in order to make dictionary learning more efficient. Despite all of the mentioned advancements, the problem is still computationally heavy due to the high cost of solving large linear systems. Motivated by that, Heide *et al*. [72] propose a new objective function which transforms the original constrained problem into an unconstrained problem by encoding the constraints in the objective using some indicator functions. Their proposed objective is further split to a set of convex functions that are easier to optimize separately. Moreover, they propose a diagonal mask matrix to the objective to handle boundary artifacts that materialize in the Fourier domain. Final-

ly, Vorsel *et al*. [75] propose a non-iterative method in the Fourier domain for computing the inversion of the convolutional operator using the matrix inversion lemma. In this dissertation, we propose a novel method to facilitate sketch parametrization and geometric representation based on a reformulation of the CSC model. To the best of our knowledge, this dissertation is the first to introduce a constrained parametric model for CSC. Inspired by Heide *et al*. [72], we solve the constrained problem using the fixed point strategy, which leads to two subproblems that can be solved in the Fourier domain. Each subproblem is approached using ADMM. As mentioned earlier, our method facilitates designing sketch based computer graphics applications.

Chapter 3

Sketch Authorship Recognition (SAR)

3.1 Introduction

Research in voice and face recognition has developed methods to recognize and distinguish individuals based on pertinent visual and audio cues [76]. Inspired by this concept, this part of the dissertation investigates whether individuals (specifically artists) are distinguishable by the way they draw, and if so, the extent to which this uniqueness can identify their drawings, so as to help train them or detect sketch fraud. This part of the dissertation is the first to address the problem of authorship recognition from drawings. This new method, called sketch authorship recognition (denoted as SAR) assumes that an artist's style can be recognized by the frequency distribution of certain mathematical descriptors, which are deduced from basic sketching strokes. A fundamental assumption is that artist style manifests itself mathematically in the artist's basic strokes, For example, we observe that Disney's Mickey Mouse tends to exhibit rounded strokes using many, nearly elliptic curves, whereas Looney Tunes' Daffy Duck consists of a combination of straighter and tightly curved strokes. By extracting enough strokes from a digitized drawing, a characteristic histogram is created. This histogram captures the inherent style of the overall drawing. The details of the authorship recognition technique, results and applications will be discussed in details in the coming sections.

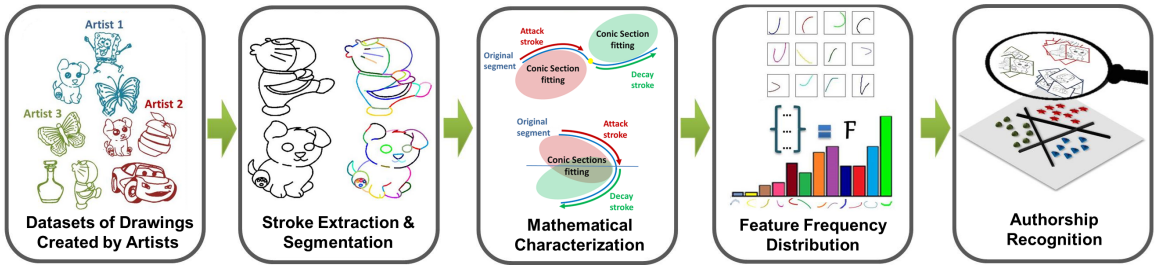


Figure 3.1: The SAR pipeline: 1. compile sketches from different artists; 2. extract strokes and split them into segments; 3. characterize each stroke segment mathematically 4. represent each sketch as a distribution of feature frequency; 5. train a machine to recognize authorship based on 4.

3.2 Sketch Authorship Recognition Approach

This section includes a detailed description of the proposed approach. The overall pipeline of SAR is depicted in Figure 3.1. Given a sketch image, the first stage in SAR is to extract major stroke contours in the image and segment these strokes into stroke segments. These strokes occur both at the silhouette and the interior of the sketch. Stroke segments from many sketches across multiple artists are grouped (using low-level image features) to form a universal dictionary of stroke segments. This dictionary is employed in a hierarchical bag-of-words model, which is used to represent a sketch image $\mathbf{I}$ as a histogram of stroke segments. This stroke histogram encodes some of the characteristics of an artist's unique style and thus can be used to discriminate this artist's sketches from others. Discrimination is performed in a supervised manner using multi-class classification, where each class designates an artist. An elaboration of the individual stages of SAR is presented in what follows.

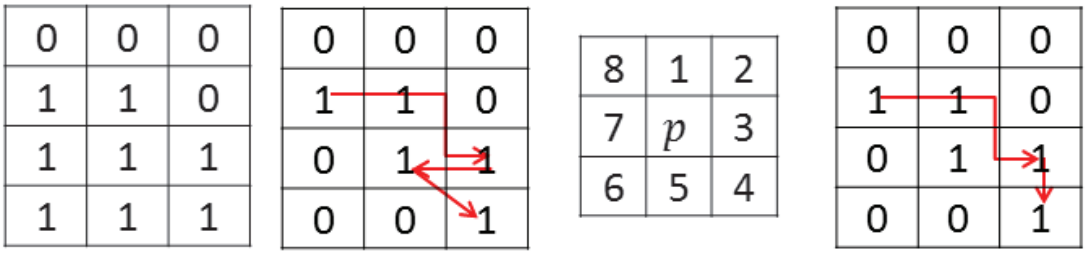
3.2.1 Stroke Extraction and Segmentation

Image $\mathbf{I}$ is decomposed into strokes that are further split into stroke segments that can expose authorship. Representing sketches at such a local scale can identify stroke segments that play an essential role in discriminating authorship.

Stroke Extraction. We use Adobe Live Trace for stroke extraction. It is an off-the-shelf digitization technique, to decompose a sketch into a set of paths, each of which comprises a number of Bézier curves depicting artistic strokes [5]. There exists a number of other commercial and non-commercial digitization techniques, such as the recent work by Noris et al. [12]. However, Adobe Live Trace is chosen in SAR, since it stays faithful to the original sketch and it is widely accessible and easy to use. With Adobe Live Trace, different levels of digitization can be applied to the original sketch. Using this tool, all the digitization parameters are kept [1] the same across all sketches of all artists, excluding the grey threshold value, which is varied among artists. This variation is needed because artists used different types of pens with different pressure. In Section 3.5.2, the effect of this digitization is evaluated on sketch style recognition. Extracted strokes are then traced and sampled as pixels in the image for stroke segmentation. It is worthwhile to note here that pixel tracing along edges has to be performed in a non-trivial manner, so as to avoid visiting the same pixels multiple times and adding redundancy to the trace.

Pixel Ordering. Both boundary and internal strokes segmentation requires contiguous pixels for representing each stroke. Usually, a pixel loop can be connected by starting from one pixel and ending at the same pixel. Connection means that each pixel has at most two neighbors in an 8-connected neighborhood, so that pixels can be tracked one after another. The Canny operator is the most widely used method for edge extraction. However, it can result in more than two neighbors for one pixel in an 8-connected neighborhood, which leads to redundant lines. This situation is shown in Figure 3.2(b), in which the edges are detected by the Canny operator from Figure 3.2(a). To solve this problem, we use a template to search neighbors in a fixed order, along with a tag vector to avoid repeat visits. Let the set of edge pixels be denoted by $P_i, i = 1, ..., l$, where each pixel $P_i = [P_i(x), P_i(y)]$, and l is the number of pixels. The process of ordering pixels connects independent pixels

[1] The values of these parameters are set as follows, Blur = 0 px, Max Stroke Weight = 30 px, Min Stroke Length = 10 px, Path Fitting = 50 px, Corner Angle = 20 degrees and noise = 2 px.

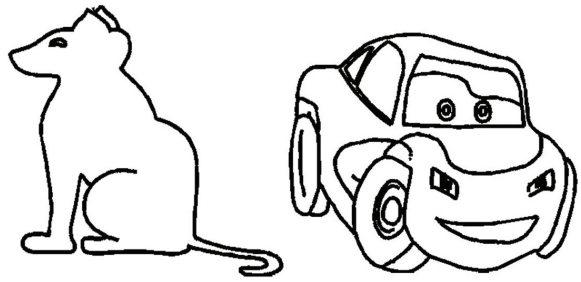


(a) (b) (c) (d)

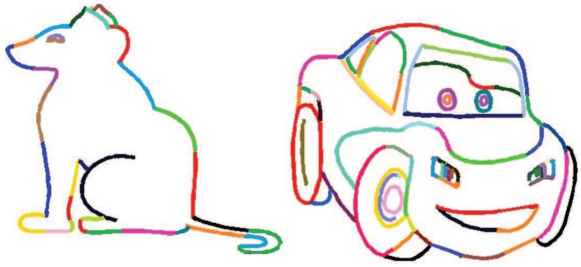
Figure 3.2: Examples of edge pixels. (a) 16 pixels from an image (d), where 0 is the background pixel and 1is the silhouette pixel; (b) an edge detected by the Canny operator from (a), where 1 represents edge pixels, red arrows indicate possible pixel order leading to multiple lines; (c) searching order of pixels, the middle p is the current pixel, the order of searching is labeled from 1 to 8. (d) the revised searching order indicated by red arrows.

P_i according to their 8-connected neighborhood. Let the reordered pixels be placed in a $l \times 2$ list L, where L_1 is the starting pixel, $L_{i+1}$is the next neighbor to L_i. Initially we choose an arbitrary pixel P_k as the starting pixel L_1, set $L_1 = P_k$, and mark pixel P_k as visited. Then iteratively search the next neighbor L_{i+1} of L_i according to the searching order given in Figure 3.2(c). Suppose L_i is located at p, starting from position 1, the first unvisited P_u is taken as L_{i+1}. The ordering process stops when the searched $L_{i+1} = L_1$. Figure 3.2(d) shows a revised search which avoids the polyline.

Stroke Segmentation. Strokes extracted from **I** can be quite long and contain a rich amount of geometric information. Modeling these strokes as a whole is a difficult task in itself, since it should encode variations that a particular stroke can take on. Instead, each stroke is broken into smaller units (called segments) that are represented in a more straightforward and conventional manner. A b-spline curve is fitted to each stroke and i-dentify break points as locations of local maximum curvature in the b-spline. Linear strokes are explicitly handled by placing break points at its two ends to avoid over-segmentation. The pixels between two consecutive breakpoints (or the beginning/end of the stroke) are grouped together and denoted as a stroke segment. Figure 3.3 shows stroke segments extracted from two sketches taken from one of our sketch datasets.



(a)



(b)

Figure 3.3: Stroke segmentation. (a) shows two sketches after digitization and (b) shows their extracted stroke segments (color-coded)

3.2.2 Mathematical Characterization

After **I** is decomposed into stroke segments, it is represented according to its stroke content. The aim is to describe a stroke segment's structure and the manner in which it is drawn. This part of the work focuses on local features that are simple to extract, representative, and invariant to various deformations (e.g. rotation, translation, and scale). Each segment is described by four simple features that encode eccentricity, symmetry, local consistency, and inflection. The first three features describe the stroke segment itself, while the last one describe its relationship to its neighbors. These features are both empirically validated and biologically inspired. When considering the process needed to analyze drawing traits at the stroke level, one fundamental factor arises, namely the neuro-geometry of hand-arm movement, i.e. the physiology in drawing strokes. In the foundational paper of [77], it was shown that different subjects produce hand/arm movements that are very similar to or coincide with simple curve strokes. Although low curvature change was generally maintained across different subjects and experiments, it was also found that individuals exhibited u-

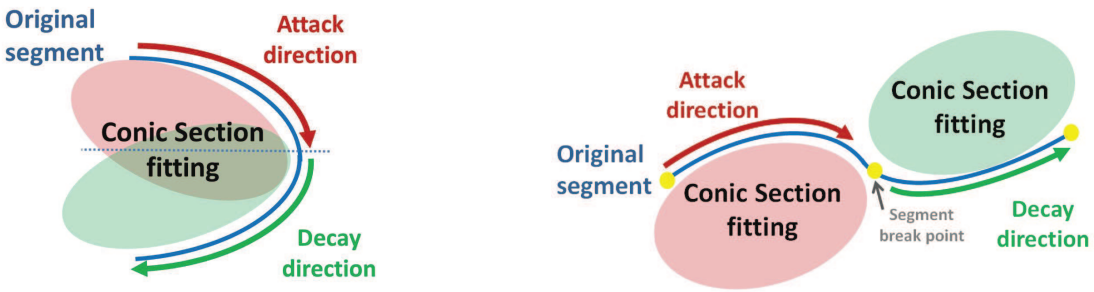
nique stroke characteristics. This finding was also confirmed in the work of [78]. In [79], a mathematical model based on minimizing jerk (change of curvature) and analyzing stroke velocity was verified empirically. In summary, this and other work (see references in [79]) strongly suggest that **(1)** gesturing (i.e. the physical mechanism of drawing) and strokes can be unique and identifiable for an individual, and **(2)** that curvature and its change are key features for analyzing simple strokes. Inspired by the above physiological findings, stroke segments are characterized by focusing on curvature and its change in each stroke segment.

Moreover, in analyzing a large number of simple strokes, it has become obvious that a conic fit to a stroke segment is faithful to its original geometry and representative enough for the purpose of authorship recognition. From each conic, eccentricity, symmetry, local consistency, and inflection features are extracted. They are conveniently invariant to rigid-body deformations. Needless to say, this invariance also holds when the entire sketch is represented using stroke segment features. Next, is a description of those four features.

Eccentricity: This feature represents the change of curvature in a stroke segment. It measures how the segment deviates from being circular, i.e. how rounded or sharp it is. By fitting a conic section to the stroke segment [80], eccentricity (ε) is computed as a function of the ratio between the conic's major and minor axis lengths as in (3.1).

$$\varepsilon = \begin{cases} \sqrt{1 - \frac{b^2}{a^2}} & \text{if the conic is an ellipse} \\ \sqrt{1 + \frac{b^2}{a^2}} & \text{if the conic is a hyperbola} \end{cases} \tag{3.1}$$

Symmetry: To crudely evaluate how symmetric a stroke segment is, the absolute difference in eccentricity of the segment's two halves is taken (refer to Figure 3.4(a)). The segment is divided into two equal length parts (denoted as *attack* and *decay* following typical sketch nomenclature) and the eccentricity of each part is computed as described above.



(a) (b)

Figure 3.4: Attack and decay parts used to construct stroke segments. (a) shows a stroke segment divided into equal length parts enabling the computation of the symmetry feature. (b) shows two adjacent stroke segments connected by an inflection breakpoint. The inflection feature is computed by comparing these two segments.

Local Consistency: This feature measures the extent of local variation within a stroke segment. It is computed as the average absolute difference between the eccentricity of the entire segment and the eccentricity of distinct overlapping parts of this segment. To generate these distinct parts, a sliding window approach is employed across the segment, where the window size is one third the size of the segment itself and the step size is half the window size. This feature captures subtle changes in shape and curvature.

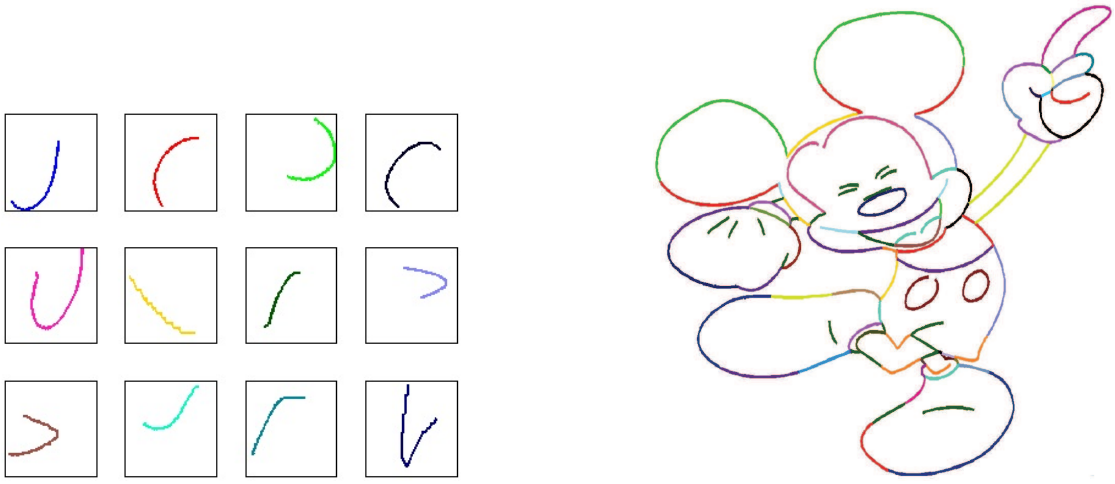
Inflection: The previous three features describe the stroke segment itself. However, characteristics of artistic style are encoded in how stroke segments are sequenced. Of special interest are locations of inflection, where the sign of curvature changes. An inflection point is detected by SAR as a breakpoint between two stroke segments that together form a stroke (refer to Figure 3.4(b) for an example). To encode such relational information at an inflection point, we compute the absolute difference in eccentricity between each pair of stroke segments that share this inflection point. For segments forming a T-junction (i.e. a stroke segment with inflection points at either end), their average eccecntricity difference is computed. For stroke segments whose breakpoints are not inflection points, their inflection feature is set to a nominal value (-1).

3.2.3 Feature Frequency Distribution

Having each stroke segment s_i of sketch image $\mathbf{I}$ characterized by the 4D feature vector described above, this stage represents $\mathbf{I}$ as a distribution of frequencies of those features.

Building a Universal Stroke Segment Dictionary. The assumption here is that the frequency in which an artist uses particular types of strokes is a suitable indicator of his/her authorship. To formalize this observation, a dictionary of stroke segments that tends to be universal among different artists is compiled. The dictionary is learned on the stroke segments of the sketches that are used in training only. To construct a dictionary of n elements, hierarchical k-means clustering is applied on a large set of stroke segments. In our experiments, we set $n = 60$. This dictionay is denoted as the universal stroke segment dictionary as it tends to capture the most commonly used stroke segments among artists. This clustering step is the first stage of the bag-of-words (BoW) model popularly used to represent natural images for image classification [81]. Hierarchical k-means is used because it is more robust and less sensitive to the choice of n than traditional k-means clustering. And it generates a tree of n cluster centers (and not only a set of centers as in the case of k-means), which encodes the membership of any stroke segment at all levels of the tree and not just the leaves. To encode a stroke segment, one has to *traverse* this tree starting from the root and recursively select among its children the nearest cluster center in 4D feature space. As a result, each stroke segment is encoded as a binary membership vector of length n, where each value reflects whether the feature traversed the corresponding tree node or not. Figure 3.5 shows image examples of cluster centers in the universal stroke segment dictionary, as well as, the color-coded membership of each stroke segment in a sketch. Such assignment is independent of where a stroke segment exists in a sketch.

Bag-of-Words Features. After constructing the universal dictionary, a sketch is represented as a histogram of stroke segments following the traditional Bag-of-Words (BoW)



(a) (b)

Figure 3.5: (a) Some elements of the universal stroke segment dictionary constructed from the segments of the training sketches. (b) Color-coded nearest neighbor assignment of segments to dictionary elements.

framework. Sketch image $\mathbf{I}$ containing m stroke segments produces m binary membership vectors each of length n. These membership vectors are pooled together to produce the BoW feature vector $\mathbf{f_I}$. Gaussian weighted mean pooling is used [82] to encode the frequency in which an artist uses each element of the universal dictionary. As discussed later, coupling $\mathbf{f_I}$ with a discriminative model enables authorship recognition. Since the original 4 stroke segment features are invariant to rotation, translation, and scale, the BoW feature vector is invariant to these deformations [83].

Dataset	# Artists	# Sketches	# Strokes	Content Variety	Constraints Level
Free Style	10	100	5500	10 different sketching objects	No constraints (Draw this sketch)
Fraud	9	90	4900	10 different sketching objects	High constraints (Draw and simulate fraud)
Line Study	11	107	6000	Up to 12 different objects	Very high constraints (Re-draw over the original Image)

Table 3.1: A summary of datasets used in SAR. We compiled the first two and reused the line study dataset of [6].

3.2.4 Authorship Recognition

So far, a sketch image is represented by a sparse n-dimensional histogram depicting the frequency in which each type of stroke segment is used by the artist in the sketch. As such, we expect the BoW feature to possess enough discriminative power to determine authorship based on simple strokes alone. We validate this assumption empirically in Section 3.5.1. Given a training set of sketches labelled according to the artist who drew them, we build a discriminative model using the training BoW features. In order to reduce testing time and to pinpoint the most discriminative portions of the BoW feature, we employ a forward feature selection procedure, which greedily appends a single feature at a time *only* if this addition improves classification accuracy [84]. Based on our experiments, only a small subset of the n features is actually used for discrimination. Using these subsampled BoW histograms, we build a multi-class classifier to assign authorship to an unseen sketch image. For sketch fraud detection, we use a binary classifier (original vs. fraudulent) instead. We experimented with a variety of classifiers and found that an RBF (Radial Basis Function) kernel SVM and a kNN classifier achieve the best performance. However, we decided on the kNN classifier because of its minimal training time. We use two experimental setups to evaluate the accuracy of our classifier: 2-fold cross validation and leave-one-out. The first is a popular setup in image classification, while the latter sheds light on how dependent the classifier performance is to the amount of training data. Following common practise, the best k value for the kNN classifier is estimated using cross-validation on a finite set of possible k values.

3.3 Sketch Datasets

To evaluate the performance of SAR, we compile two new sketch datasets collected from experienced artists. Participating artists were screened, so as to guarantee high levels of sketching capability. The chosen artists were graphical or interior designers by profession,

each with 7-10 years of sketching experience. We gave them the freedom to draw using a pen, pencil or digitally at any scale. Moreover, they were allowed to correct their strokes or redraw the entire sketch with no time limit. Since we had direct access to these professionals, we were able to administer different sketching scenarios. This allowed us to control the *content variety* (i.e. what the artist draws in a sketch) and the extent of *sketching constraints* (i.e. how the artist should sketch). These two factors impact the strokes an artist chooses and ultimately his/her style. To our knowledge, this work is the first to compile such a diverse set of sketch data for the purpose of studying authorship from strokes. To allow for further research in this area, we make all these datasets (images and annotations) publicly available. Next, we describe the datasets used in this work (refer to Table 3.1 for a summary).

Free Style Dataset. We collected 10 images of objects from diverse semantic categories and then asked 10 artists to sketch them. The objects were chosen to be detailed enough to adequately reflect artistic style and to be relatively easy to replicate by an experienced artist in a reasonable amount of time. On average, it took each artist 2 weeks to submit the 10 sketched objects. In total, the 10 artists collectively drew 100 sketches. No constraints or specific instructions were given to the artists, thus, giving them complete sketching freedom.

Fraud Dataset. Here, we simulate a sketch fraud scenario. Using the same images in the free style dataset, we chose the sketches of one of the artists as *original* drawings. We asked the other 9 artists to draw all the original sketches. We provided them with an instruction sheet, where they were requested to draw the original sketches in such a way that it would be very hard to distinguish their *own* drawings from the originals, thus, simulating sketch fraud. The 9 artists were prohibited from using methods or supplies (e.g. translucent tracing paper) that could produce copies of the original sketches. Each artist took an average of

2 weeks to finish the copying task. We collected a total of 90 new sketches. (refer to Figure 3.6 for examples). Since the artists were requested to simulate sketch fraud, this dataset is highly constrained from an artistic style point of view. Therefore, it poses a substantial challenge to any authorship recognition system, albeit manual or automatic.

Line Study Dataset. We also use the line study dataset generated in [6]. This dataset is a collection of sketches of various objects (e.g. mechanical tools, automobile parts, and bones) drawn by multiple artists. The sketching tasks were highly constrained, since artists were asked to draw their sketched lines over a faded copy of the original image. Clearly, the task of recognizing sketch authorship in this dataset is more difficult than the previous two. We selected all sketches from artists, who drew at least 6 sketches, thus, leading to a dataset of 107 sketches from 11 artists.

3.4 Human Sketch Authorship Recognition

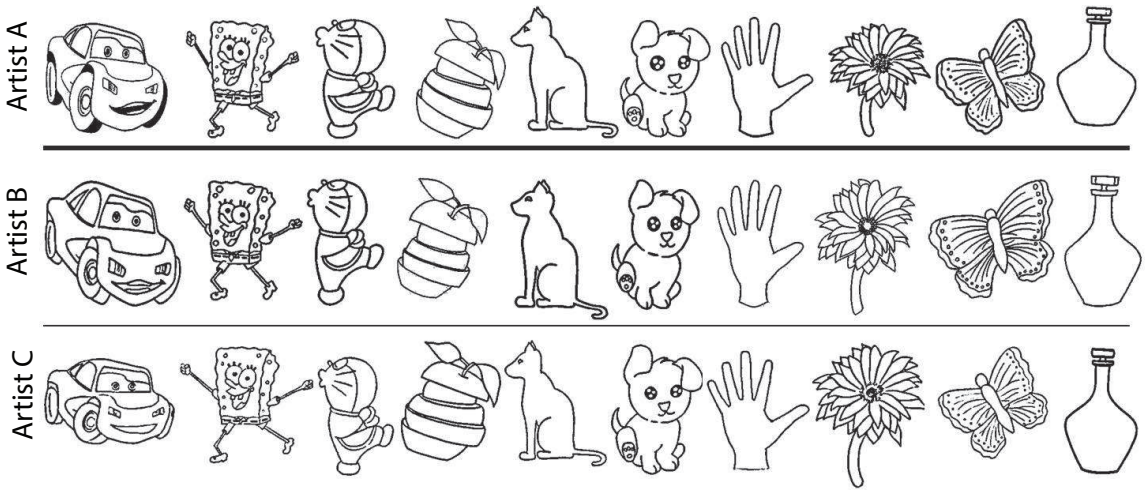


Figure 3.6: Samples of sketches from the fraud dataset. The first row shows the original sketches drawn by Artist A. The other two rows show the sketches of 2 other artists (Artists B and C), who attempted sketch fraud. Obviously, the *fraudulent* sketches look extremely similar to the originals, making authorship recognition (manual or automatic) quite difficult.

For comparison and as a baseline, we study the inherent difficulty of the authorship

recognition problem for humans. The human visual system (HVS) is renowned for its effectiveness in successfully completing many high-level recognition tasks (e.g. object and action recognition), so much so, that it remains the *gold standard* to which automated recognition systems aspire. Despite significant advances in computer vision, the HVS almost always outperforms automated methods in such tasks. However, there *do exist* recognition tasks, in which the HVS underperforms. These tasks usually deal with *fine-grained* recognition of objects (e.g. faces), where the inter-class variation is minimal and on par with the intra-class variation. We motivate this fact with an example. Although the HVS can easily discriminate between a 'chair' and a 'dog', it does not do so well in recognizing a person's face in a large dataset of people (e.g. the entire population of a country) from the same cultural and racial background. The differences between people's faces are so subtle that the minute details discriminating them are hard to uncover by the HVS. However, it is exactly these details that automated recognition methods focus on. This enables them to outperform the HVS in these tasks (e.g. robust face recognition [85]). In this section, we provide extensive empirical evidence from two user studies that highlight the sheer difficulty people and experienced artists face when trying to recognize authorship from 2D sketches.

Authorship Recognition by Non-Artists. The aim of this study is to shed light on how accurately people can recognize sketch authorship. As an online quiz, participants are first shown sample sketches from the free style dataset labeled with artists who drew them. We showed 5 randomly selected sketches from each of the 10 artists. Next, we administered for each online participant a total of 5 questions, each of which asked him/her to assign an *not previously seen* sketch to one of the 10 artists that he/she thought drew it. We chose to show users only 5 of the 10 images per artist so as not to overwhelm them and to enable a fair comparison with the 2-fold experiments we conducted using our automated method. We allowed participants to zoom-in to the images when needed. To reach a large number

of participants, we published our user study on Amazon Mechanical Turk (AMT). To validate the quality and seriousness of each Turker's answers (as usually done in practise), we administer control questions at the beginning of the quiz. Moreover, Turkers are randomly directed to one of the many versions of the online study and answers from unique workers are recorded.

After 6 weeks of activity and more than 2000 unique participants, the accumulated results of this study show that people can successfully recognize the authorship of a sketch (among 10 different artists) with an average accuracy of 29% and with standard deviation of 8%. Participants took an average of 4 minutes to complete the assigned quiz. We see that average human performance is moderately better than a random choice classifier (i.e. choose one of the 10 artists at random irrespective of what the sketch is) with an average accuracy of 10%. Obviously, this establishes that sketch authorship recognition is quite difficult for people. More importantly, we show later that our automated SAR method achieves an accuracy of 56% under the same conditions.

Sketch Fraud Detection by Artists. We conduct a similar study, but we target artists specifically because it is conceivable that the average person might find this task difficult due to his/her lack of sketching experience and/or artistic talent. The aim here is to evaluate the performance of artists in discriminating fraudulent sketches from original ones. A total of 25 experienced artists were given an online quiz, where 5 original and 5 fraudulent sketches were made known to each user. The sketches were taken from the fraud dataset described earlier. Each artist is then given a set of 5 recognition questions, whereby he/she needs to label the unseen sketch as original or fraudulent. Moreover, artists were given the chance to view their quiz results and to email us feedback.s

The quiz results show that the artists could distinguish original sketches from fraudulent ones only 47% of the time, as compared to 50% random chance. Each artist took an average of 5 minutes to complete the task. As expected, most of the feedback we received

elaborated on how truly difficult the task is. Later, we show that SAR significantly outperforms the artists' recognition accuracy, thus, motivating the plausibility of using SAR in an automated fraud detection system for sketches (e.g. patent drawings and cartoon sketches).

3.5 Experimental Results and Evaluation

This section presents a number of experimental results we obtained to assess multiple facets of the proposed SAR approach. **(1)** To evaluate the effect of sketching constraints on SAR, as well as, its overall effectiveness in recognizing sketch authorship and detecting sketch fraud, we test SAR on the datasets described in Section 3.3. **(2)** We also evaluate the sensitivity of SAR to a number of algorithmic variations. These results, in turn, are used to design the most representative and discriminative variant of SAR.

3.5.1 Computational Authorship Recognition

We ask the following questions: Are simple strokes unique to the artist who draws them? If they are, then to what extent can they identify the author who drew them? To our knowledge, these questions are not adequately addressed in prior work. To answer them, we conduct several experiments on three separate datasets, which provide a rich diversity of sketch content and sketching constraints. In what follows, we report SAR classification accuracy for each dataset and provide cross-dataset analysis. We also evaluate SAR's ability to detect sketch fraud using the fraud dataset.

Free Style Dataset. On this constraint-free dataset, SAR accuracy on test data is 56% and 53% using leave-one out and 2-fold validation respectively with 7% standard deviation. For comparison, random chance in this case is 10%. To verify the statistical significance of our classification results, we run a standard student t-test with a significance level of 95%. The null hypothesis, "SAR performs similar to a random classifier", is rejected 97% of the time (i.e. p-value is 0.03).

Fraud Dataset. Is artistic style among different artists still preserved when they are consciously attempting to commit fraud? To answer this, we use our fraud dataset to build a 10-class SAR classifier to discriminate among the 10 artists. Using this dataset, random chance is 10%. Here, SAR reaches a significantly higher average accuracy of 48% (leave-one-out) and 43% (2-fold cross validation) with 6% standard deviation. This result suggests that an artist's sketching style is still distinguishable (to a certain extent) even when attempting sketching fraud. To visualize the discrimination power of SAR, we show the BoW features of three sketches (same object) drawn by 3 artists from this dataset in Figure 3.7. One of the sketches is an original and the rest are fraudulent. Although all three sketches look very similar, their underlying features have obvious differences, which SAR capitalizes on to determine authorship. By conducting the student t-test with a 95% significance level, we conclude that SAR is more discriminative than a random classifier with a p-value of 0.04.

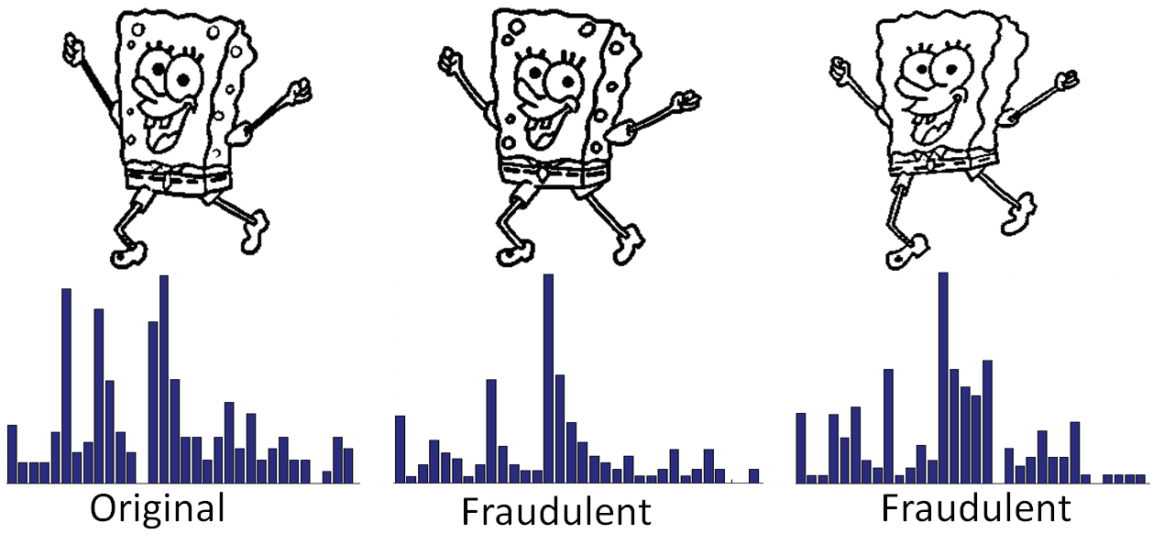


Figure 3.7: BoW features of the same sketch object drawn by 3 artists (one original and two fraudulent). Although they look very similar, their features are different.

Line Study Dataset. The sketching task in the dataset compiled by [6] is highly con-

strained as discussed in Section 3.3. As it is similar in spirit to the fraud dataset, we expect similar results. We train and test SAR in the same way as before but with a total of 11 artists and 107 sketches. With 11 classes, random chance is 9%, which is significantly lower than SAR's average accuracy of 33% (leave-one out) and 30% (2-fold) with 5% standard deviation. This noteworthy discrepancy in performance indicates that artists still maintain some uniqueness in their sketching style, even though they are forced to copy a different style altogether. The p-value of the student t-test in this case is 0.05.

We give a performance summary for SAR on the three datasets in Table 3.5.1. The slight difference in accuracy between leave-one out (i.e. all but one sketch are used for training) and 2-fold cross validation (i.e. only 50% of sketches are used for training) indicates that SAR is reasonably insensitive to the amount of training data used. This also suggests that SAR is generalizable, a promising property for a classifier in the presence of unseen data.

	Rand Chance %	Leave-one out %	2-Fold %	Standard Deviation
Free Style	10	56	53	7
Fraud	10	48	43	6
Line Study	9	33	30	7

Table 3.2: A summary of SAR performance on three datasets. Average accuracy (%) is reported for leave-one-out and 2-fold setups. Random chance accuracy (%) is also reported for comparison.

Cross-Dataset Analysis. In the datasets above, the contributing artists were subject to varying levels of sketching constraints, ranging from unconstrained (free style dataset) to highly constrained (line study dataset). To investigate the sensitivity of SAR performance with sketching constraints, we compare its accuracy across all datasets in Figure 3.8(a). The datasets are ordered in increasing levels of constraint. To normalize the effect of different numbers of classes across datasets, we plot the ratio of SAR accuracy to random chance. As expected, recognizing sketch authorship becomes harder as more constraints are imposed on the artist. However, on all the datasets, SAR performance is significantly

(at least 2.5 times) higher than random chance. This suggests that artists do *not* lose all characteristics of their unique style even under the strictest of constraints.

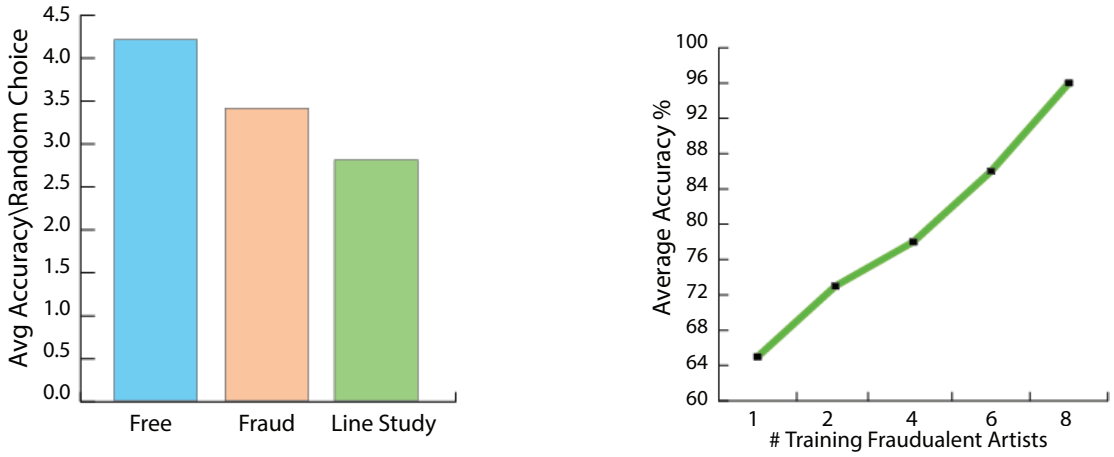


Figure 3.8: (a) The ratio between SAR accuracy and random chance on three datasets organized in increasing order of sketching constraints. Clearly, the more constraints imposed on artists, the less discriminative their strokes become. (b) Fraud recognition performance improves as more fraudulent artists are used in training, especially on sketches from artists who were not included in training.

Fraud Recognition Experiment. Here, we setup a fraud recognition experiment (original vs. fraudulent) that predicts SAR performance in a real-world scenario. It is unrealistic to assume that the SAR classifier will have access to fraudulent training examples from *all* artists. However, we expect that when more fraudulent samples are used in training, SAR's test performance on fraudulent sketches from artists, who were *not* included in training, will improve. In other words, knowing more about how fraud *looks* like will help SAR better detect sketch fraud, even for fraudulent artists whose sketches are not trained on. To validate this expectation, we train the SAR binary (fraud vs. original) classifier with fraudulent sketches from increasingly more fraudulent artists (i.e. from 1 to 5 artists). In each case, 50% of the original sketches are used for training (2-fold validation). Then, each SAR classifier is tested on the remaining original and fraudulent sketches. This means that SAR will always be tested on fraudulent sketches from artists, whose style has not

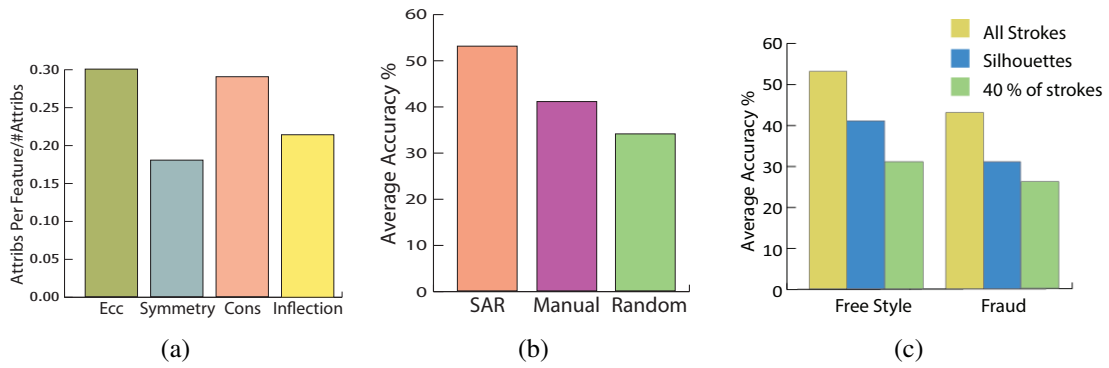


Figure 3.9: (a) Contribution of each feature type to SAR recognition. (b) SAR accuracy on the free style dataset when its segmentation method is replaced with a manual or random one. SAR accuracy drops by 15% when the segmentation is done manually and 20% when it is done randomly. (c) SAR accuracy on the free style dataset, when the stroke segments are taken to be all the segments in the entire sketch, all the segments in the silhouette of the sketch, or a randomly sampled fixed percentage of the segments in the entire sketch. In the last case, we only consider the percentage of segments in the sketch that are from the silhouette, which is estimated to be around 40% for each sketch. Adding internal stroke segments *does* improve accuracy but the improvement is only about 10%. Silhouette strokes are more discriminative than the same number strokes sampled from the entire sketch. This shows the discriminative power of the silhouette.

been seen during training. In Figure 3.8(b), we plot the average test accuracy of SAR as the number of fraudulent artists used in training increases. Clearly, this result endorses our aforementioned expectation. In fact, SAR's fraud detection performance reaches 96% when 8 of the 9 fraudulent artists are used for training and the 9^{th} for testing. Despite the extremely high similarity between fraudulent and original sketches, we conclude that SAR is able to effectively spot artistic fraud. Moreover, this result suggests that SAR can be used in an online active learning setup, where test samples are sequentially classified and in turn weighted and then used in re-training the classifier.

3.5.2 Algorithmic Details

Varying the details of the different computational modules of SAR (refer to Figure 3.1), leads to a variant of SAR. In this section, we investigate many of these variations and provide algorithmic details to reproduce the SAR classifier.

Feature Contribution. Stroke segments are represented by 4 biologically inspired features, as discussed in Section 3.2.3. Here, we study the contribution of each feature to SAR's discriminative power. To do this, we employ a conventional forward feature selection method that greedily determines which features (along with their importance weights) should be incorporated into the SAR classifier. This is a data-driven process, so different training-test splits of the dataset can lead to different feature selections and weights. By training SAR (with 2-fold validation) multiple times on the free style dataset, we accumulate an average selection weight for each feature (refer to Figure 3.9(a)). We see that eccentricity and local consistency are the most discriminative followed by inflection and symmetry with no one feature dominating.

Variations in Stroke Segmentation. To justify our stroke segmentation method (refer to Section 3.2.1), we compare it against two baseline methods: manual and random stroke segmentation. For manual segmentation, we asked 10 artists to manually segment sketches into strokes as they see fit, while in the random case, breakpoints are selected uniformly at random within each extracted stroke. We set a minimum length for each stroke segment to prevent singularities. SAR accuracy (2-fold) using each of the three segmentation methods on the free style dataset is summarized in Figure 3.9(b). Clearly, our automated segmentation method outperforms the other two. Similar to the human recognition result in Section 3.4, SAR improvement over manual segmentation lends more evidence to the fact that human performance is suboptimal in fine-grained tasks, such as authorship recognition.

Silhouettes Vs. All Strokes. In many cases (e.g. the Mickey Mouse cartoon character), the silhouette of a sketch can be very discriminative, so much so, that silhouettes have been used as primary features in other recognition tasks (e.g. action recognition [86]). Here, we study how discriminative silhouette stroke segments are on their own within the SAR pipeline. In Figure 3.9(c), we use two datasets to compare SAR accuracy when the stroke

segments are taken to be all the silhouette segments, all the sketch segments, or a sample set of the the sketch segments, where the sample size is based on the percentage of silhouette stroke segments in the sketch. This percentage is 40% on average. Interestingly, silhouettes seem to be quite discriminative in their own right, as SAR performance remains high even when only silhouettes are used. In fact, including stroke segments from the sketch interior only improves accuracy by about 10%. When comparing silhouette segments to the same number of segments sampled from the entire sketch, we see that the former is much more discriminative, thus, validating the discriminative power of a sketch's silhouette.

Effects of Digitization. As mentioned in Section 3.2.1, stroke segments are extracted from a sketch using the well-known Adobe Live Trace digitization technique. The level of digitization is controlled by the user to tradeoff faithfulness to the original sketch with compactness of representation. For instance, high levels of digitization tend to smooth out strokes, thus, masking aspects of the artist's intrinsic style and possibly affecting SAR performance. To investigate the effect of digitization on SAR, it is applied to the fraud dataset after applying three distinct levels of digitization. This is done by varying the path fitting parameter in Live Trace. We report the average accuracies due to this parameter variation in Table 3.3. As expected, the higher the digitization level), the lower the SAR accuracy is. This result suggests another application that could benefit from SAR, namely the quantitative assessment of various digitization techniques. SAR can conceivably be used to evaluate how much of the unique sketching style is preserved after digitization.

Style not Content. SAR is designed to represent and classify artistic sketch style, while not being significantly sensitive to sketch content in general. This is a primary factor that distinguishes SAR from existing shape matching techniques discussed in chapter 2. In fact, our experiments show that sketches of different content (objects) drawn by the same artist tend to have more similar BoW features than sketches of the same content drawn by differ-

level of digitization	leave-one-out	2-fold cross validation
Low	**48**	**43**
Medium	40	36
High	28	23

Table 3.3: Effects of digitization on SAR accuracy. Sketches from the fraud dataset are processed using three levels of digitization in Adobe Live Trace. SAR accuracy decreases significantly with higher levels of digitization.

ent artists. For instance, the BoW feature of a flower drawn by one artist is more similar to a butterfly drawn by that artist than the feature of the same flower drawn by another artist, as illustrated in Figure 3.10. To quantify this observation, we use a normalized Gaussian similarity measure $s(\mathbf{I}_1, \mathbf{I}_2)$ based on the Euclidean distance between two BoW features. From the free style dataset, we randomly select a sketch $\mathbf{I}_i^A$ drawn by artist A and the same sketch $\mathbf{I}_i^B$ drawn by another artist B to compute $s(\mathbf{I}_i^A, \mathbf{I}_i^B)$. We then randomly select a sketch $\mathbf{I}_j^A$ drawn by A with $i \neq j$ and compute $s(\mathbf{I}_i^A, \mathbf{I}_j^A)$. By performing this operation across the whole dataset, we compute the average inter-artist similarity $s(\mathbf{I}_i^A, \mathbf{I}_i^B) = 68\%$ and the average intra-artist similarity $s(\mathbf{I}_i^A, \mathbf{I}_j^A) = 89\%$. This result shows that SAR can abstract an artist's style from different sketch content. We would like, however, to emphasize that strokes are not completely independent from the content. For example, SAR cannot recognize the authorship of sketches of a single square if an author's training data *only* includes circles. Therefore, SAR inherently assumes that the training data contains non-trivial sketches with sufficient stroke diversity characterizing the overall style of any particular artist.

3.6 Applications

In this section, we show SAR's feasibility in two useful applications other than sketch fraud detection, namely sketch style training and sketch synthesis evaluation.

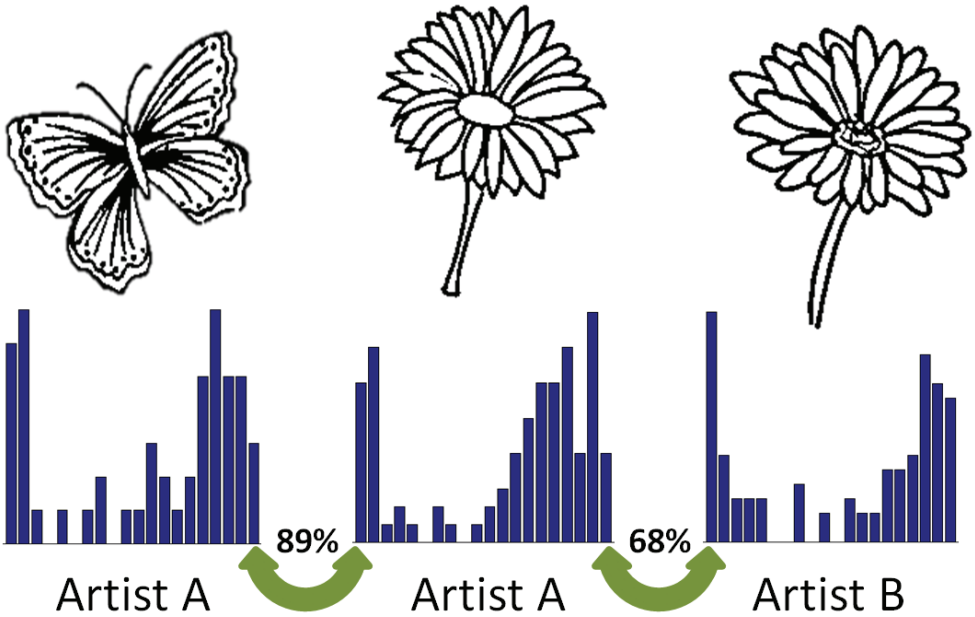


Figure 3.10: A visual comparison between the BoW features of three sketches from the free-style dataset: a flower drawn by Artist A (middle), a butterfly drawn by Artist A (left), and the same flower object drawn by Artist B (right). From the features themselves and their average pairwise similarity scores, we see that SAR discrimination abstracts artistic style and is not significantly affected by sketch content.

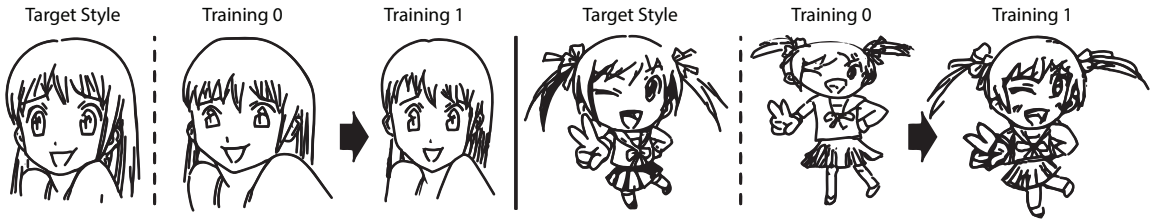


Figure 3.11: Samples sketches drawn by artists at the first and final stages of training along with the target style sketches. Sketches on the left correspond to the intermediate artist while ones on the right to the novice artist. The progress in their sketching styles is noticeable for both artists but more so for the novice one.

3.6.1 SAR for Sketch Style Training

How persistent are artistic styles when artists undergo extensive training for the explicit purpose of altering their style and adopting another? For example, this type of training is administered in major cartoon companies (e.g. Walt Disney). For this purpose, we develop a SAR-based application that allows artists, designers, and animators in-training to quantify their progress in adopting the *target* style of a particular artist, whose sketches have been

encoded in SAR as demonstrated next.

Style Training Setup We first determined a target style by finding a reasonably known artist, who has published his work online along with specific instructions on how to follow his drawing style. The target artist also provided YouTube videos showing how his sketches can be drawn step-by-step. We selected fives sketches from his portfolio, specifically those with video instructions. Then, we identified three artists with varying levels of sketching and artistic experience (novice, intermediate, and advanced) to undergo the style training process. The advanced level artist is a professional with 10 years of experience and the intermediate artist has 3, while the novice artist only sketches as a hobby.

In the first stage of training, we asked all three artists to draw the five target sketches and then used the SAR-based tool to give them quantitative feedback on how similar their sketching style is to the target. The measure based on BoW features defined earlier in Section 3.5.2 is used to compute style similarity. In the second stage, we asked the artists to draw the sketches again after consulting a set of textual instructions on how to draw the target sketches. These instructions were obtained from the target artist's website. In the third stage, we provided the artists with their new SAR similarity scores along with instructional online videos. After submitting their newest sketches, we again computed their SAR similarity scores. Sample sketches of both novice and intermediate level artists at the first and last stages of training are shown in Figure 3.11. It is obvious from these results that the novice artist has improved more significantly than the intermediate artist in adopting the target style.

Style Training Results Figure 3.12 plots the evolution of the SAR similarity score throughout the 3-stage training process for each artist. As expected, each trainee's score increases with training. The novice artist exhibits an overall improvement of 10%, which can be attributed to the explicit use of textual and video instruction for target training. The initial

similarity scores for the intermediate and advanced level artists are much higher than the novice one, but they also exhibit an improvement of 4% and 2% respectively. The slight improvement by the advanced artist conveys how difficult it is for an artist with a well-defined style and extensive experience to adopt a new style, as well as, how much easier it is for a novice artist to do the same. Based on our results, we conclude that SAR can be successfully employed to quantitatively monitor training progress. A training executable is publicly available online. We will also release the training source code to extend this application to a larger set of sketches from more participating artists and to allow users to build their own training applications using their own sketch data.

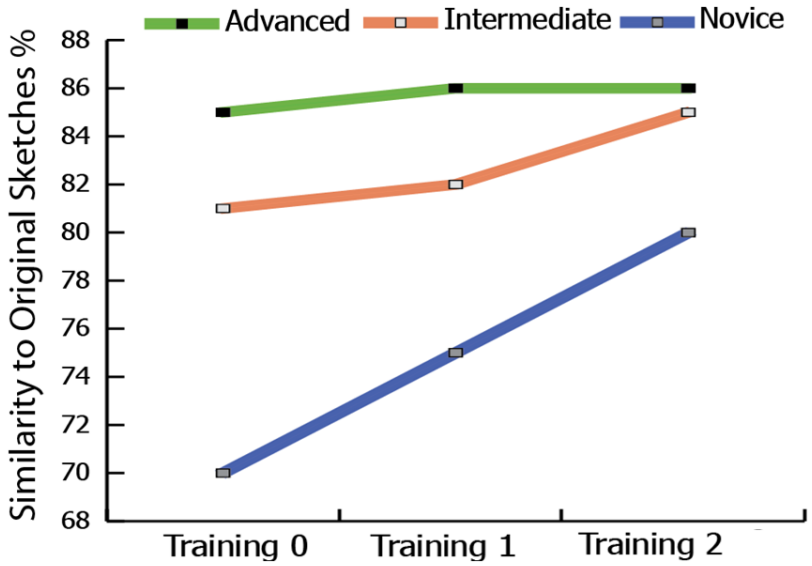


Figure 3.12: SAR similarity-to-target scores for 3 novice, intermediate, and advanced level artists after 3 stages of training. The novice artist has the least score in the initial stage but progresses the most with training.

3.6.2 SAR for Evaluating Sketch Synthesis Results

As mentioned in chapter 2, there is a number of recent methods that focus on sketch synthesis and artistic style analysis. Their aim is to develop automatic sketch synthesis tools that mimic a particular artistic style. Most of these tools do not provide a quantitative assessment of how close the synthesized style is to the target one, thus, making it difficult to

compare synthesis methods. A few of them have validated their synthesis quality through extensive user studies, which are cumbersome to compile and require careful analysis. We propose using SAR to quantitatively evaluate sketch synthesis methods without the need to conduct tedious user studies. To exemplify this application, we use SAR to analyze the synthesis results of [4]. In their work, portrait sketches are synthesized from artistic styles of 7 artists, each of which drew 24 portrait sketches. The real and synthesized sketches are publicly available.

We first use SAR to classify artistic style among the real portrait sketches only. In this case, SAR accuracy is 52% (leave-one-out) and 50% (2-fold), as compared to random chance accuracy of 14%. When classifying only synthesized sketches, we obtain 54% accuracy (leave-one out) and 51% accuracy (2-fold) respectively. The similarity in performance between the two scenarios suggests that the synthesized portraits are as hard to distinguish as the real ones and that the synthesis tool of [4] maintains a very similar amount of style diversity as in the real sketches. Next, we use SAR to classify the real sketches using the synthesized ones only as training and vice versa. In this case, SAR accuracy dropped to 26% (leave-one-out) and 25% (2-fold). This performance *drop* can be used as a quantitative measure of synthesis quality, as it allows for comparison between different synthesis methods. In fact, an ideal synthesis tool should produce a SAR accuracy drop close to zero, since the style of the synthesized sketches should be indistinguishable from that of the real sketches.

Our results are on par with those of the three user (perceptual) studies in [4]. In fact, we setup the SAR classification experiments above to follow the same setup used in these user studies, where the only difference between the two is that SAR performs the classification automatically without any human feedback. The participants in the user studies registered a very similar accuracy when classifying the real and synthesized portraits separately (as did SAR). More importantly, the drop in classification by human participants is around 20%, which is comparable to the 25% drop reported by SAR. This result validates that SAR

can be used to automatically and quantitatively evaluate sketch synthesis tools, without the need to conduct cumbersome user studies.

3.7 Conclusion

In this par of the dissertation, we shed light on a new direction in sketch analysis, namely authorship recognition through stroke analysis. We propose a stroke authorship recognition (SAR) approach that discriminates between artistic sketch styles based on the choice and frequency of use of basic strokes. SAR is motivated by real-world issues, including the fact that fraudulent copies of original sketches with patent and copyright protection are sold on some websites, which claim them to be original. As it is hard for an individual to discriminate authentic and fraudulent sketches (Section 3.4), original sketches from known companies and artists are in danger of replication and copyright infringement. Moreover, cartoon artists undergo months of training before they can officially contribute, so as to ensure that the style in which the characters are drawn is not distorted when artists are replaced (Section 3.6.1). Similar to handwriting and signatures, sketches can be used for user authentication as discussed in chapter 2. Thus, detecting major and minor differences in sketching style is important. Furthermore, design patents that contain sketches of commercial designs need to be protected, as is apparent in the ongoing lawsuits between Apple and Samsung for example. All of this shows the need to have automatic tools, such as SAR, to assess the authenticity of sketched objects and identify their authorship.

From our extensive experiments and user studies (Section 3.5.1), we provide empirical evidence that SAR *does* encode unique and consistent characteristics of an artist's sketching style, which are in turn used to discriminate one artist's sketches from others. We also showed that the extent to which SAR can be used for discrimination is dependent on the sketching constraints imposed on the artist. Moreover, our extensive user studies empirically validate the difficulty of this fine-grained recognition problem and indicate that SAR can improve upon human performance. All the compiled data (datasets and user studies)

and source code will be made publicly available to enable further research on this exciting topic and to allow for quantitative comparison.

Chapter 4

Sketch Style Transfer

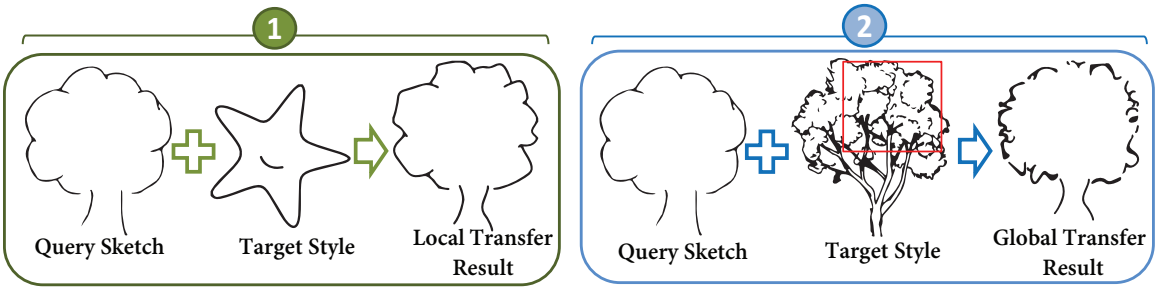


Figure 4.1: An overview of the proposed style transfer pipeline. The first module (*stroke style transfer*) deals with transforming a line-drawing at the stroke-level so its style is similar to that of a stroke transfer line-drawing. The second module (*brush style transfer*) transforms a line-drawing into a stylized sketch given a sketch with a unique style.

4.1 Introduction

In this chapter, we propose an efficient technique for automatic sketch style transfer, which transforms a sketch (a query sketch) in such a way that it represents another artistic sketching style (of a target sketch), regardless of the sketch content. We propose two methods of transfer as shown in Figure 4.1. The first method focuses on *stroke-style transfer*, by transforming the query sketch at the stroke level. It is built on concepts and insights uncovered by the discriminative SAR model, proposed in chapter 3. In chapter 3, we demonstrate that the subtleties of an artist's style can be recognized by analyzing the frequency of some representative geometric stroke-level features. In stroke-style transfer, we exploit this local stroke representation by first modeling stroke segments as rational (parametric) quadratic Bézier curves for both query and target line drawings. Based on the parametrization of the

Bézier curves, the distribution of geometric features (e.g. stroke eccentricity) in the query sketch can be modeled and its SAR distance to the target sketch style can be minimized. In doing so, all strokes in the query are modified to closer resemble the target's stroke-level style.

The second transfer method (i.e. *brush-style transfer*) focuses on transferring the brush look of a stylized sketch (a target sketch with a unique brush look) to a line drawing (the query sketch). We find that convolutional sparse coding (CSC) can model/approximate the sketching process of artists and, therefore, it is used for brush-style transfer stage. This assumption is empirically validated in our work through a user experiment. Interestingly, we cast brush-style transfer as a mapping problem between stylized brush elements and their corresponding abstractions using simple strokes. CSC is used here to synthesize a simple line-drawing that depicts the content of the target but using simple line strokes.

Our technique can be used to apply both subtle and major transformations on hand-drawn sketches. For subtle transformations, artists can use the stroke-style transfer module to get their sketches closer to a target style, which they are trying to mimic. For example, this is essential when a new artist is assigned to draw a well-known cartoon character by following the same style as another artist. Local stroke-level features are hard to mimic in this case, but they are crucial to maintaining the overall sketch characteristics of the cartoon character. On the other hand, major sketch transformations are made possible by our brush-style transfer module. With it, artists are able to automatically apply a unique brush look of a target sketch to their simple line drawings. This automatic transfer saves artists a painstakingly large amount of effort, when compared to creating this transferred brush look manually. The word *style* is interpreted in so many different ways in different contexts. Thus, each definition, as we show in chapter 2, tackles the area of sketch synthesis and transfer differently leading to different results and targeting and serving different applications. Our proposed methodology adds new perspective to the problem and defines sketching style to include both stroke-level and brush-level aspects of a sketch. With both

methods of style transfer, we aim to present a comprehensive technique for sketch transfer, which is feasible in various artistic sketch applications.

4.2 Stroke Style Transfer

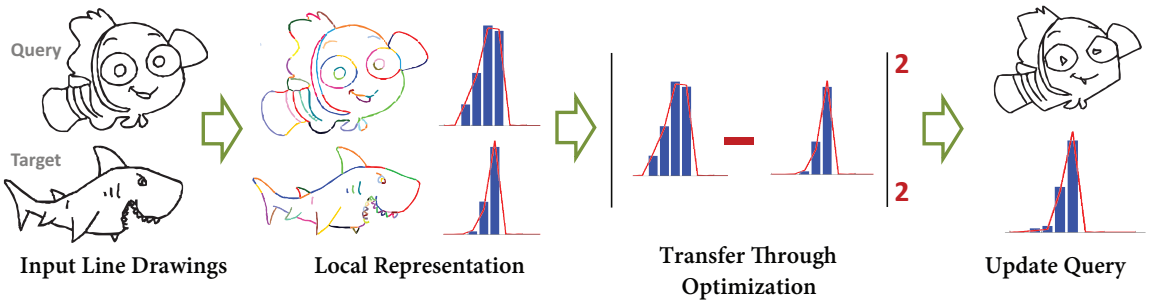


Figure 4.2: The work flow towards stroke-style transfer: (1) given a query and a target line drawings ;(2) apply techniques for stroke extraction and segmentation and characterize each stroke segment mathematically upon which each sketch is represented as a distribution of feature frequency; (3) solve an optimization to approximate the query distribution to the target's; (4) update the query sketch accordingly.

In this section, we give a detailed description of the first sketch style transfer method stroke-style transfer. It focuses on transferring the style of a sketch at the stroke level. Our method is built on the concepts discussed in chapter 3. In chapter 3, we show that the frequency in which an artist uses a set of basic strokes is unique to that artist and can be used for stroke authorship recognition (SAR). This work developed a SAR distance (or loss) that can be used to measure the difference in style between two line drawings. We exploit this distance and extend its use to sketch style transfer.

Given the query line drawing $\mathbf{I}_q$ and a target stroke-style line drawing $\mathbf{I}_t$, we aim to transform the stroke segments of $\mathbf{I}_q$ such that the SAR distance between the transformed sketch $\mathbf{I}_S$ and that of $\mathbf{I}_t$ is minimized, as demonstrated in Figure 4.2.

4.2.1 Stroke-Based Sketch Representation

To formulate the SAR distance, drawings $\mathbf{I}_q$ and $\mathbf{I}_t$ are decomposed into strokes, which are further split into smaller segments, small enough to be modeled by conic sections. Each

stroke segment is parameterized as a quadratic rational Bézier curve. Strokes are grouped together based on their eccentricity values to form a dictionary of stroke segments. Based on that, each sketch is represented as a histogram of stroke segments using a hierarchical bag-of-words (BoW) model, which encodes some aspects of the artist's unique style. Further elaboration is provided in what follows.

Stroke Extraction and Segmentation. We use the vectorized version of the free style sketch dataset provided in chapter 3. The free-style sketches, composed of 70 sketches by 10 artists, are vectorized using Adobe Live Trace. It is an of-the-shelf digitization technique, which generates sketches that stay faithful to the original sketch [5]. Strokes extracted from the vectorized sketches are quite long and contain a lot of geometric information. Thus, they need to be further split into smaller segments. We fit a b-spline curve to each stroke and locate break points at local maximum curvature locations in the b-spline. To avoid over-segmentation, linear strokes are handled by placing break points at its two ends. Each stroke segment is identified by the pixels between two consecutive break points.

Mathematical Characterization. After $\mathbf{I}_q$ and $\mathbf{I}_t$ are decomposed into stroke segments, each segment is parameterized using a quadratic rational Bézier curve model, as in Eq (4.1).

$$f(t) = \frac{\sum_{i=0}^{2} w_i \mathbf{p}_i B_i^2(t)}{\sum_{i=0}^{2} w_i B_i^2(t)}, \quad t \in [0,1]\,, \tag{4.1}$$

where $\mathbf{p}_i$ is the i^{th} control point of the 2D curve, w_i is the corresponding weight, and $B_i^2(t)$ is the quadratic Bernstein polynomial (i.e. $(1-t)^2,\ 2t(1-t),\ t^2$). Following the standard form in [87], we set $w_0 = 1, w_1 = w > 0$ and $w_2 = 1$. Bézier curves have been widely used in computer graphics to model smooth curves [87, 88]. By manipulating the control points of a Bézier curve, it can be intuitively deformed. Quadratic Bézier curves are enough to model conic sections and only three control points $\{\mathbf{p}_i\}_{i=0}^2$ are needed.

A fundamental parameter of a conic section is its eccentricity, i.e. the deviation of

that conic from a circle. Thus, eccentricity can be used not only to distinguish between different types of conics (as done in chapter 3), but also to provide information of their overall shape. This feature is also invariant to rigid body deformations. In the previous chapter, eccentricity is mainly used as a local feature to mathematically characterize stroke segments for sketch authorship discrimination. All of these stated motives encourage us to choose eccentricity as the local feature to describe a stroke segment. The eccentricity e of a rational quadratic Bézier curve is computed as in Eq (4.2).

$$e(\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2)^2 = \frac{2\sqrt{\triangle}}{\sqrt{\triangle} - b}, \tag{4.2}$$
$$\text{where } \triangle = \left[k\|\mathbf{s} - \mathbf{t}\|^2 - 2(\|\mathbf{s}\|^2 + \|\mathbf{t}\|^2)\right]^2 + 4(k-1)(\|\mathbf{s}\|^2 - \|\mathbf{t}\|^2)^2.$$

To provide some control over the representation as in [89], we set $k = \frac{1}{w^2}$, $\mathbf{s} = \mathbf{p}_0 - \mathbf{p}_1$, $\mathbf{t} = \mathbf{p}_2 - \mathbf{p}_1$ and $b = -k\|\mathbf{s} - \mathbf{t}\|^2 - 4(\mathbf{s} \cdot \mathbf{t})$. For derivation, proof and further discussion on eccentricity of rational Bézier curves refer to [89].

Eccentricity Distribution. Given each stroke segment s_i characterized by an eccentricity value e_i, where $i \in \{1, ..., M\}$ and M is the number of stroke segments in $\mathbf{I}_q$, we now represent this sketch as a distribution of eccentricity values. Following the framework in the previous chapter (chapter 3), we build a stroke segment dictionary of K elements using hierarchical k-means clustering on the stroke segments of the free style dataset. We choose $K = 20$ for our experiments. This same stroke segment dictionary is used in the brush-style transfer module in Section 4.3. Using the K elements of this dictionary as high dimensional quantized bins and following the traditional BoW technique [81], we can represent a sketch as a histogram of these stroke segments using their eccentricity information. In other words, a sketch (either query $\mathbf{I}_q$ or target $\mathbf{I}_t$) can be represented using a K dimensional histogram, which quantifies the frequency in which each stroke segment is used in the sketch.

To modify the strokes in $\mathbf{I}_q$, we need to parameterize its BoW histogram, so as to generate a smooth functional form that can be varied to minimize a particular objective. We use kernel density estimation (KDE) for this purpose. We estimate the probability of each bin ($\mathbf{b}_k \; \forall k \in 1, \cdots, K$), in the histogram using Gaussian kernel density estimation as shown below:

$$f(\mathbf{b}_k) = \frac{1}{M\sqrt{2\pi h}} \sum_{i=1}^{M} \exp\left(\frac{-[e_i(\mathbf{p}_0^i, \mathbf{p}_1^i, \mathbf{p}_2^i) - \mathbf{b}_k]^2}{2h^2}\right),$$

where h is the bandwidth value ($h = 0.1$ for our experiments), e_i is the eccentricity in Eq (4.2) of the i^{th} stroke segment.By concatenating the probabilities of all K bins, we obtain the desired sketch representation histogram $\mathbf{h}_q$ for $\mathbf{I}_q$. Note that this histogram is parameterized by the control points of all M stroke segments $\{\mathbf{p}_0^i, \mathbf{p}_1^i, \mathbf{p}_2^i\}_{i=1}^M$.

$$\mathbf{h}_q(\{\mathbf{p}_0^i, \mathbf{p}_1^i, \mathbf{p}_2^i\}_{i=1}^M) = [f(\mathbf{b}_1) \; \cdots \; f(\mathbf{b}_k)]^\top \tag{4.3}$$

4.2.2 Transferring Stroke-Style from $\mathbf{I}_t$ to $\mathbf{I}_q$

In this section, we show that stroke-style transfer can be realized by solving a properly formulated optimization problem. It is based on the assumption that sketches drawn by the same artist have more similar representative feature distributions (KDEs) than sketches drawn by different artists, while remaining independent of sketch content. This assumption is validated in [11]. We aim to transfer the style of $\mathbf{I}_t$ (represented by $\mathbf{h}_t$) to that of $\mathbf{I}_q$ (represented by $\mathbf{h}_q$). This can be done by minimizing the objective in Eq (4.4).

$$\{\triangle\mathbf{x}_i^*\}_i^M = \underset{\{\triangle\mathbf{x}_i\}_i^M}{\operatorname{argmin}} \; \left\|\mathbf{h}_q\left(\{\triangle\mathbf{x}_i\}_i^M\right) - \mathbf{h}_t\right\|_2^2 + \lambda \sum_{i=1}^{M} \|\triangle\mathbf{x_i}\|_2^2, \tag{4.4}$$

where $\mathbf{h}_q$ is the KDE of eccentricities $(e_i, ...e_M)$ from sketch $\mathbf{I}_q$, and $\mathbf{h}_t$ is the same KDE for sketch $\mathbf{I}_t$. The set of optimization variables in this optimization is $\{\triangle\mathbf{x}_i\}_i^M$, where $\triangle\mathbf{x}_i = [\triangle{\mathbf{x}_i}^0; \; \triangle{\mathbf{x}_i}^1; \; \triangle{\mathbf{x}_i}^2] \in \mathbf{R}^6$ is the concatenation of the three offset vectors for the

2D control points $\mathbf{p}_0^i$, $\mathbf{p}_1^i$ and $\mathbf{p}_2^i$ of the i^{th} segment s_i of $\mathbf{I}_q$. In other words, $\mathbf{I}_q$ will be transformed into $\mathbf{I}_S$ by transforming each segment s_i in $\mathbf{I}_q$ into another segment $\hat{s}_i$ with the following three control points: $\hat{\mathbf{p}}_j^i = \mathbf{p}_j^i + \triangle {\mathbf{x}_i}^{*j}$ for $j \in \{0, 1, 2\}$. As in many inverse problems, we breakdown the cost function into two complementary parts: a data term and a regularizer. The data term $\left\| \mathbf{h}_q \left(\{\triangle \mathbf{x}_i\}_i^M \right) - \mathbf{h}_t \right\|_2^2$ encourages that the SAR distance between the query and target eccentricity histograms be minimum, i.e. the query style gets closer to the target style. The regularizer term $\sum_{i=1}^{M} \|\triangle \mathbf{x_i}\|_2^2$ penalizes a degenerate transfer that simply copies the target. In our work, we set the tradeoff coefficient $\lambda = 0.5$.

Implementation Details. Each stroke segment curve passes through $\mathbf{p}_0$ and $\mathbf{p}_2$, the initial and last points of the Bézier curve. To preserve connectivity in our transferred sketch, we enforce ${\triangle \mathbf{x}_i}^0 = \triangle \mathbf{x}_{i-1}^2$ (or equivalently, $\hat{\mathbf{p}}_0^i = \hat{\mathbf{p}}_2^{i-1}$) in the optimization, whenever the $(i-1)^{\text{th}}$ and i^{th} curves are connected in $\mathbf{I}_q$. In addition to connectivity, we should maintain similar coherence between consecutive stroke segments in the transformed sketch as in the query sketch (refer to Figure 4.3 for an example). In other words, tangent values between connected curves should be preserved in $\mathbf{I}_S$. Since we are only interested in finding the tangent at the beginning and end of the curve segment (at $t \in \{0, 1\}$), we produce an analytical form of the tangent at both ends of each curve segment and force tangent values at the end of one segment and the beginning of the connected segment to be equal. We add this equality to the objective function of our optimization.

We minimize the cost function in Eq (4.4) by using a Quasi-Newton method with a cubic line search procedure for updating the approximation of the Hessian matrix [90], which is implemented in the *fminunc* function of MATLAB. To speedup the process and since the cost is smooth, we can generate the analytical form of its gradient w.r.t. the variables $\{\triangle \mathbf{x}_i\}_i^M$. Since the cost is non-convex, the proposed optimization method can only guarantee a local minimum. However, our experiments show that this local solution is quite reasonable and results in visually appealing style transfer.

After obtaining the $\{\triangle \mathbf{x}_i^*\}_i^M$ solution from the optimization, we can generate the control points of the transformed sketch $\mathbf{I}_S$. Knowing these control points is enough to generate (and sample) the underlying Bézier curves using the de Casteljau algorithm [91], thus, rendering the transferred sketch.

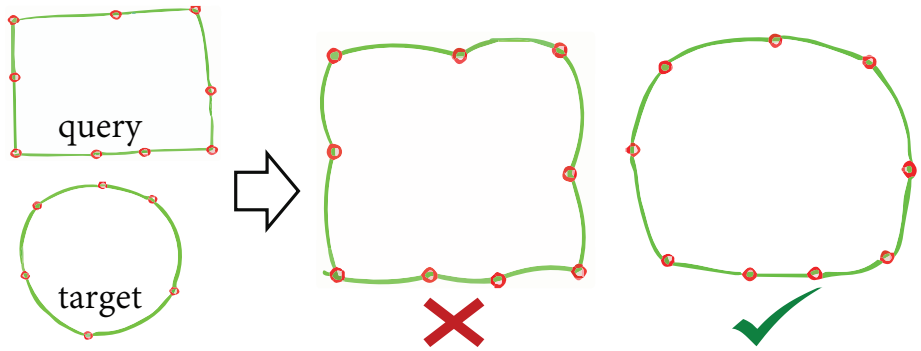


Figure 4.3: A comparison of transfer results when a rectangle sketch is used as the query and a circle as the target. The synthesis result on the left uses our optimization without maintaining local coherence in tangent, which leads to poor non-smooth (cloudy like) transfer. The transfer on the right, however, enforces this coherence leading to a more visually appealing result.

4.3 Brush Style Transfer

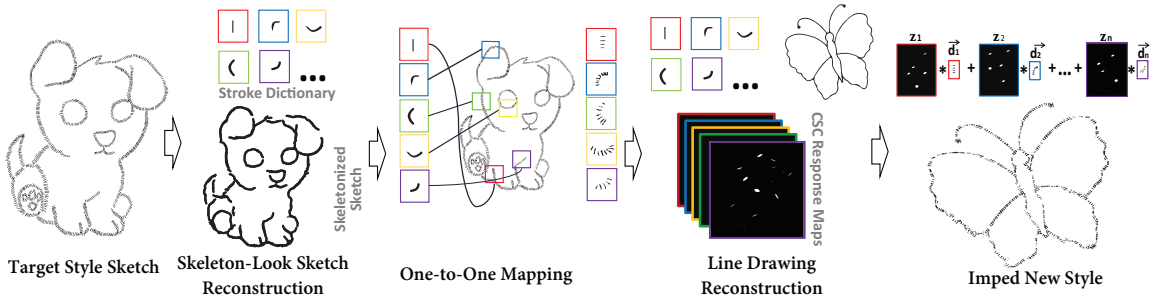


Figure 4.4: Brush-style transfer module: (1) given a target stylized sketch ;(2) construct a skeleton-look of the target sketch (enforcing sparsity) using our stroke segment dictionary; (3) match brush strokes from the target image with each of the original stroke segment dictionary; (4) Obtain the sparse codes of a line drawing by solving an optimization problem and using our stroke dictionary; (5) replace each stroke in the dictionary with its matching brush stroke and convolve with the original sparse codes to obtain a transferred sketch.

Up to this stage, we have applied style transfer at the stroke level to produce subtle but distinctive transformations in $\mathbf{I}_S$. The second transfer method in this chapter, applys style transfer at the brush level, where the brush-look of the entire sketch is transferred, leading to significant sketch transformation ($\mathbf{I}_B$). We asked a number of artists to draw target sketches characterized by a unique brush look and used their *stylized* sketches as a target $\mathbf{I}_t$ for our brush transfer. For this type of transfer, we need to find a mapping between a stroke segment (from a line-drawing) and its corresponding brush element. To do this automatically (without manual interaction), we generate a skeletonized version of the target sketch $\mathbf{I}_t$ by applying convolutional sparse coding (CSC) and the stroke segment dictionary defined earlier in Section 4.2.1. Since it can be argued that CSC approximates how sketches are drawn, overlaying this skeleton on top of $\mathbf{I}_q$ (query sketch) finds one-to-one mappings between brush image patches in $\mathbf{I}_t$ and corresponding stroke segments used to reconstruct this patch in its skeletonized version. As such, brush-style transfer becomes equivalent to mapping the stroke segment dictionary used to represent sketches in the stroke-style transfer to a corresponding dictionary of brush element patches used in $\mathbf{I}_t$. The overall brush-style transfer procedure is depicted in Figure 4.4. In what follows, we introduce the

CSC model in the context of brush-style transfer. Since it is not commonly used in this area, we validate our choice of CSC intuitively and empirically through a user experiment. Then, we elaborate on each stage of the brush-style transfer module.

4.3.1 Sketch Reconstruction using CSC

CSC represents an image by a sum of convolutional responses generated from image filters and sparse maps. These filters constitute a CSC dictionary $\mathbf{D} = \{\mathbf{d}_1, ..., \mathbf{d}_K\}$, and when convolved with their corresponding sparse maps $\mathbf{Z} = \{\mathbf{z}_1, ..., \mathbf{z}_K\}$, the image is reconstructed. CSC has become an important tool in several computer vision applications, including classification, localization, tracking, denoising, deblurring and inpaiting [92]. Given the CSC dictionary, inferring the sparse maps for an input image $\mathbf{x}$ is done by solving the general CSC optimization in Eq (4.5). The data term enforces the CSC image representation model and an ℓ_1 term is added to enforce sparsity on the elements in $\mathbf{Z}$.

$$\arg\min_{\mathbf{z}} \quad \frac{1}{2}\left\|\mathbf{x} - \sum_{k=1}^{K} \mathbf{d}_k * \mathbf{z}_k\right\|_2^2 + \beta \sum_{k=1}^{K} \|\mathbf{z}_k\|_1 . \tag{4.5}$$

This model is closely related to sketch drawing. The i^{th} map $\mathbf{z}_i$ can be viewed as a sparse score map, which scores each pixel based on whether the filter or stroke $\mathbf{d}_i$ is localized there or not. A pixel location with a high score means that $\mathbf{d}_i$ contributes significantly at this pixel to form the corresponding patch in the original image $\mathbf{x}$. To solve the optimization in Eq (4.5), we use the conventional ADMM algorithm [93]. We solve the problem efficiently in the Fourier domain, as discussed in detail in [94].

4.3.2 Validating CSC for Sketch Drawing

In our work, we utilize CSC in implementing the brush-style transfer stage motivated by the way image reconstruction takes place in such a model. As discussed above, CSC reconstructs an image using a summation over a convolution between response maps $\mathbf{Z}$ spec-

ifying *where* in the image a response takes place and a set of filters $\mathbf{D}$, which reflect *what* the response (stroke) will look like. We believe that this model approximates how an artist would draw a sketch, i.e. making the decision where to draw a stroke and what the stroke will look like (from a dictionary of strokes), except that it is a union operation rather than a pixel-level summation as in CSC. To validate our assumption, we use CSC to reconstruct line-drawings, where the filters are taken to be the elements of our stroke dictionary (20 strokes) introduced in Section 4.2.1. Figure 4.5 shows the evolution of the reconstruction for a sketch drawing at different iterations of the ADMM optimization. The sketch is reconstructed with high visual clarity, thus, justifying CSC and the stroke dictionary we use. We also validate the choice of CSC in brush-style transfer through an empirical case study, which estimates how many times an artist (on average) passes through a certain square region while drawing a stylized sketch. We demonstrate details of the experiment below.

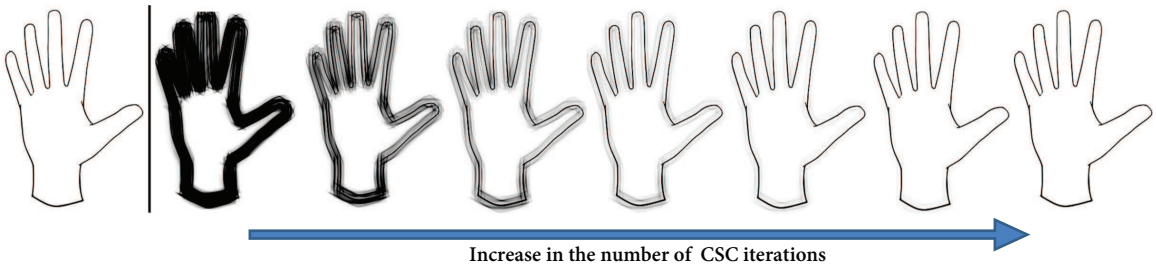


Figure 4.5: Evolution of CSC reconstruction across various ADMM iterations.

Experiment Setup. Using a Wacom and a stylus pen, three artists are asked to freely draw a given sketch. We parse the Wacom data files containing the sketches and divide the sketches into square areas of 60×60, 30×30, and 10×10 pixels, where the whole image is of 300×300 pixel resolution. For each square region, we count the average number of times an artist passes his/her pen through this area. This is done using the spatial and temporal information generated from the Wacom device. We eliminate areas that contain no strokes.

Experiment Results. Results of this experiment are shown in Table 4.1.According to this experiment, the sparsity of strokes used in patches of a sketch reaches a reasonable amount when using patches of 30×30 pixels, which is the size we adopt for the CSC dictionary

elements in our work. Using larger sizes (i.e. 60×60) results in less sparsity, since artists tend to use overlapping strokes that cover larger areas. Using smaller sizes (i.e. 10×10), on the other hand, carries trivial details of a stylized sketch that is not enough to represent a brush look. We conclude that artists need only pass their stylus pens through the same area of a stylized sketch a sparse number of times. This observation validates the concept of using response maps showing where strokes (filters) would be sparsely placed in a sketch, thus, justifying the overall use of CSC.

Table 4.1: The frequency in which an artist needs to pass his/her stylus pen through a 60×60, 30×30 and 10×10 pixel region while drawing a stylized sketch of resolution 300×300 pixels. Refer to the text for details.

	1-pass	2-passes	3-passes	4-passes
60×60 pixels	16%	40%	32%	12%
30×30 pixels	60%	30%	10%	0%
10×10 pixels	80%	18%	2%	0%

4.3.3 Brush-Style Transfer from $\mathbf{I}_t$ to $\mathbf{I}_q$ using CSC

Since CSC can be viewed as a way to model the sketching process, we can reformulate the brush-style transfer problem from $\mathbf{I}_t$ to $\mathbf{I}_q$ as a matching problem between the stroke segments used to generate $\mathbf{I}_q$ and their corresponding brush elements. If this mapping is known, then the transferred sketch can be generated using the CSC formation model, where the sparse response maps $\mathbf{Z}$ are generated by encoding $\mathbf{I}_q$ using the stroke segment dictionary $\mathbf{D}$ (i.e. the K strokes found using kmeans in Section 4.2.1) and the filters are replaced by the equivalent brush stroke dictionary. In this case, the placement of the stroke is the same but the strokes themselves are replaced. Next, we focus on how this mapping can be done automatically.

Skeletonized Sketch Generation from $\mathbf{I}_t$. The one-to-one mapping between brush elements and stroke segments could be easily inferred, if the same content of the target sketch $\mathbf{I}_t$ is drawn again but using the simple strokes in $\mathbf{D}$. In other words, the artist of $\mathbf{I}_t$ would be asked to redraw his sketch but only using a limited number of strokes, primarily one stroke

per location in the canvas. In doing so, we overlay both sketches and generate a one-to-one mapping between each brush element (of a particular patch size) and the simple stroke that the artist used at the same location. Obviously, generating such a *skeletonized* sketch of $\mathbf{I}_t$ requires significant manual interaction by the artist, as well as, pixelwise accurate registration for a correct overlay. To avoid this tedious step, we generate the skeletonized sketch *automatically*.

Given the target $\mathbf{I}_t$, we need to output a skeletonized sketch $\mathbf{I}_{skeleton}$ (refer to the second stage of Figure 4.4 for an example). In fact, $\mathbf{I}_{skeleton}$ approximates what $\mathbf{I}_t$ would look like if only simple strokes are used to draw it and the elaborate brush style is suppressed. To produce $\mathbf{I}_{skeleton}$ automatically, we encode $\mathbf{I}_t$ using the CSC model in Eq (4.5), where the CSC dictionary $\mathbf{D}$ is taken to be the stroke segment dictionary in Section 4.2.1. Since the tradeoff parameter β can significantly impact the quality of the reconstruction (e.g. higher values result in sparser maps and more abstract reconstructions, while smaller values give more exact reconstructions), we use a high value of β to skeletonize $\mathbf{I}_t$.

The skeletonized sketch $\mathbf{I}_{skeleton}$ is produced according to Eq (4.6), where $\mathbf{d}_k^s$ is the k^{th} stroke segment in $\mathbf{D}$ and $\mathbf{z}_k^s$ is its corresponding sparse response map. Figure 4.6 shows the an example skeletonized sketch reconstruction for $\beta = 5$.

$$\mathbf{I}_{skeleton} = \sum_{k=1}^{K} \mathbf{d}_k^s * \mathbf{z}_k^s \ . \tag{4.6}$$

Brush-Style Matching. Now that $\mathbf{I}_{skeleton}$ has been constructed, what remains is to match brush patches from the target $\mathbf{I}_t$ with stroke segments in $\mathbf{I}_{skeleton}$. To do this, we use the sparse maps inferred from the CSC coding of $\mathbf{I}_{skeleton}$. For each sparse map $\mathbf{z}_k^s$, we find the pixel location $\mathbf{c}_k$ with the highest score. This location corresponds to the region in the skeletonized sketch, where the corresponding stroke $\mathbf{d}_k^s$ is most prevalent. Then, this stroke is matched with the brush stroke patch $\mathbf{d}_k^b$ from $\mathbf{I}_t$ localized at $\mathbf{c}_k$, as per Eq. (4.7).

$$\mathbf{d}_k^b = \Pi(\mathbf{I}_b, \mathbf{c}_k), \tag{4.7}$$

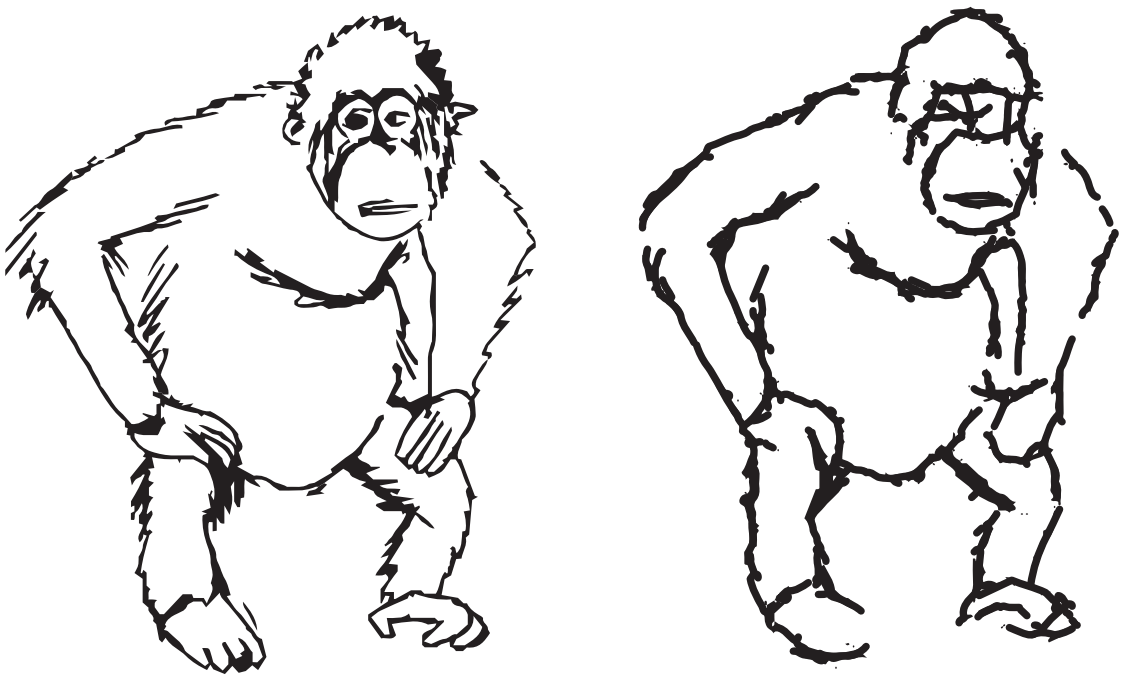


Figure 4.6: Skeletonized reconstruction of the left image using CSC and the stroke segment dictionary.

where $\Pi(\mathbf{I}, \mathbf{c})$ represents a local patch from image $\mathbf{I}$ around $\mathbf{c}$ with size equal to the size of a stroke dictionary element. This step is depicted in the middle part of Figure 4.4. It is worth mentioning here that we can delete some parts of the matched patch, if it has edges or traces from other parts of the image especially at corners. This sometimes occurs when the size of the stroke element is relatively large.

Stylizing $\mathbf{I}_q$ with Brush-Style from $\mathbf{I}_t$. Finally, to generate the final synthetic sketch $\mathbf{I}_B$, we use the CSC formation model, where the response maps $\{\mathbf{z}_k\}_{k=1}^K$ are adopted from encoding $\mathbf{I}_q$ using the stroke dictionary $\{\mathbf{d}_k^s\}_{k=1}^K$, while the filters used for convolution are the mapped brush elements $\{\mathbf{d}_k^b\}_{k=1}^K$ inferred from the skeletonized sketch $\mathbf{I}_{skeleton}$. Eq. (4.8) summarizes this formation. This step is depicted in the rightmost part of Figure 4.4 for a line-drawing from the free style sketch dataset of [11].

$$\mathbf{I}_q \approx \sum_{k=1}^{K} \mathbf{d}_k^s * \mathbf{z}_k \quad \xrightarrow{brush\ style} \quad \mathbf{I_B} = \sum_{k=1}^{K} \mathbf{d}_k^b * \mathbf{z}_k. \tag{4.8}$$

4.4 Results and Validation

In this section, we provide an assessment and discussion of the visual results obtained using our sketch style-transfer approach along with a validation user study. As in any synthesis and transfer problem, a successful algorithm should meet the following three requirements [95]:

- *Style requirement*: transferred sketches should be close to the hand-drawn sketches drawn by artists.
- *Identity requirement*: transferred sketches should not influence the identity of the object, i.e. content independent style transfer.
- *Quality requirement*: transferred sketches should have good visual quality, i.e. limited over-smoothing and noise.

Motivated by these requirements, we first discuss the results of the stroke-style transfer method and the then brush-style transfer method to assess the synthetic transferred results for both methods.

4.4.1 Stroke-Style Transfer Results

We test our algorithm on different sketches from the free-style sketch data set compiled in chapter 3. We build our stroke-style transfer method based on their findings (namely the SAR distance between sketches), which emphasize that an artistic sketching style is sufficiently determined by the frequency of a collection of artist's stroke that he/she uses while sketching. We take our previous findings one step further and experiment if such uniqueness can be used in stroke transfer. As shown in Figure 4.7, transferring a sharp edgy shark sketch using a rounder and a fuller gold fish sketch leads to the replacement of those edgy strokes with rounder ones. It is also obvious that the content is still preserved and that the transferred sketch still represents a shark but a fatter one. Moreover, comparing

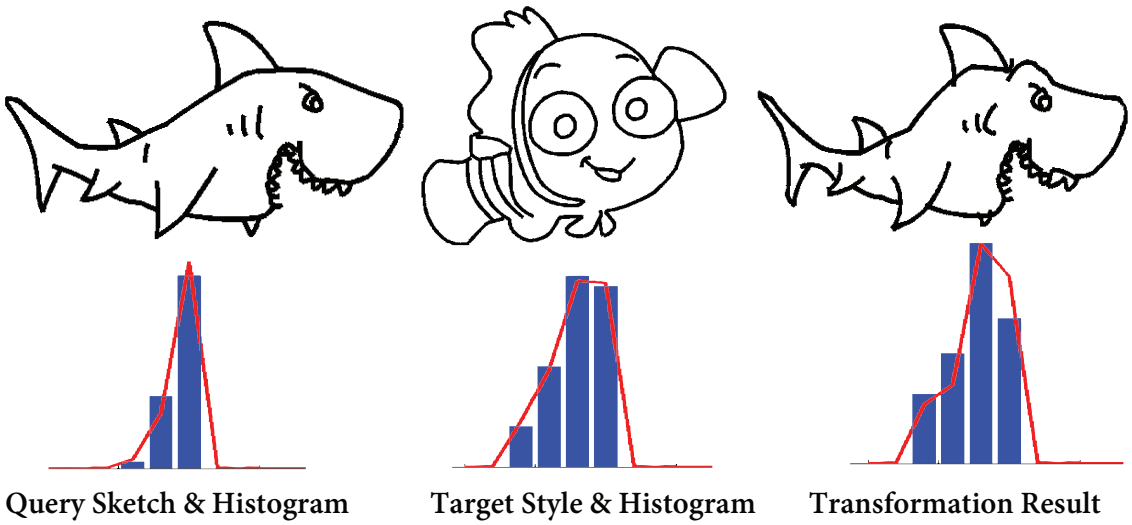


Figure 4.7: An example of stroke style transfer. The transferred shark sketch is rounder and less edgy in comparison to the prior transfer sketch (query). Moreover, the feature frequency distribution of the transferred sketch is closer to the distribution of the target golden fish sketch than it is to the query.

the histograms of the eccentricity feature of the query, target, and transfer result shows that the transferred sketch distribution is closer to the target sketch than it is to the query, which has the same content. This validates our choice of minimizing the SAR distance in our stroke-style transfer optimization in Eq (4.4).

Another example is demonstrated in Figure 4.8, where we transfer the stroke-style of one sketch into multiple styles using different target sketches. We choose our targets to represent the same content (a star), but with varying stroke characteristics (frequencies of eccentricity). These three targets represent different levels of sharpness and roundness. Interestingly, the transfer results reflect the influence of the target sketch. In other words, the sharper the target is, the sharper the transferred results are and vice versa. This is quantitatively evident when we visualize the eccentricity histograms of the transferred sketches.

4.4.2 Brush-Style Transfer Results

As discussed in Section 4.3, we also enable the user to change the brush look of a line-drawing. Given a stylized sketch with a unique brush look, we apply our brush-style trans-

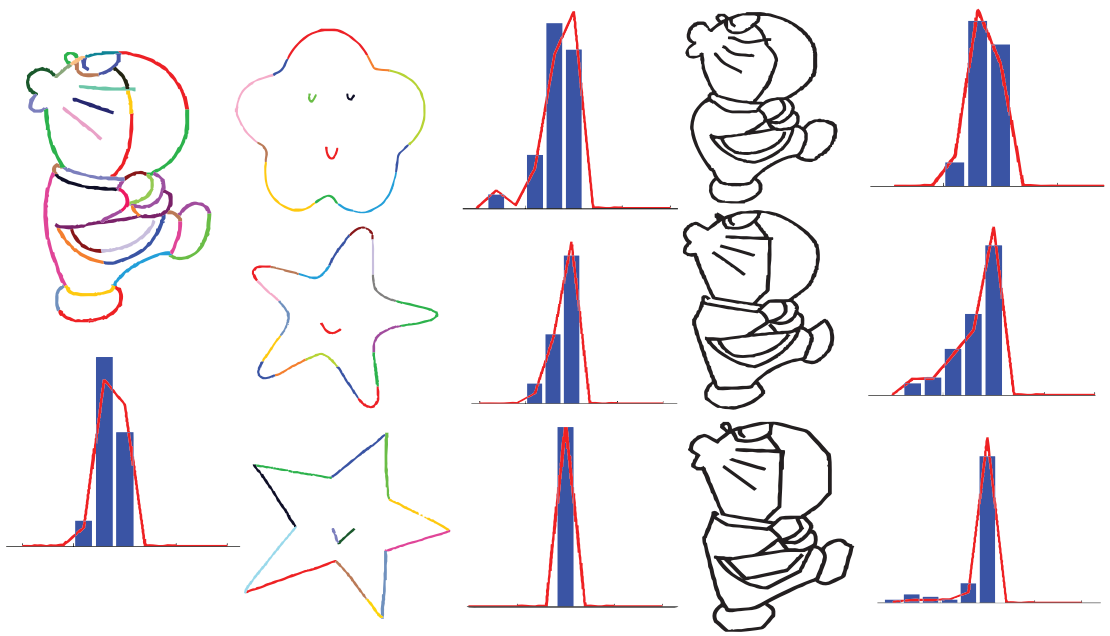


Figure 4.8: An example of stroke-style transfer of a line drawing using various target sketches. Different transferring results are obtained reflecting the roundness and sharpness properties of target sketches.

fer module to produce the tree sketch example in Figure 4.11 and the various sketches in Figure 4.12. Going back to the previously listed requirements of successful transfer, we believe that the *style*, *identity* and *quality* requirements are sufficiently achieved in our proposed transfer algorithm.

4.4.3 Sketch Abstraction via Skeletonization

Sketch abstraction is a form of visual representation that aims to omit extraneous details of a sketched object, while not affecting the perception of the sketch content. It has recently gained attention in the community [37]. In fact, the skeletonization component of our brush-style transfer module can be applied to the field of sketch abstraction as well (refer to Figure 4.9 for an example). Different abstraction levels can be achieved by varying the tradeoff parameter β in Eq (4.6).

Relationshiop with Previous Work. Our sketch style transfer pipeline is fully automated,

so it can be used to transfer any existing sketching style regardless of the sketched content and the method of drawing. Existing sketch style transfer methods (Section **??**), on the other hand, are either content dependent, as is the case of portrait sketches, or semi-automated with various levels of user supervision required. Other methods require sketches to be collected with specific devices, such as a Wacom and stylus pen. They cannot handle scanned versions of a sketch. Moreover, we target sketch transfer at both stroke and brush levels which distinguish our technique in comparison with related transfer methods. Moreover, our technique is designed to transfer sketches to any target style represented in an example stylized sketch, while existing methods tend to target only specific well-defined styles, such as the style of pencil drawing. Since these methods are either content dependent or too specific in their applicability, they cannot be easily extended to the general transfer problem (no constraints on sketch content and target styles) we are focusing on in this paper. Therefore, a fair comparison with these transfer methods is not possible.

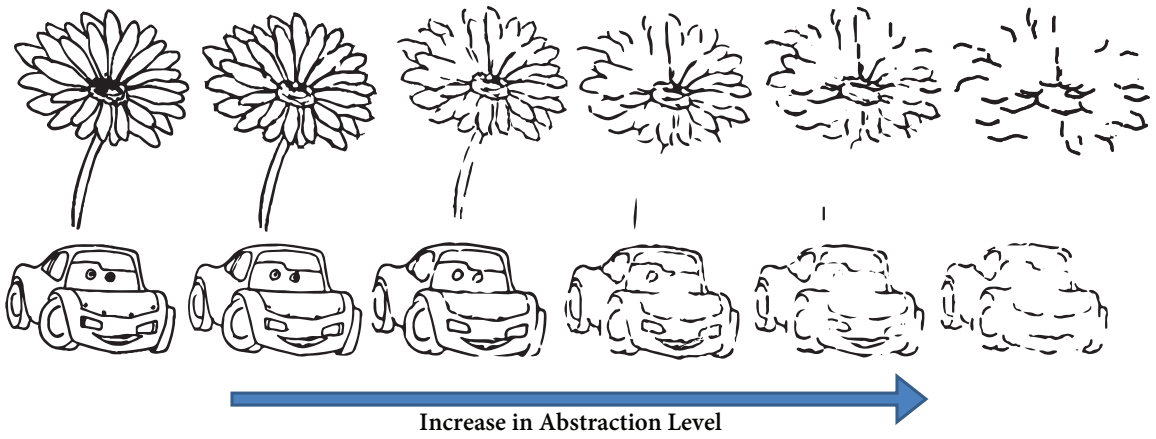


Figure 4.9: Results of applying our brush-style transfer in sketch abstraction. Different levels of abstraction are presented.

4.4.4 Evaluation of Style Transfer: A User Study

We conducted a user study to quantitatively assess our transfer results. In this study, participants are asked to match a transferred version of a line-drawing to the stylized sketch

(i.e. with a unique brush look), which they believe was used for style transfer. Such assessment studies are common in the validation process of sketch transfer and synthesis results as in [21]. A total of 4 questions were asked and each has 3 choices to choose from. To illustrate this study further, an example of one of these questions is given in Figure 4.10. The experiments were conducted on Amazon Mechanical Turk and a total of 500 master workers participated. The recognition rate was 86%, as opposed to 33% random chance.

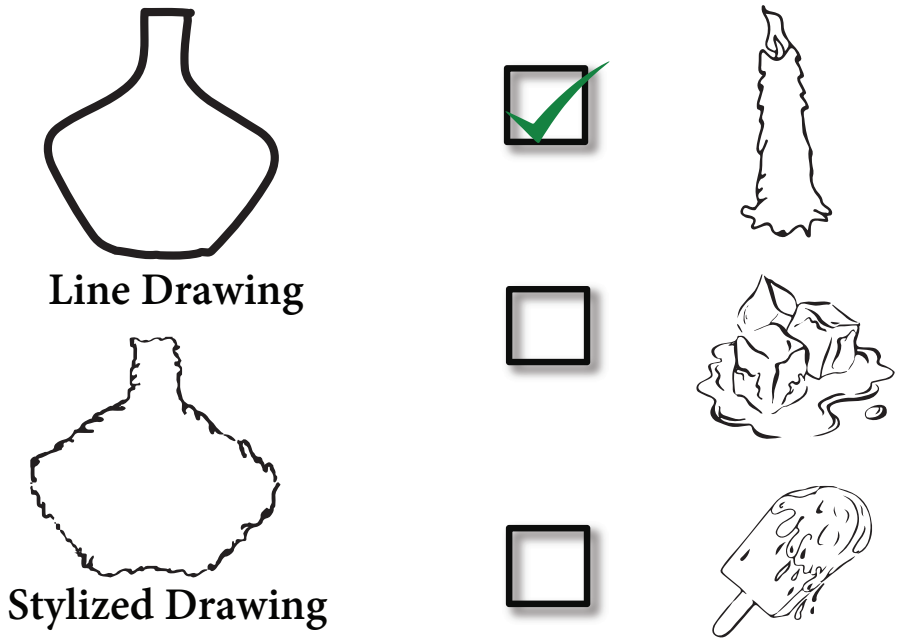


Figure 4.10: An example of one question presented in the validation study. Given a line drawing and its transferred style, participants need to match them to one of the sketches in the options which they think was used in the transfer process.

4.5 Conclusion

In this chapter, we shed light on a new automatic technique for sketch style transfer from one hand drawing to another with different sketching style. It consists of two methods of transfer: stroke-style and brush-style transfer. Stroke-style transfer targets the transfer of sketching at the stroke level through modifying the parameters of stroke segments of a sketch. To do so, a global sketch distance between both query and target sketches is min-

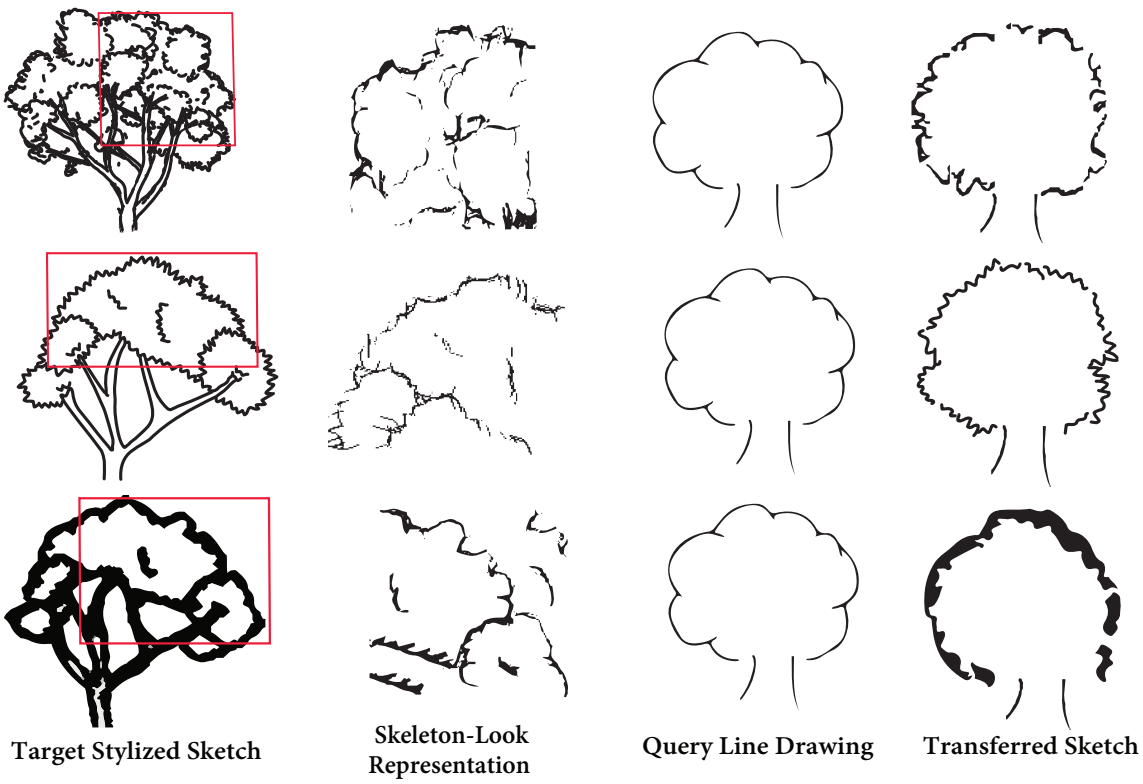


Figure 4.11: Examples of applying sketch style transfer to the same query sketch.

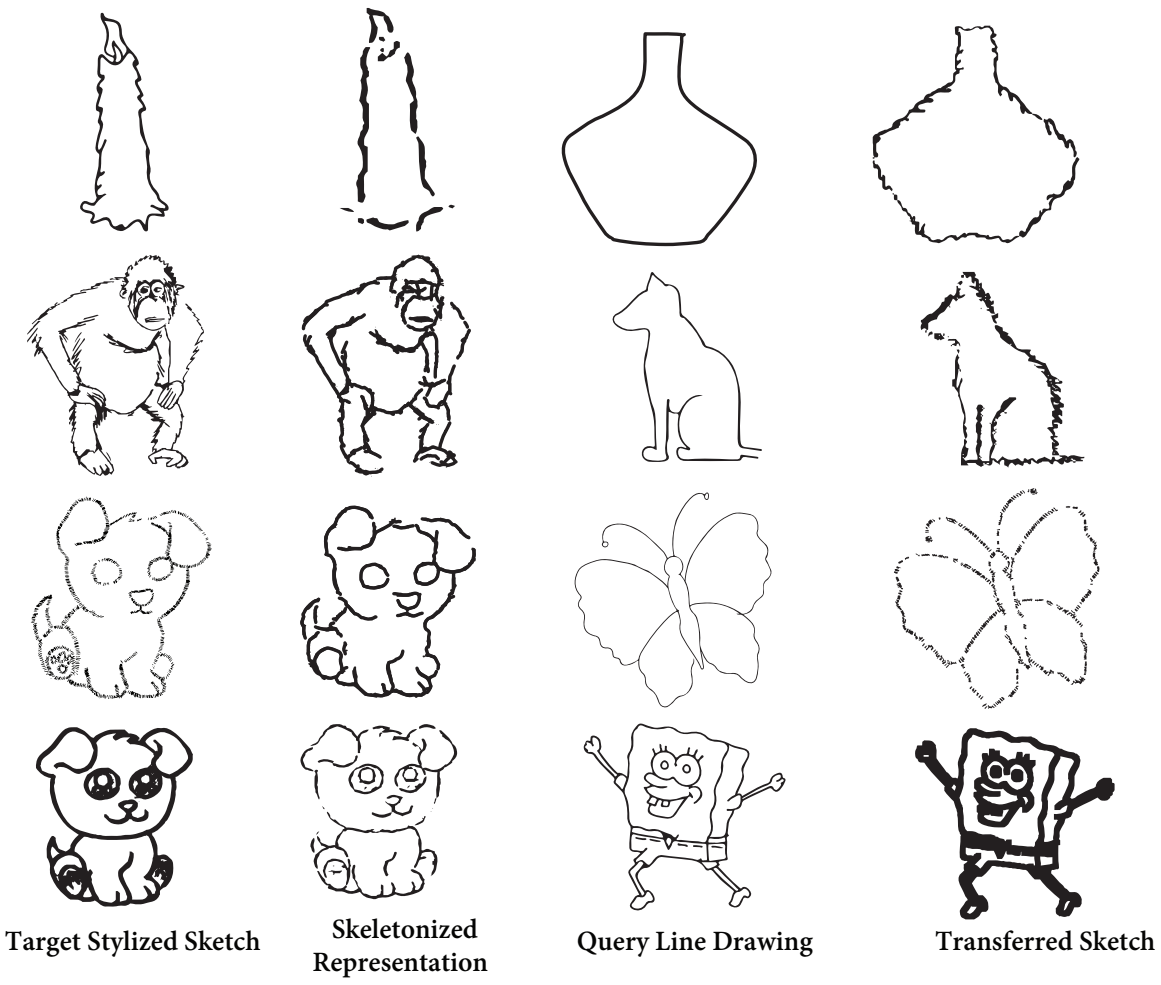


Figure 4.12: Examples of sketch style transfer of various query and target sketches.

imized. Our global sketch representation is based on insights from our stroke authorship recognition technique, discussed in chapter 3, which assumes that an artist's style can be recognized by the frequency distribution of certain mathematical descriptors collected from basic sketching strokes. Brush-style-transfer, on the other hand, aims to transfer the brush look of a sketch to an input line-drawing. For this stage of transfer, we find a mapping between a stroke segment and its corresponding brush element in the stylized sketch. This is done automatically by using convolutional sparse coding (CSC). As it is intuitively and empirically validated in our work, CSC is found to model the sketching process well. In fact, brush-style transfer becomes a matter of mapping the stroke segments to their corresponding brush element patches.

With these two methods of sketch transfer, our sketch style transfer technique can be used to automatically apply both subtle and major transformations on sketches. Furthermore, it can be used to transfer a sketching style independent of the sketched content and for sketches drawn either digitally or by hand. It is also capable of transferring a sketch style to any target style of a given example sketch. Our pipeline adds a new comprehensive perspective to the problem of style transfer. Our extensive experiments show that our transfer results are visually appealing both qualitatively and quantitatively (through a large user study).

Chapter 5

Sketch Geometric Modeling

5.1 Introduction

Sketch vectorization, which is defined as representing a sketch as a number of connected parameterized curves, is of great demand in the graphics community. It enables sketch transformations that are invariant to scale and non-rigid body transformations. It also fuels various applications related to sketch style manipulation, analysis and synthesis [11]. Previous work on those topics tends to avoid the non-trivial geometric solutions of sketch vectorization [12] and replaces it with some manual interaction to access strokes of sketches (strokes are the building blocks of a sketch). This involves pre-processing steps such as building large libraries of strokes or digitally collecting sketches using the Wacome device [7, 13]. As such, utilizing existing modern computing algorithms for *automatic* sketch parametric representation is a valuable contribution to the graphics research community.

In this chapter, we propose an extension to the formulation of the well-known convolutional sparse coding (CSC) model, such that it can better model the reconstruction process of line drawings. CSC plays an essential role in many computer vision applications ranging from image compression to deep learning. Here, we spot the light on a new application where CSC can effectively serve, namely line drawing analysis. This can be done by constraining the filters in CSC to a pre-defined group of parametric curves. This results in learned filters that, in their content, describe the sketch geometrically as a set of curves. Such representation can be utilized for sketch vectorization. We believe that our work is the first to introduce geometry constraints to the standard CSC formulation.

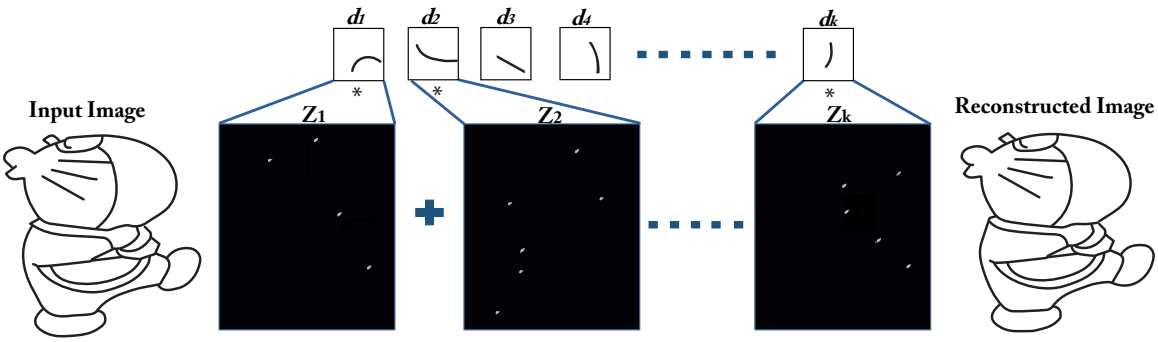


Figure 5.1: Constrained Convolutional Sparse Coding (CSC) Model. An input image is represented by a sum of dictionary elements belonging to set of parametric curves convolved with corresponding sparse maps and resulting in a reconstructed image that is similar to the input image.

CSC represents an image by a sum of convolutional responses generated from image filters and sparse maps. These filters constitute a CSC dictionary $\mathbf{D} = \{\mathbf{d}_1, .., \mathbf{d}_K\}$, and when convolved with their corresponding sparse maps $\mathbf{Z} = \{\mathbf{z}_1, .., \mathbf{z}_K\}$, the image is reconstructed. Given an input image $\mathbf{x}$, the CSC optimization is described in Eq (5.1).

$$\begin{aligned} \underset{\mathbf{d}_k, \mathbf{z}_k \; \forall k}{\arg\min} \; & \frac{1}{2}||\mathbf{x} - \sum_{k=1}^{K} \mathbf{d}_k * \mathbf{z}_k||_2^2 + \beta \sum_{k=1}^{K} ||\mathbf{z}_k||_1 \\ \text{s.t.} \;\; & ||\mathbf{d}_k||_2^2 \leq 1 \;\; \forall k \in \{1, ..., K\} \end{aligned} \tag{5.1}$$

CSC approximates how sketches are drawn. If the filters in the dictionary are generated from actual parametric strokes, then the summation of convolution responses between these filters and the sparse maps specify *where* in the image a stroke emerges and the set of filters reflect *what* the response (stroke) will look like. We believe that this model approximates how an artist would draw a sketch; however, a more accurate depiction of this process would be to replace the summation with a spatially localized max operation (representing a union instead of a sum). Unfortunately, this higher fidelity model makes the optimization significantly harder to do, although it could still be solved using standard fixed point (alternating) optimization techniques popularly used for bi-convex problems.

To validate our choice of CSC for the task of line drawing analysis, we conducted a

user experiment. Using a Wacom and a stylus pen, three artists are asked to freely draw a given sketch. We parse the Wacom data files containing the sketches and divide the sketches into square areas of 30×30 pixels, where the whole image is of 300×300 pixel resolution. For each square region, we count the average number of times an artist passes his/her pen through this area. This is done using the spatial and temporal information generated from the Wacom device. We eliminate areas that contain no strokes. According to this experiment, the sparsity of strokes used in patches of a sketch reaches a reasonable amount with only 1-pass with the stylus pen over a squared region 60% of the times. This observation validates the use of sparse response maps, thus, justifying the overall use of CSC.

Figure 5.1 gives an illustrative example of how our proposed constrained CSC model is used for line drawing reconstruction using filters that are learnt from a set of parametric curves. We solve the resulting non-convex optimization using ADMM. Through experiments, we empirically validate the convergence of our method to a feasible solution. As this novel method is expected to provide a reliable and automatic method for parametric sketch representation, a number of sketch manipulation examples are demonstrated in this chapter.

5.2 Constrained Parametric CSC Optimization

In this section, we give a brief overview of the standard CSC model. Then, we reformulate the problem to include our proposed parametric projection constraint. In this work, we consider the CSC problem with circular boundary conditions as discussed in the work of Brisow *et al*. [10].

5.2.1 Standard CSC Model

The CSC problem is generally expressed in Eq (5.1), where $\mathbf{d}_k \in \mathbb{R}^M$ are the vectorized 2D patches representing K dictionary elements, and $\mathbf{z}_k \in \mathbb{R}^D$ are the vectorized sparse

maps corresponding to each of the dictionary elements. The data term represents the image $\mathbf{x} \in \mathbb{R}^D$ as the sum of 2D convolutions of the dictionary elements with the sparse maps, and the ℓ_1 term is added to enforce sparsity onto the feature maps with β controlling the level of sparsity. The CSC objective function can be easily extended to multiple training images, where K sparse maps are learned for each image and all images share the same K dictionary elements.

The objective in Eq (5.1) is not convex, yet it is bi-convex. So, solving it for one variable while keeping the other fixed leads to two convex subproblems: the coding subproblem and the dictionary learning subproblem.

Learning Subproblem. The learning subproblem is shown in Eq (5.2). It learns the dictionary elements for a set of sparse feature maps.

$$\arg\min_{\mathbf{d}} \ \frac{1}{2}\|\mathbf{x} - \mathbf{Z}\mathbf{d}\|_2^2 \quad \text{s.t.} \ \ \|\mathbf{d}_k\|_2^2 \leq 1 \ \ \forall k \tag{5.2}$$

Here, $\mathbf{Z} = [\mathbf{Z}_1 \dots \mathbf{Z}_K]$ is a concatenation of Toeplitz convolution matrices of the sparse maps, and $\mathbf{d} = [\mathbf{d}_1^T \dots \mathbf{d}_K^T]^T$ is a concatenation of all the dictionary elements.

Coding Subproblem. The coding subproblem is detailed in Eq (5.3). It computes optimal sparse maps for a set of dictionary elements.

$$\arg\min_{\mathbf{z}} \ \frac{1}{2}\|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2 + \beta\|\mathbf{z}\|_1 \tag{5.3}$$

Similar to above, $\mathbf{D} = [\mathbf{D}_1 \dots \mathbf{D}_K]$ is a concatenation of Toeplitz convolution matrices of the dictionary elements, and $\mathbf{z} = [\mathbf{z}_1^T \dots \mathbf{z}_K^T]^T$ is a concatenation of the vectorized sparse maps.

One approach to solve each of the above subproblems is to cast it in an ADMM framework. Also, it can be solved efficiently in the Fourier domain. For more details about the

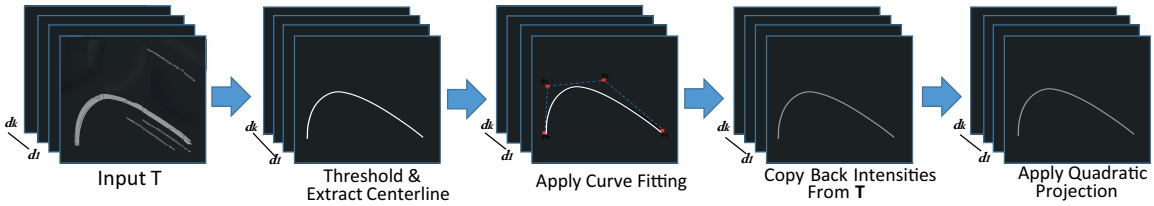


Figure 5.2: The work flow of parametric curve projection for each filter patch: (1) given an input filter patch T ;(2) extract the largest set of connected pixels with intensity values above a threshold and then extract its centerline; (3) apply cubic Bézier curve fitting; (4) copy the intensity values of pixels where the fitted curve located at T to the generated curve patch; (5) apply quadratic projection to the curve patch.

CSC model and for a discussion on various methods to solve it, we refer the reader to the work of Bristow *et al*. [74].

5.2.2 Constrained Parametric CSC

In this section, we describe the variations we make on the standard CSC model to solve for a constrained parametric CSC (i.e.to allow for filters to be constrained to a set of parametric curves). This reformulation enforces geometrical properties on the filters during the training step. As we explain the model, it will become obvious how such reformulation enables a parametric curve representation of the reconstructed line drawings. We reformulate the CSC problem by adding a parametric constraint as follows:

$$\begin{aligned} \underset{\mathbf{d}_k, \mathbf{z}_k \; \forall k}{\arg\min} \; & \frac{1}{2} \|\mathbf{x} - \sum_{k=1}^{K} \mathbf{d}_k * \mathbf{z}_k\|_2^2 + \beta \sum_{k=1}^{K} \|\mathbf{z}_k\|_1 \\ \text{s.t.} \;\; & \mathbf{d_k} \in S_b; \; \forall k \in \{1, ..., K\} \end{aligned} \tag{5.4}$$

This constraint is added to the learning subproblem so that the learned dictionary patches are geometrically constrained to belong to a set of parametric curves S_b. We define the set of parametric curves as follows:

$$S_b = \left\{ \mathbf{a} : \|\mathbf{a}\|_2^2 \leq 1 \quad s.t. \text{ support of } \mathbf{a} \in f(t) \right\}, \tag{5.5}$$

where the support of $\mathbf{a}$ denotes the set of pixel indices with non-zero intensity in $\mathbf{a}$ and $f(t)$ is the cubic Bézier curve function in Eq (5.6).

$$f(t) = \sum_{i=0}^{3} \mathbf{p}_i B_i^3(t), \quad t \in [0, 1]\,, \tag{5.6}$$

where $\mathbf{p}_i$ is the i^{th} control point of the 2D curve, and $B_i^3(t)$ is the cubic Bernstein polynomial (i.e. $(1-t)^3,\ 3t(1-t)^2, 3t^2(1-t),\ t^3$) [87]. By manipulating the control points of a Bézier curve, it can be intuitively deformed. Cubic Bézier curves are enough to model various curve shapes and only four control points $\{\mathbf{p}_i\}_{i=0}^3$ are needed as shown in Figure 5.3. Unlike normal functions, parametric Bézier curves do not define a y coordinate in terms of an x coordinate. Instead, they link the values to *control* variables. If we vary the value of t, then with every change we get two new values, which we can use as (x, y) coordinates in the curve plot. The variable t can be viewed as the variable which we sample the curve along as its value changes over a defined range [88]. We choose cubic Bézier curves to represent underlying strokes in line drawings, since they are widely used in computer graphics and geometric modeling applications that focus on smooth curves [87, 88].

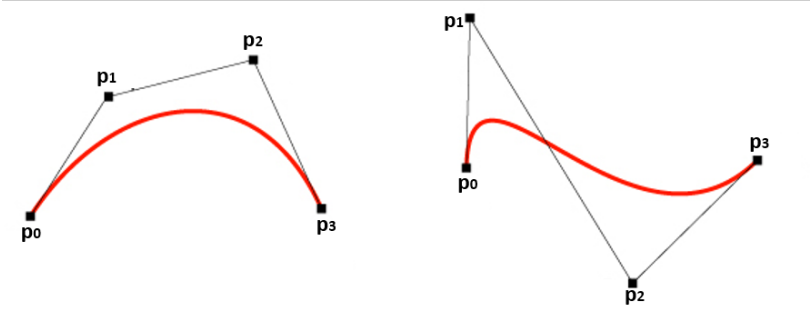


Figure 5.3: Examples of cubic Bézier curves. Four control points are enough to represent the curve.

Learning Subproblem. Changes on the standard CSC are only reflected in the dictionary learning subproblem. The learning subproblem corresponding the constrained parametric CSC is shown in Eq (5.7). The inequality constraint on the dictionary elements shown in

Eq (5.1) is now embedded as part of the parametric projection proximal of the set S_b as defined earlier in Eq (5.5). To be able to solve this subproblem using ADMM, we add the second equality constraint and introduce an auxiliary variable $\mathbf{u}$ as follows:

$$\arg\min_{\mathbf{d},\mathbf{u}} \ \frac{1}{2}\|\mathbf{x}-\mathbf{Z}\mathbf{d}\|_2^2 \ \text{ s.t. } \begin{cases} \mathbf{u_k} \in S_b \\ \mathbf{u_k} = \mathbf{d_k} \end{cases} \quad \forall k \tag{5.7}$$

Using an indicator $\mathbb{I}$ for the constraint, we can move it into the objective yielding the following objective with one constraint, which constitutes our proposed constrained parametric CSC:

$$\begin{aligned} \arg\min_{\mathbf{d},\mathbf{u}} \ & \frac{1}{2}||\mathbf{x}-\mathbf{Z}\mathbf{d}||_2^2 + \sum_{k=1}^{K} \mathbb{I}_{\{\mathbf{u_k}\in S_b\}} \\ \text{s.t. } & \mathbf{d_k} = \mathbf{u_k};\ \forall k \in \{1,...,K\} \end{aligned} \tag{5.8}$$

To apply ADMM to Eq (5.8), we define the augmented Lagragian function $\mathcal{L}$ as follows:

$$\begin{aligned} \mathcal{L}(\mathbf{d},\mathbf{u},\boldsymbol{\lambda}) := & \frac{1}{2}||\mathbf{x}-\mathbf{Z}\mathbf{d}||_2^2 + \sum_{k=1}^{K} \mathbb{I}_{\{\mathbf{u_k}\in S_b\}} \\ & + \boldsymbol{\lambda}^{\boldsymbol{T}}(\mathbf{u}-\mathbf{d}) + \frac{\rho}{2}||\mathbf{u}-\mathbf{d}||_2^2 \end{aligned} \tag{5.9}$$

where $\boldsymbol{\lambda}$ is the Lagrangian dual variable of the constraint. The iterative ADMM steps to minimize Eq (5.9) w.r.t. the primal variables $(\mathbf{d},\mathbf{u})$ are described in Algorithm1. ADMM updates are performed by optimizing for the variables $\mathbf{d}$ and $\mathbf{u}$ one at a time, while keeping the other. Then, the dual variable $\boldsymbol{\lambda}$ is updated using gradient ascent. Now, we list the optimization solutions to update the primal variables in the ADMM algorithm.

Update $\mathbf{d^{t+1}}$: At the t^{th} iteration, the update of this variable is a simple least squares problem, which has the following solution: $\mathbf{d^{t+1}} = \left(\mathbf{Z}^T\mathbf{Z}+\rho I\right)^{-1}\left(\mathbf{Z}^T\mathbf{x}+\boldsymbol{\lambda}^{\boldsymbol{t}}+\mathbf{u^t}\right)$. We initialize $\mathbf{d}$ as randomly generated parametric curves.

Algorithm 1 ADMM for learning subproblem

Set ADMM optimization parameter $\rho > 0$
Initialize variables $\mathbf{d}$, $\mathbf{u}$, $\boldsymbol{\lambda}$
while not converged **do**
    $\mathbf{d^{t+1}} = \arg\min_d \frac{1}{2}||\mathbf{x} - \mathbf{Zd}||_2^2 - \boldsymbol{\lambda^T}\mathbf{d}) + \frac{\rho}{2}||\mathbf{u} - \mathbf{d}||_2^2$
    $\mathbf{u^{t+1}} = prox_{\frac{f}{\rho}}\left(\mathbf{d^{t+1}} + \frac{\boldsymbol{\lambda^t}}{\rho}\right)$
    where $f(u) = \sum_{k=1}^{K} \mathbb{I}_{\{\mathbf{u_k} \in S_b\}}$

    $\boldsymbol{\lambda^{t+1}} = \boldsymbol{\lambda^t} + \rho\,(\mathbf{u^{t+1}} - \mathbf{d^{t+1}})$
    $\rho = \rho + c$
Output solution variables

Update $\mathbf{u^{t+1}}$: Updating this variable is done through the proximal operator $g(.)$ for the indicator function of set S_b. Mathematically, the underlying optimization is:

$$g(\mathbf{T}) = \arg\min_{\mathbf{u}} \quad \frac{\rho}{2}\|\mathbf{u} - \mathbf{T}\|_2^2 + \sum_{k=1}^{K} \mathbb{I}_{\{\mathbf{u_k} \in S_b\}}$$

where $\mathbf{T} = \mathbf{d^{t+1}} + \frac{\boldsymbol{\lambda^t}}{\rho}$. Consequently, the goal of our parametric projection proximal operator $g(.)$ is to find the best cubic Bézier curve, which defines the support of an image patch *closest* in intensity to $\mathbf{T}$. In this way, we are geometrically guiding the indices of the non-zero elements of the learned filters to form a proper Bézier curve. To achieve that, we develop a method to perform this parametric projection to a Bézier curve, as illustrated in Figure 5.2. First, we extract the largest set of connected pixels with intensities above the average intensity of $\mathbf{T}$. We then apply morphological skeletonization [96] to extract the centerline of those high intensity pixels. This will generate a skeleton representation of the extracted pixels. Next, we fit a cubic Bézier curve on the extracted skeleton using Eq (5.6) and then copy the intensity values at these pixels from $\mathbf{T}$. Lastly, we project the resulting image patch onto the unit ball to enforce that $\|\mathbf{u_k}\|_2^2 \leq 1;\ \forall k \in \{1, ..., K\}$. This ensures that the solution vector is of the proper scale. It is important to mention that the set S_b is a non-convex set, thus, the parametric projection is not necessarily globally optimal.

However, our experiments, as we demonstrate later, show that our multi-stage technique to define the proximal operator $g(.)$ converges to feasible solutions.

Coding Subproblem. The coding subproblem of the constrained parametric CSC is the same as in the standard CSC discussed in Section 5.2.1. It is solved by applying ADMM where the update steps involve solving a linear system and a proximal operator for the ℓ_1 norm. The linear system is solved efficiently in the Fourier doamin making use of the circulant structure of the convolution matrices. The proximal operator, on the other hand, is solved using soft-thresholding. We refer the reader to the work of Heide *et al*. [72] for details on how to solve the coding subproblem and for overall complexity analysis.

5.3 Experimental Results and Evaluation

In this section, we provide an assessment and a discussion of the proposed constrained parametric CSC. We start with an overview of the implementation details and the parameters selected throughout our experiments. Then, we discuss the convergence of our technique and provide visual evaluation of the learned parametric filters, as compared to standard CSC, along with image reconstruction quality. Towards the end of this section, we present some qualitative examples of sketch synthesis and manipulation using our parametric model.

5.3.1 Implementation Details

The line drawings used in our experiments are from the free style line drawings dataset of the stroke authorship recognition model presented in chapter 3. It comprises 70 clean line drawings of multiple experienced artists. Before learning the dictionary elements, we apply contrast normalization to the sketch images to generate normalized images with black background and white sketch drawings in the foreground. All these images of 300×300 spatial resolution.

The number of dictionary elements is chosen to be $K = 100$, where each element has a spatial size of 31×31 pixels. We found that this size of individual dictionary elements is representative enough of the different curves in a line drawing, thus, allows for a various collection of Bézier curves to be fit to the learned image filters. Smaller dictionary elements, on the other hand, lead to non-significant details per dictionary patch while larger filter sizes carry lots of details such that it is impossible to fit a single Bézier curve per dictionary patch. We initialize each dictionary element as a normalized image with black background and a white foreground with a support that traces a *randomly generated* Bézier curve.

We choose the value of the sparsity coefficient $\beta = 0.6$, which we find to deliver a good trade-off between sparsity and data fit. Moreover, for the ADMM learning steps we start with a value of $\rho = 0.1$ which increases by 2% at every iteration.

5.3.2 Simple Reconstruction Example

In Figure 5.4, we present an example for constructing a simple primitive shape (a circle) using $K = 4$ dictionary elements using our constrained parametric CSC model. Starting with initial dictionary patches that are almost linear Bézier curves, it is obvious how the dictionary patches in the last iteration have changed to represent a new set of curves that are of better representation power for the end shape (a circle). Using this simple example, we demonstrate how our constrained parametric CSC model works in general. The learned parametric filters can now be used to describe the reconstructed image geometrically. Moreover, the quality of reconstruction is reflected on how the constructed circle image is very close to the input.

5.3.3 Evaluation of Constrained Parametric CSC

Multiple intuitive questions arise while discussing our proposed parametric CSC, which we address in this section. First, we study the reconstruction quality of the parametric CSC and

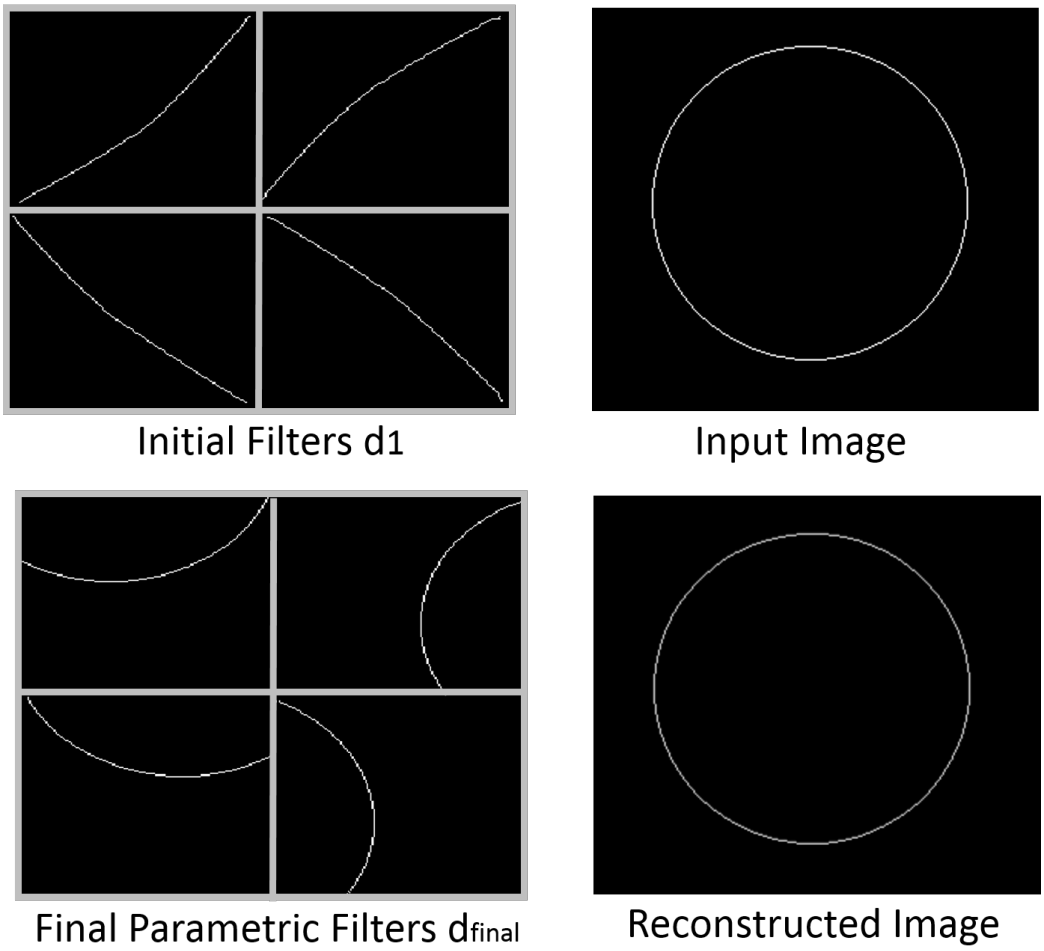


Figure 5.4: An example of reconstructing a circle using our constrained parametric CSC. Filters in the last iteration have deformed to a new set of curves that are clearly representing the reconstructed image and describe it geometrically. Furthermore, the reconstructed image quality is preserved when compared with the input image.

show how the dictionary elements look like upon convergence. As illustrated in the first 2 columns of Figure 5.5, learned dictionary elements (strokes) change significantly when compared against the initial random curves. Such change yields to dictionary elements that represent the image geometrically through the parametrically defined filters. Also, we see that the reconstruction quality is good with a Peak Signal-to-Noise Ratio (PSNR) of 29dB.

We also compare our parametric model with standard CSC. Dictionary elements learned on the standard model are illustrated in Figure 5.5. It is obvious that they do not carry clear geometric information evident in line drawings and they cannot be used for parametric

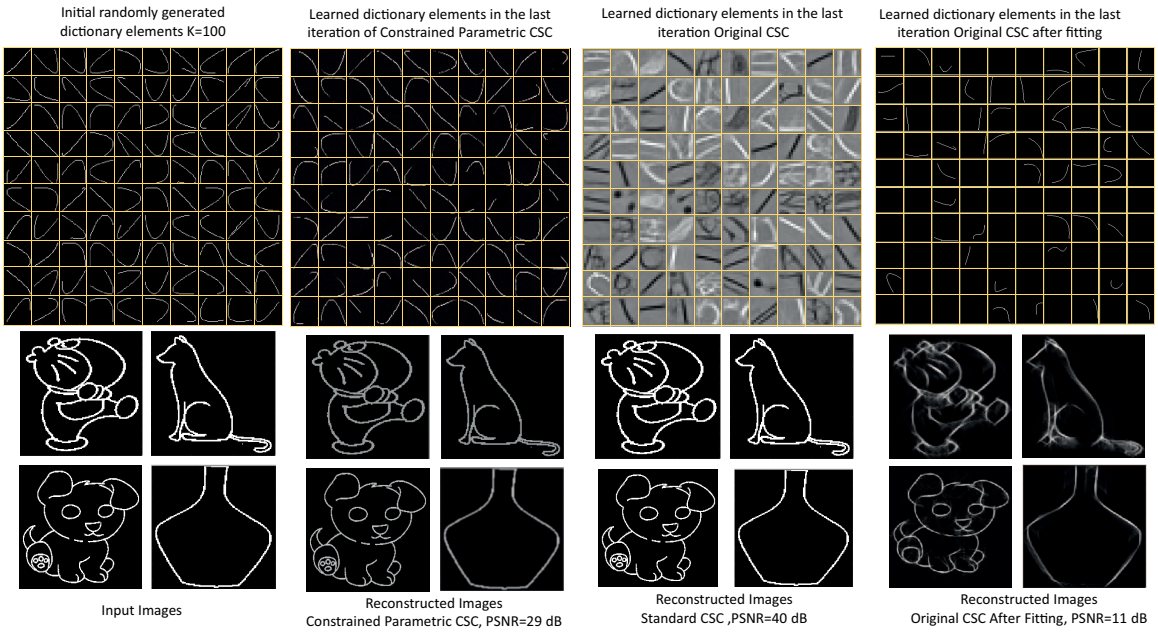


Figure 5.5: A comparison of the the dictionary learning elements and the quality of the reconstructed images between (1) the constrained parametric CSC, (2) the standard CSC and (3) the standard CSC with curves fitting (particularization) in the last iteration.

representation of sketches. However, the reconstruction quality of the standard CSC is higher than our parametric CSC with PSNR=40dB. This is expected, since standard CSC ideally offers a lower bound objective to that of our constrained parametric CSC. It is important to mention that all experiments are conducted at the same sparsity level (of 60%) for the sparse codes.

To evaluate the need for applying the proximal operator $g(.)$ within the ADMM framework, we apply this operator to fit cubic Bézier curves to the filters learned by standard CSC. As shown in Figure 5.5, these learned filters are not representative of the content of the line drawing, since it is hard to fit one curve to each filter generated through standard CSC. Projecting these learned filters onto S_b generates subpar results that are not reflective of the strokes drawn by the artist. In addition, the reconstruction quality using these projected filters is 18dB lower than our constrained parametric CSC. Therefore, we conclude that our method is capable of generating high reconstruction quality line drawings, while constraining its filters to be parametric Bézier curves.

5.3.4 Convergence

Our constrained parametric CSC optimization is non-convex in general. This is because the set S_b in Eq (5.7) is non-convex, i.e.the convex combination of any two patches with cubic Bézier curve support does necessarily generate a patch whose support is a cubic Bézier curve. Despite this non-convexity, our experiments show that the ADMM iterations do empirically converge (refer to Figure 5.6). The plots in Figure 5.6 are generated when constructing multiple line drawings with 300×300 pixel resolution and with $K = 100$ dictionary elements of resolution 31×31 pixels, similar to Figure 5.5. The first plot on top shows how the overall objective value of parametric CSC decreases as the fixed point (or alternative optimization) iterations progress, refer to Eq (5.4). The second plot is an evidence of convergence for a single learning subproblem that uses ADMM, Eq (5.8). The learning objective eventually decreases and then saturates after a certain number of iterations. The early increase in this plot is due to the fact that the current solution is *not* feasible, i.e.the variables $\mathbf{d}$ and $\mathbf{u}$ are far away from each other. This is clear from the bottom plot, which measures how the primary variables $\mathbf{d}$ and $\mathbf{u}$ are getting closer as the ADMM iterations progress for the single learning subproblem. As this distance measure reaches zero, the resulting variables are guaranteed to be the same, thus, reaching a feasible solution to the constrained problem in Eq (5.7).

As in any non-convex optimization problem, initialization of the optimization variables is an important element to solve the problem so it converges to a feasible solution. Our parametric optimization, on the other hand, is not significantly sensitive to initialization. This is evident on how our optimization is able to converge and generate promising results when using randomly generated cubic Bézier curves images as its initial dictionary elements. However, the closer the initialized parametric filters to the actual stroke of the drawings, the faster the convergence and the higher the overall reconstruction quality. Lastly, we leave the proof of convergence of ADMM on this non-convex problem to future work.

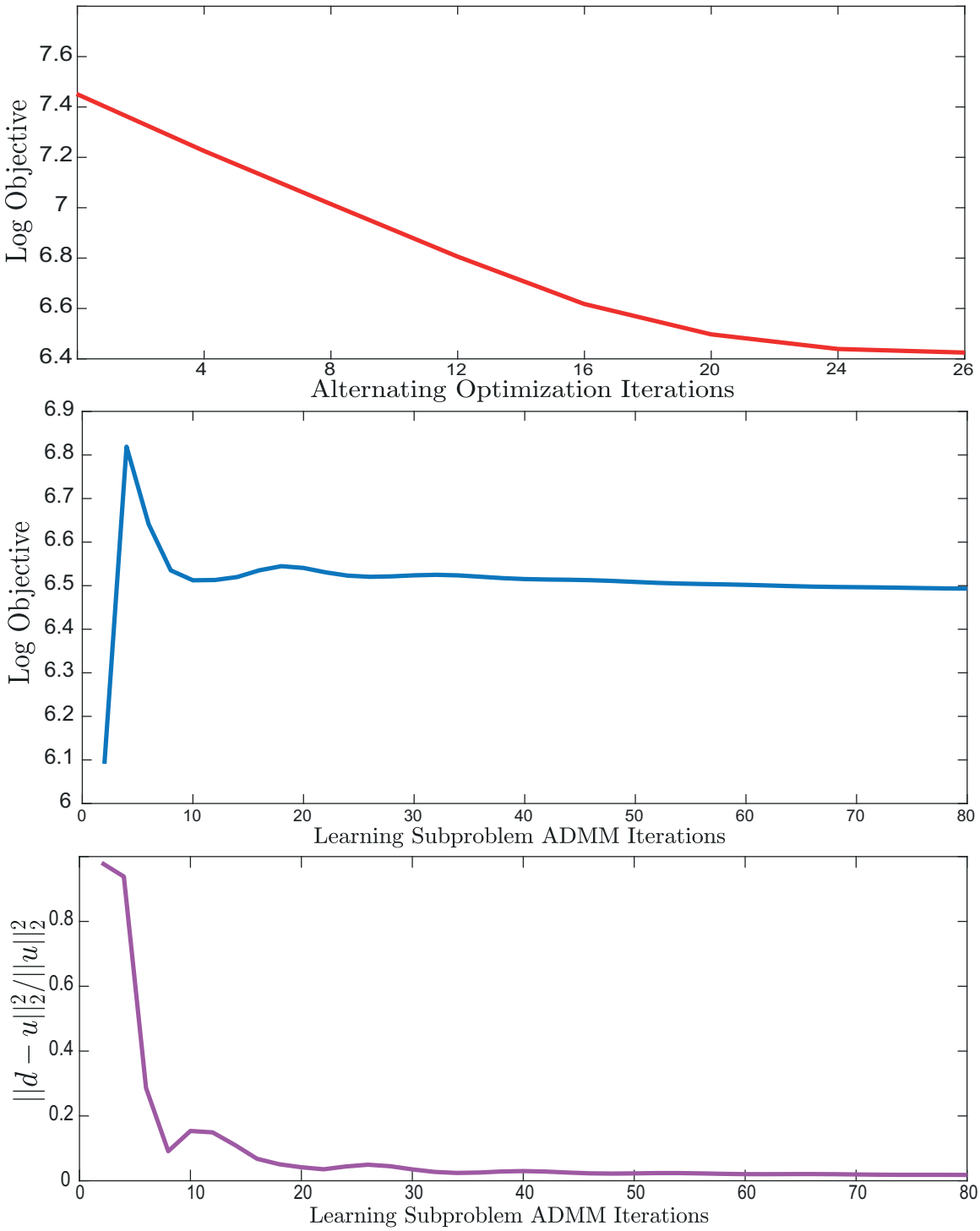


Figure 5.6: Convergence experiments. The first plot shows how the objective function value decreases over alternating optimization iterations. The second plot shows how the ADMM subproblem objective value decreases over the dictionary learning iterations and then saturated. The third plot shows how the primary variables $\mathbf{u}$ and $\mathbf{d}$ are getting closer to each other until the difference between them is almost 0.

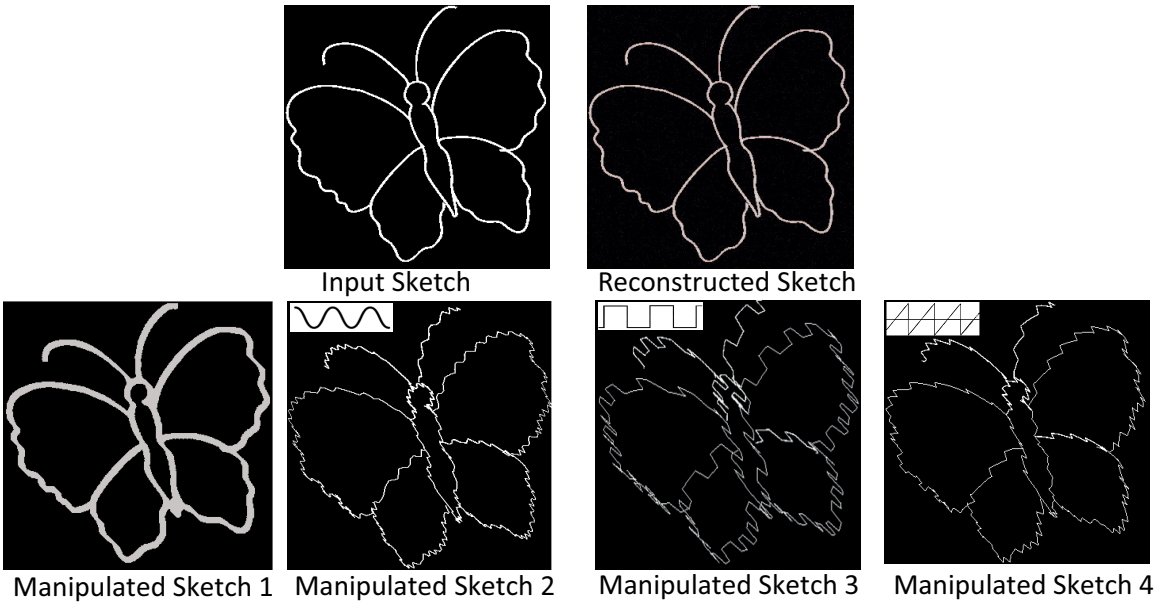


Figure 5.7: Examples demonstrating how sketch stylization and manipulation is enabled upon having parametric representation of sketches.

5.3.5 Qualitative Sketch Manipulation Results

As mentioned earlier in Section 5.1, parametric representation of sketches is vital to enable for automatic sketch synthesis and manipulation. Here we demonstrate a number of line drawing manipulation or stylization examples (i.e. embedding a new style into a line drawing) on one hand-drawn sketch as shown in Figure 5.7. Given the parametric filters learned by our constrained parametric CSC model, as in Figure 5.5, we can now directly apply various geometric transformations to those parametric curves. This will generate a set of new dictionary elements which we use to reconstruct the same line drawing using the same sparse feature maps inferred from the constrained parametric CSC model. Consequently, this will embed a new style to the strokes of the line drawing.

The first manipulation example we include in this work aims to increase the thickness of strokes in the sketched line drawing. This is simply achieved by increasing the thickness of the parametric curve in each patch by generating the same curve and take two copies of it and place them one pixel above and one pixel below the original curve. The process of shifting and copying the curve can be repeated as many times as needed around the original

parametric curve with increasing pixel values until getting the desired thickness. Then we replace the parametric learned filters with the modified ones and reconstruct the line drawings using the earlier inferred feature maps to embed this new style. Thickness stylization result is shown on the sketch labeled as "Manipulated Sketch 1" of Figure 5.7. Another kind of manipulation can be generated by replacing each parametric curve with a periodic function, $g(t)$, over the curve path. We choose three types of periodic functions: sin, square and sawtooth periodic functions. Such stylization can be accomplished by replacing each parametric curve $f(t)$ in Eq (5.6) with $f(t) + g(t)$, where $g(t)$ is a periodic function. Similarly, this will generate a new set of dictionary elements which are used to reconstruct the line drawing to embed the new style. Manipulation results are shown in Figure 5.7. It shows examples of periodic sin, square and sawtooth manipulations in the sketches labeled "Manipulated Sketch 2" to "Manipulated Sketch 4" respectively.

5.3.6 Conclusion

In this chapter, we present an automatic method for representing a sketch geometrically as a set of parametric curves. It is based on utilizing the well known convolutional sparse coding (CSC) model. This is based on our observation that CSC is closely related to the line drawing process. The i^{th} map $\mathbf{z}_i$ can be viewed as a sparse score map, which scores each pixel based on whether the filter or stroke $\mathbf{d}_i$ is localized there or not. This scenario resembles how one would draw a sketch, i.e.making the decision what stroke to draw and where to place it. To make it a better fit to the task of line drawing analysis, we reformulate the standard CSC model such that the dictionary elements are constrained to belong to a set of images, whose support is a parametric cubic Bézier curve. As such, sketches are represented geometrically through the learned CSC filters. Such reformulation leads to a non-convex learning subproblem of the CSC, which we solve using ADMM. Although convergence is not theoretically guaranteed in this non-convex case, experiments show that our algorithm reliably converges to a good solution.

Chapter 6

Concluding Remarks

In this chapter, the main conclusions and contributions of the previous work will be summarized and additional comments for future work will be provided. In this dissertation, we provided new techniques and applications in the field of sketch analysis, style transfer and geometric representation. The conclusions of each proposed technique is presented below:

Sketch Authorship Recognition (Chapter 3): In this chapter, we shed light on new direction in sketch analysis, namely authorship recognition through stroke analysis. We propose a stroke authorship recognition (SAR) approach that discriminates between artistic sketch styles based on the choice and frequency of use of basic strokes. It is motivated by real-world issues, including the fact that fraudulent copies of original sketches with patent and copyright protection are sold on some websites, which claim them to be original. As it is hard for an individual to discriminate authentic and fraudulent sketches (section 3.4), original sketches from known companies and artists are in danger of replication and copyright infringement. Moreover, cartoon artists undergo months of training before they can officially contribute, so as to ensure that the style in which the characters are drawn is not distorted when artists are replaced (Section 3.6.1). Similar to handwriting and signatures, sketches can be used for user authentication as discussed in chapter 2. Thus, detecting major and minor differences in sketching style is important. Furthermore, design patents that contain sketches of commercial designs need to be protected, as is apparent in the ongoing lawsuits between Apple and Samsung for example. All of this shows the need to have automatic tools, such as SAR, to assess the authenticity of sketched objects and identify their

authorship. Our new shape representation and authorship classification technique is based on analyzing local features of the sketch's silhouettes and internal strokes which are then used to represent a sketch as a histogram of universal strokes segments. The later representation is used in a k-NN based authorship classification model. The proposed method is applicable to sketches in various sizes, and invariant to non-rigid motion, scaling and rotation.

From our extensive experiments and user studies (section 3.5.1), we provide empirical evidence that SAR *does* encode unique and consistent characteristics of an artist's sketching style, which are in turn used to discriminate one artist's sketches from others. We also showed that the extent to which SAR can be used for discrimination is dependent on the sketching constraints imposed on the artist. Moreover, our extensive user studies empirically validate the difficulty of this fine-grained recognition problem and indicate that SAR can improve upon human performance. All the compiled data (datasets and user studies) and source code are publicly available to enable further research on this exciting topic and to allow for quantitative comparison. Below is a listed summary of major findings and experiments of our stroke authorship recognition technique:

- *Uniqueness of Sketch Style.* Based on results in Section 3.5.1, we conclude that SAR *does* encode unique and consistent characteristics of an artist's sketching style, which are in turn used to discriminate one artist's sketches from others. This result validates SAR's applicability in important real-world tasks, such as sketch fraud detection (e.g. for design patents and cartoon characters) and training/teaching artistic style.

- *Style and Sketch Constraints.* The extent to which SAR can be used for discrimination is dependent on the sketching constraints imposed on the artist. Although overall SAR accuracy decreases with more constraints, unique elements of artistic style still persist even under the strictest of constraints (fraud).

- *Silhouettes.* We identify that a sketch's silhouette is a richer source of discriminative

information than internal strokes in general.

- *Human Performance.* Our extensive user studies empirically validate the difficulty of this fine-grained recognition problem and indicate that SAR can improve upon human performance. All the compiled data (datasets and user studies) and source code will be made publicly available to enable further research on this exciting topic and to allow for quantitative comparison with future methods.

Sketch Style Transfer (Chapter 4): In this chapter, we present a new automatic technique for sketch style transfer from one hand drawing to another with different sketching style. It consists of two methods of transfer: stroke-style and brush-style transfer. Stroke-style transfer targets the transfer of sketching at the stroke level through modifying the parameters of stroke segments of a sketch. To do so, a global sketch distance between both query and target sketches is minimized. Our global sketch representation is based on insights from sketch style discrimination discussed in chapter 3. It assumes that an artist's style can be recognized by the frequency distribution of certain mathematical descriptors collected from basic sketching strokes. Brush-style-transfer, on the other hand, aims to transfer the brush look of a sketch to an input line-drawing. For this method of transfer, we find a mapping between a stroke segment and its corresponding brush element in the stylized sketch. This is done automatically by using convolutional sparse coding (CSC). As it is intuitively and empirically validated in our work, CSC is found to model the sketching process well. Based on that, a one-to-one mapping between brush image patches in a stylized sketch and corresponding stroke segments used to reconstruct this patch in its skeletonized version is found by overlaying the obtained skeleton-look on top of a line drawing. In fact, brush-style transfer becomes a matter of mapping the stroke segments to their corresponding brush element patches.

With its two method of sketch transfer, our sketch style transfer technique can be used to automatically apply both subtle and major transformations on sketches. Furthermore, it can be used to transfer a sketching style independent of the sketched content and for

sketches drawn either digitally or by hand. It is also capable of transferring a sketch style to any target style of a given example sketch. Our pipeline adds a new comprehensive perspective to the problem of style transfer. A number of sketch style transfer results of both modules are presented in 4.4. They sufficiently satisfy an assessment criteria which includes *style*, *identity* and *quality* requirements. Our style Transferred sketches are also validated through a recognition study where participants with high accuracy managed to match a transferred sketch with a sketch they believe was used as the target sketch in the transfer.

Sketch Geometric Modeling (Chapter 5): In this chapter, we present an automatic method for representing a sketch geometrically as a set of parametric curves. It is based on utilizing the well known convolutional sparse coding (CSC) model. This is based on our observation that CSC is closely related to the line drawing process. The i^{th} map $\mathbf{z}_i$ can be viewed as a sparse score map, which scores each pixel based on whether the filter or stroke $\mathbf{d}_i$ is localized there or not. A pixel location with a high score means that $\mathbf{d}_i$ contributes significantly at this pixel to form the corresponding patch in the original image $\mathbf{x}$. This scenario resembles how one would draw a sketch, i.e.making the decision what stroke to draw and where to place it. To make it a better fit to the task of line drawing analysis, we reformulate the standard CSC model such that the dictionary elements are constrained to belong to a set of images, whose support is a parametric cubic Bézier curve. As such, sketches are represented geometrically through the learned CSC filters. Such reformulation leads to a non-convex learning subproblem of the CSC. We solve the resulting non-convex optimization using ADMM. While proof of convergence is left for future work, our experiments show that our ADMM solver converges to a good feasible solution. We solve the problem efficiently in the Fourier domain [72].

6.1 Future Research Work

We aim to improve the discriminative nature of our sketch authorship recognition technique (chapter 3) by enhancing the learned universal stroke segment dictionary. One way to do this is to investigate how strokes are actually generated by the artists through tracking their hand movements while sketching using data collected in 3D through a tracking camera. We want to investigate if the way an artist holds the pen and positions his/her arm with respect to the plane conveys unique information discriminating a certain artistic style from others. We also aim to consider the way artists draw by collecting sketches using a digitizing pen so we can take advantage of the pressure, tilt and speed used during the creation of the drawing. Furthermore, for future work, we aim to utilize the learned parametric dictionary elements generated by the geometric representation model (discussed in chapter 5) to help design fully automated techniques for sketch synthesis and transfer.

REFERENCES

[1] C. Grimm, "Results of an observational study on sketching," WUCSE-2011-57, Washington University in St. Louis, Tech. Rep., 2011.

[2] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *TOG*, 2012.

[3] J. Lu, C. Barnes, C. Wan, P. Asente, R. Mech, and A. Finkelstein, "DecoBrush: Drawing structured decorative patterns by example," in *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 2014.

[4] I. Berger, A. Shamir, M. Mahler, E. Carter, and J. Hodgins, "Style and abstraction in portrait sketching," *TOG*, 2013.

[5] ADOBE, "Illustrator," 2010.

[6] F. Cole, A. Golovinskiy, A. Limpaecher, H. S. Barros, A. Finkelstein, T. Funkhouser, and S. Rusinkiewicz, "Where do people draw lines?" *TOG*, 2008.

[7] W. T. Freeman, J. B. Tenenbaum, and E. C. Pasztor, "Learning style translation for the lines of a drawing," *ACM Trans. Graph.*, vol. 22, no. 1, pp. 33–46, Jan. 2003.

[8] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa, "Sketch-based image retrieval: Benchmark and bag-of-features descriptors," *TVCG*, 2011.

[9] G. Norris, D. Sykora, A. Shamir, S. Coros, A. Hornung, R. Sumner, M. Simmons, B. Whited, and M. Gross, "Smart scribbles for sketch segmentation," 2013.

[10] H. Bristow, A. Eriksson, and S. Lucey, "Fast Convolutional Sparse Coding," *CVPR*, 2013.

[11] S. Shaheen, A. Rockwood, and B. Ghanem, "Sar: Stroke authorship recognition," *CGF*, 2015.

[12] G. Noris, A. Hornung, R. W. Sumner, M. Simmons, and M. Gross, "Topology-driven vectorization of clean line drawings," *TOG*, 2013.

[13] J. Lu, F. Yu, A. Finkelstein, and S. DiVerdi, "Helpinghand: example-based stroke stylization," *TOG*, vol. 31, no. 4, p. 46, 2012.

[14] S. Shaheen and B. Ghanem, "Stroke Style Transfer," in *EG 2017 - Short Papers*. The Eurographics Association, 2017.

[15] A. Limpaecher, N. Feltman, A. Treuille, and M. Cohen, "Real-time drawing assistance through crowdsourcing," *TOG*, 2013.

[16] J. Lu, F. Yu, A. Finkelstein, and S. DiVerdi, "Helpinghand: Example-based stroke stylization," in *TOG*, 2012.

[17] S. C. Hsu, I. H. Lee, and N. E. Wiseman, "Skeletal strokes," in *Proceedings of the 6th annual ACM symposium on User interface software and technology*, 1993, pp. 197–206.

[18] J. Xie, A. Hertzmann, W. Li, and H. Winnemöller, "Portraitsketch: Face sketching assistance for novices." ACM, 2014.

[19] S. Zhang, X. Gao, N. Wang, and J. Li, "Robust face sketch style synthesis," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 220–232, 2016.

[20] N. Wang, S. Zhang, X. Gao, J. Li, B. Song, and Z. Li, "Unified framework for face sketch synthesis," *Signal Processing*, vol. 130, pp. 1–11, 2017.

[21] N. Ben-Zvi, J. Bento, M. Mahler, J. Hodgins, and A. Shamir, "Line-drawing video stylization," *CGF*, 2015.

[22] J. Lu, C. Barnes, S. DiVerdi, and A. Finkelstein, "Realbrush: painting with examples of physical media," *TOG*, vol. 32, no. 4, p. 117, 2013.

[23] A. Hertzmann, N. Oliver, B. Curless, and S. M. Seitz, "Curve analogies," in *EGRW '02*, 2002, pp. 233–246.

[24] H. Li, H. Zhang, Y. Wang, J. Cao, A. Shamir, and D. Cohen-Or, "Curve style analysis in a set of shapes," *CGF*, 2013.

[25] M. Kim and H. J. Shin, "An example-based approach to synthesize artistic strokes using graphs," *CGF*, vol. 29, no. 7, pp. 2145–2152, 2010.

[26] E. Kalogerakis, D. Nowrouzezahrai, S. Breslav, and A. Hertzmann, "Learning hatching for pen-and-ink illustration of surfaces," *TOG*, 2012.

[27] M. Gerl and T. Isenberg, "Technical section: Interactive example-based hatching," *Comput. Graph.*, vol. 37, no. 1-2, pp. 65–80, Feb. 2013.

[28] K. Lang and M. Alexa, "The markov pen: Online synthesis of free-hand drawing styles," in *Proceedings of NPAR*, 2015, pp. 203–215.

[29] J. Fišer, P. Asente, S. Schiller, and D. Sýkora, "Advanced drawing beautification with shipshape," *Comput. Graph.*, vol. 56, no. C, pp. 46–58, May 2016.

[30] M. Lukáč, J. Fišer, J.-C. Bazin, O. Jamriška, A. Sorkine-Hornung, and D. Sýkora, "Painting by feature: Texture boundaries for example-based image creation," *ACM Trans. Graph.*, 2013.

[31] J. Northrup and L. Markosian, "Artistic silhouettes: a hybrid approach," in *Proceedings of the 1st international symposium on NPAR*. ACM, 2000, pp. 31–37.

[32] R. Ando and R. Tsuruno, "Segmental Brush Synthesis with Stroke Images," in *Eurographics 2010 - Short Papers*, H. P. A. Lensch and S. Seipel, Eds. The Eurographics Association, 2010.

[33] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk, "State of the art in example-based texture synthesis," in *EG-STAR*. Eurographics Association, 2009.

[34] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra, "Texture optimization for example-based synthesis," *TOG*, vol. 24, no. 3, pp. 795–802, 2005.

[35] H. Yang, Y. Kwon, and K. Min, "A stylized approach for pencil drawing from photographs," *CGF*, vol. 31, no. 4, pp. 1471–1480, 2012.

[36] S. Simhon and G. Dudek, "On the elaboration of hand-drawn sketches," in *International Computer Science Conference on Active Media Technology*. Springer, 2001, pp. 355–364.

[37] R. K. Sarvadevabhatla and V. B. R, "Eye of the dragon: Exploring discriminatively minimalist sketch-based abstractions for object categories," in *Proceedings of the 23rd ACM International Conference on Multimedia*, 2015, pp. 271–280.

[38] J. McCrae and K. Singh, "Neatening sketched strokes using piecewise french curves," in *Proceedings of the Eighth Eurographics SBIM*. ACM, 2011, pp. 141–148.

[39] S. Simhon and G. Dudek, "Learning refinements on curve-strokes," in *TInternational Conference on Tools with Artificial Intelligence*, 2003, pp. 306–315.

[40] J. E. Kyprianidis, J. Collomosse, T. Wang, and T. Isenberg, "State of the art: A taxonomy of artistic stylization techniques for images and video," *TVCG*, vol. 19, no. 5, pp. 866–885, May 2013.

[41] M. Indu and K. Kavitha, "Survey on sketch based image retrieval methods," in *ICCPCT*. IEEE, 2016, pp. 1–4.

[42] Z. Sun, C. Wang, L. Zhang, and L. Zhang, "Sketch2tag: automatic hand-drawn sketch recognition," in *ACM MM*, 2012.

[43] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros, "Data-driven visual similarity for cross-domain image matching," in *SIGGRAPH*, 2011.

[44] F. Mokhtarian and A. Mackworth, "A theory of multiscale, curvature-based shape representation for planar curves," *PAMI*, 1992.

[45] S. Belongie, J. Malik, and J. Puzicha, "Matching shapes," in *ICCV*, 2001.

[46] R. Jin, A. Hauptmann, and R. Yan, "Image classification using a bigram model," in *AAAI Spring Symposium on Intelligent Multimedia Knowledge Management*, 2003.

[47] A. Berg, T. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," in *CVPR*, 2005.

[48] D. Michel, I. Oikonomidis, and A. Argyros, "Scale invariant and deformation tolerant partial shape matching," *IVC*, 2011.

[49] A. Ion, N. M. Artner, G. Peyré, W. G. Kropatsch, and L. D. Cohen, "Matching 2d and 3d articulated shapes using the eccentricity transform," *Computer Vision and Image Understanding*, 2011.

[50] S. N. Srihari and G. Leedham, "A survey of computer methods in forensic document examination," 2003.

[51] D. Impedovo and G. Pirlo, "Automatic signature verification: The state of the art," *IEEE Transactions on Systems, Man, and Cybernetics*, 2008.

[52] B. Kovari and H. Charaf, "A study on the consistency and significance of local features in off-line signature verification," *Pattern Recognition*, 2013.

[53] D. Rivard, E. Granger, and R. Sabourin, "Multi-feature extraction and selection in writer-independent off-line signature verification," *International Journal on Document Analysis and Recognition*, 2013.

[54] S. N. Srihari, S.-H. Cha, H. Arora, and S. Lee, "Individuality of Handwriting," *Journal of Forensic Sciences*, 2002.

[55] K. Franke and M. Kppen, "A computer-based system to support forensic studies on handwritten documents," *International Journal on Document Analysis and Recognition*, 2001.

[56] S. Srihari and Z. Shi, "Forensic handwritten document retrieval system," in *International Workshop on Document Image Analysis for Libraries*, 2004.

[57] C. Varenhorst, M. Kleek, and L. Rudolph, "Passdoodles: A lightweight authentication method," *Research Science Institute*, 2004.

[58] N. S. Govindarajulu and S. Madhvanath, "Password management using doodles," in *ICMI*, 2007.

[59] K. Renaud, "On user involvement in production of images used in visual authentication," *Journal of Visual Languages and Computing*, 2009.

[60] M. Oka, K. Kato, Y. Xu, L. Liang, and F. Wen, "Scribble-a-secret: Similarity-based password authentication using sketches," in *ICPR*, 2008.

[61] A. Gani, "A new algorithm on graphical user authentication (gua) based on multi-line grids," *Scientific Research and Essays*, 2010.

[62] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[63] M. Aharon, M. Elad, and A. M. Bruckstein, "On the uniqueness of overcomplete dictionaries, and a practical way to retrieve them," *Linear algebra and its applications*, vol. 416, no. 1, pp. 48–67, 2006.

[64] F. Couzinie-Devy, J. Mairal, F. Bach, and J. Ponce, "Dictionary learning for deblurring and digital zoom," *arXiv preprint arXiv:1110.0957*, 2011.

[65] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[66] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang, "Convolutional sparse coding for image super-resolution," *ICCV*, 2015.

[67] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

[68] F. Heide, L. Xiao, A. Kolb, M. B. Hullin, and W. Heidrich, "Imaging in scattering media using correlation image sensors and sparse convolutional coding," *Optics express*, 2014.

[69] T. Zhang, A. Bibi, and B. Ghanem, "In Defense of Sparse Tracking: Circulant Sparse Tracker," *CVPR*, 2016.

[70] Y. Zhu and S. Lucey, "Convolutional sparse coding for trajectory reconstruction," *PAMI*, 2015.

[71] B. Kong and C. C. Fowlkes, "Fast Convolutional Sparse Coding," *Tech. Rep. UCI*, 2014.

[72] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and Flexible Convolutional Sparse Coding," *CVPR*, 2015.

[73] W. Zhong and J. T.-Y. Kwok, "Fast stochastic alternating direction method of multipliers." in *ICML*, 2014, pp. 46–54.

[74] H. Bristow and S. Lucey, "Optimization Methods for Convolutional Sparse Coding," *arXiv Prepr. arXiv1406.2407v1*, 2014.

[75] M. Šorel and F. Šroubek, "Fast convolutional sparse coding using matrix inversion lemma," *Digital Signal Processing*, vol. 55, pp. 44–51, 2016.

[76] A. Tolba, A. El-Baz, and A. El-Harby, "Face recognition: A literature review." *International Journal of Signal Processing*, vol. 2, no. 2, 2006.

[77] P. Morasso, "Spatial control of arm movements," *Experimental brain research*, 1981.

[78] W. Abend, E. Bizzi, and P. Morasso, "Human arm trajectory formation." *Brain*, 1982.

[79] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *The journal of Neuroscience*, 1985.

[80] G. Taubin, "Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation," *PAMI*, 1991.

[81] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *ICCV*, 2003.

[82] Y.-L. Boureau, "Learning hierarchical feature extractors for image recognition," Ph.D. dissertation, New York University, 2014.

[83] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International journal of Remote sensing*, 2007.

[84] A. Jain and D. Zongker, "Feature selection: evaluation, application, and small sample performance," *PAMI*, 1997.

[85] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *PAMI*, 2009.

[86] W. Li, Z. Zhang, and Z. Liu, "Expandable data-driven graphical modeling of human actions based on salient postures," *Transactions on Circuits and Systems for Video Technology*, 2008.

[87] X. Wei, "Rational bezier representation of conics," 1991.

[88] G. Farin, "From conics to nurbs: A tutorial and survey," *IEEE Comput. Graph. Appl.*, vol. 12, no. 5, pp. 78–86, 1992.

[89] C. Xu, T.-w. Kim, and G. Farin, "The eccentricity of conic sections formulated as rational bézier quadratics," *Computer Aided Geometric Design*, vol. 27, no. 6, pp. 458–460, 2010.

[90] C. G. Broyden, "The convergence of a class of double-rank minimization algorithms 1. general considerations," *IMA Journal of Applied Mathematics*, vol. 6, no. 1, pp. 76–90, 1970.

[91] P. De Casteljau, "Mathématiques et cao. volume 2: formes à pôles," 1985.

[92] X. Hu, Y. Deng, X. Lin, J. Suo, Q. Dai, C. Barsi, and R. Raskar, "Robust and accurate transient light transport decomposition via convolutional sparse coding," *Optics letters*, vol. 39, no. 11, pp. 3177–3180, 2014.

[93] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[94] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and flexible convolutional sparse coding," in *CVPR*. IEEE, 2015, pp. 5135–5143.

[95] N. Wang, S. Zhang, X. Gao, J. Li, B. Song, and Z. Li, "Unified framework for face sketch synthesis," *Signal Processing*, vol. 130, pp. 1 – 11, 2017.

[96] B. K. Jang and R. T. Chin, "Analysis of thinning algorithms using mathematical morphology," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, 1990.

APPENDICES

A Appendix A - Awards

- KAUST Fellowship, KAUST, Saudi Arabia, 2011 - 2017
- KAUST Discovery Scholarship, KAUST, Saudi Arabia, 2008 - 2010