Free-hand Sketch Understanding and Analysis

Yi Li

Submitted to Queen Mary University of London in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Queen Mary University of London

2016

Abstract

With the proliferation of touch screens, sketching input has become popular among many software products. This phenomenon has stimulated a new round of boom in free-hand sketch research, covering topics like sketch recognition, sketch-based image retrieval, sketch synthesis and sketch segmentation. Comparing to previous sketch works, the newly proposed works are generally employing more complicated sketches and sketches in much larger quantity, thanks to the advancements in hardware. This thesis thus demonstrates some new works on free-hand sketches, presenting novel thoughts on aforementioned topics.

On sketch recognition, Eitz et al. [32] are the first explorers, who proposed the large-scale TU-Berlin sketch dataset [32] that made sketch recognition possible. Following their work, we continue to analyze the dataset and find that the visual cue sparsity and internal structural complexity are the two biggest challenges for sketch recognition. Accordingly, we propose multiple kernel learning [45] to fuse multiple visual cues and star graph representation [12] to encode the structures of the sketches. With the new schemes, we have achieved significant improvement on recognition accuracy (from 56% to 65.81%). Experimental study on sketch attributes is performed to further boost sketch recognition performance and enable novel retrieval-by-attribute applications.

For sketch-based image retrieval, we start by carefully examining the existing works. After looking at the big picture of sketch-based image retrieval, we highlight that studying the sketch's ability to distinguish intra-category object variations should be the most promising direction to proceed on, and we define it as the fine-grained sketch-based image retrieval problem. Deformable part-based model which addresses object part details and object deformations is raised to tackle this new problem, and graph matching is employed to compute the similarity between deformable part-based models by matching the parts of different models. To evaluate this new problem, we combine the TU-Berlin sketch dataset and the PASCAL VOC photo dataset [36] to form a new challenging cross-domain dataset with pairwise sketch-photo similarity ratings, and our proposed method has shown promising results on this new dataset.

Regarding sketch synthesis, we focus on the generating of real free-hand style sketches for general categories, as the closest previous work [8] only managed to show efficacy on a single category: human faces. The difficulties that impede sketch synthesis to reach other categories include the cluttered edges and diverse object variations due to deformation. To address those difficulties, we propose a deformable stroke model to form the sketch synthesis into a detection process, which is directly aiming at the cluttered background and the object variations. To alleviate the training of such a model, a perceptual grouping algorithm is further proposed that utilizes stroke length's relationship to stroke semantics, stroke temporal order and Gestalt principles [58] to perform part-level sketch segmentation. The perceptual grouping provides semantic part-level supervision automatically for the deformable stroke model training, and an iterative learning scheme is introduced to gradually refine the supervision and the model training. With the learned deformable stroke models, sketches with distinct free-hand style can be generated for many categories.

Declaration

I hereby declare that this thesis has been composed by myself and that it describes my own work. It has not been submitted, either in the same or different form, to this or any other university for a degree. All verbatim extracts are distinguished by quotation marks, and all sources of information have been acknowledged.

Some parts of the work have previously been published (or submitted) as:

- **Chapter 3**
  - Y. Li, Y. Song, S. Gong, Sketch Recognition by Ensemble Matching of Structured Features, In *British Machine Vision Conference*, pages 35.1–35.11, 2013.
  - Y. Li, T. M. Hospedales, Y. Song, S. Gong, Free-hand Sketch Recognition by Multi-Kernel Feature Learning, *Computer Vision and Image Understanding*, 137(C):1-11, 2015.
- **Chapter 4**
  - Y. Li, T. M. Hospedales, Y. Song, S. Gong, Fine-grained Sketch-based Image Retrieval by Matching Deformable Part Models, In *British Machine Vision Conference*, 2014
- **Chapter 5**
  - Y. Li, Y. Song, T. M. Hospedales, S. Gong, Free-hand Sketch Synthesis with Deformable Stroke Models , under major revision of *International Journal of Computer Vision*, 2015

Acknowledgements

My first gratitude will send to my supervisor Dr. Yi-Zhe Song. Studying for a PhD degree had always been my biggest dream before, and without this position created by him in Queen Mary University of London it could be hardly fulfilled. He has also offered countless guidance, concerns and encouragements to me in a lot of critical situations which help me to always carry on. Next, I would like to give great thanks to my co-supervisor Prof. Shaogang gong. His devoting attitude to work, philosophical points of view on academic value and the logical way to summarize and organize knowledge have largely influenced my way of researching. Also, I highly appreciate Dr. Timothy M. Hospedales for his detailed and tireless instructions on the technical parts of most of my works, and Dr. Ioannis Patras for being an objective independent assessor to offer encouraging and inspiring evaluation to each stage of my PhD study.

Besides aforementioned instructive mentors, I would also like to thank Tim Kay and Lukasz Zalewski from EECS system group, who have sent me selfless helps in many of my awkward scenarios caused by the restive computers and servers.

My PhD life in London is full of joy, and it could not happen without my dear colleagues and housemates. Special thanks to Xun(Alex) Xu, Guangwei Jiang, Zhiyuan(Patrick) Shi, Xiatian(Eddy) Zhu, Wenzhao Li, Peng Lin, Yun Zhou and Yongxin Yang for the time we shared and enjoyed together.

Finally and sincerely, great great thanks to my parents, for all the supports from the beginning to the end of my PhD study, for all the understanding of my life choices and for all the freedom of my right and wrong decisions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Free-hand sketches are an instinctive means of people to depict their perceptions of the objects and scenes existing in their daily life. This tradition commonly exists in informal discussions in everyday life and can be traced back to prehistoric times in the form of cave paintings. Therefore, as early as in the 1990s, sketches had already attracted research attentions from the computer vision community as the sketch-based image retrieval problem [37, 48, 60, 77]. But during a very long period, i.e. before 2010, most sketch datasets only employed very few sketches that mostly were just contour depictions due to the difficulty to collect sketches of the electronic format. But in the past several years, with the proliferation of the touchscreens, the sketches of the electronic format are getting more and more abundant, obtained either by styluses or just human fingers. This phenomenon has brought the study of the free-hand sketches to a whole new era whose symbol is the proposals of the datasets with more complicated sketches and more diverse categories [32–34, 53, 57, 94]. Among these datasets, the largest TU-Berlin dataset [32] has already included 250 categories and 80 sketches in each category, contrasting to fewer than a dozen sketches common in the previous works. A comparison of recently proposed TU-Berlin dataset to some previous datasets is shown in Figure 1.1. With these new datasets, our understanding of the sketches is profoundly deepened. Existing applications are further pushed to more realistic scenarios, and novel and interesting applications keep emerging. This thesis sits on the new sketch booming era, and starts from interpreting the existing works and datasets in order to obtain new understandings of them. Based on these new understandings, novel methods are raised to improve existing applications, and new applications are proposed as well to expand the free-

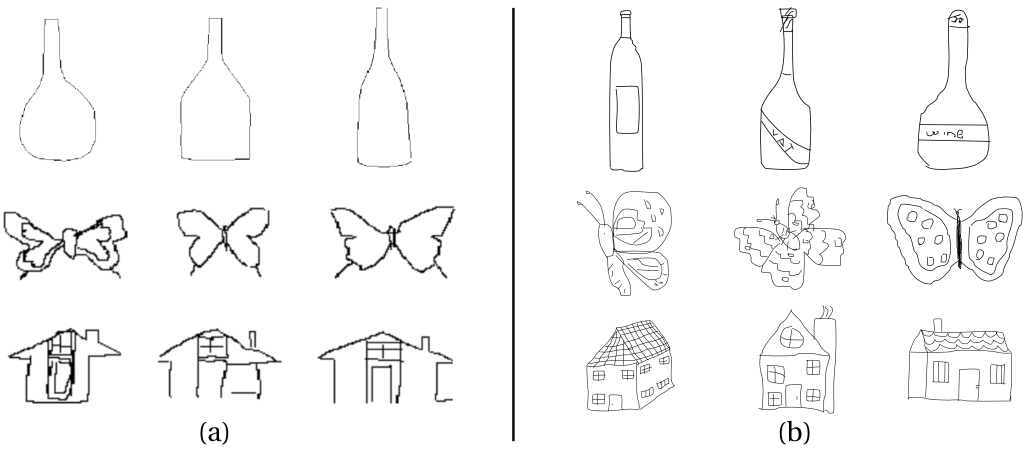


Figure 1.1: The representative sketches of (a) the previous sketch datasets [9, 20] and (b) the newly proposed TU-Berlin dataset [32].

hand sketch territory. The contents of this thesis are organized under three applications: sketch recognition, sketch-based image retrieval and sketch synthesis. .

1.1 Sketch Recognition

The first application we will look into is sketch recognition, as it is the novel application only made possible by the large-scale sketch datasets. Sketch recognition tries to identify the given sketch's category and can be used to establish some human-computer interaction systems based on sketches. This kind of sketch input system will benefit for the illiterate people and in the case that typing is not convenient. An demonstration of sketch recognition is available in Figure 1.4.

Eitz et al. [32] were the first to study both humans' and computers' sketch recognition abilities. They found that humans were good at interpreting sketches of different objects, but different people's sketches of the same object were quite different in terms of abstraction level and drawing style. More importantly, the sketches were iconic representations of the objects and thus were very different from images and 3D models. So to realize sketch recognition, a specific large-scale sketch dataset that could represent general humans' sketching skills was mandatory. They thus collected the large-scale TU-Berlin dataset which comprised sketches drawn by thousands of ordinary people and included enough variations on abstraction level and drawing style. They finally achieved good recognition accuracy (56%) with support vector machine (SVM) [2] classifiers trained on this dataset.

As their emphases were to realize sketch recognition and on various analyses, they did not give any in-depth thinking on the sketch recognition challenges for the computers. Following

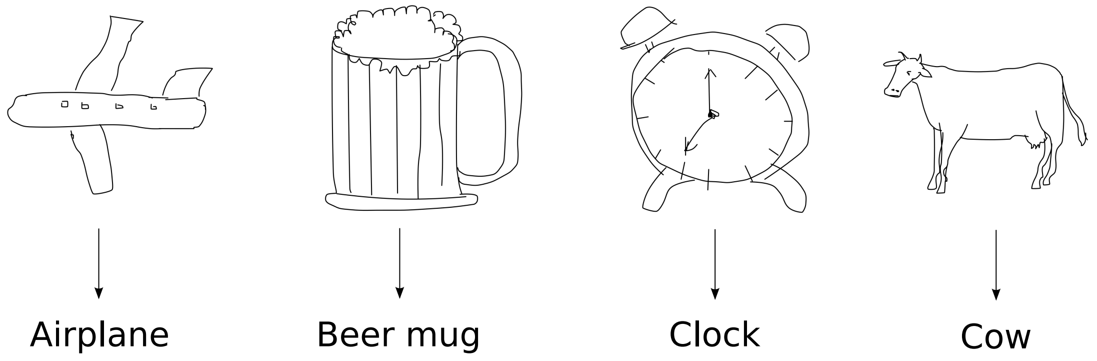


Figure 1.2: A demonstration of sketch recognition.

their lead, we look into the dataset with great caution and have found that the visual cue sparsity and internal structural complexity are the biggest challenges for sketch recognition. Acknowledging those challenges, multiple kernel learning [45] and star graph representation [12] are proposed accordingly to address them, which greatly increase the sketch recognition accuracy to 65.81%. Furthermore, we perform a preliminary study on how sketch attributes can benefit sketch recognition, and demonstrate some novel applications enabled by sketch attributes.

1.2 Sketch-based Image Retrieval

New features are also injected into the long-standing sketch-based image retrieval (SBIR) problem. SBIR aims to enhance text-based image retrieval by employing the shape information expressed by the query sketch. Some typical SBIR examples are illustrated in Figure 1.3.

As only simple sketches considered, the previous SBIR works [9, 20, 37, 48, 60, 72, 77, 83] are generally weak at addressing sketch details, and some works [9, 37, 72] only study single contours. Thus, they are not suitable to deal with more complicated sketches that come with the advancement of touchscreens. To address the complicated sketch details as well as the large scale, sophisticated edge matching and indexing methods [15, 80] and descriptor based methods [33, 34, 51] have been proposed.

However, we have further discovered that one advantageous feature of the sketches is generally ignored by the existing works but very promising to explore: their ability to distinguish intra-category object variations. And the definition of the object variations here goes beyond just holistic contour variations studied in [9] to articulated body part deformations and fine local details. Large intra-category sketch variations exist in the newly proposed TU-Berlin dataset, which provides us a handy condition to experiment on. We thus propose a fine-grained SBIR system

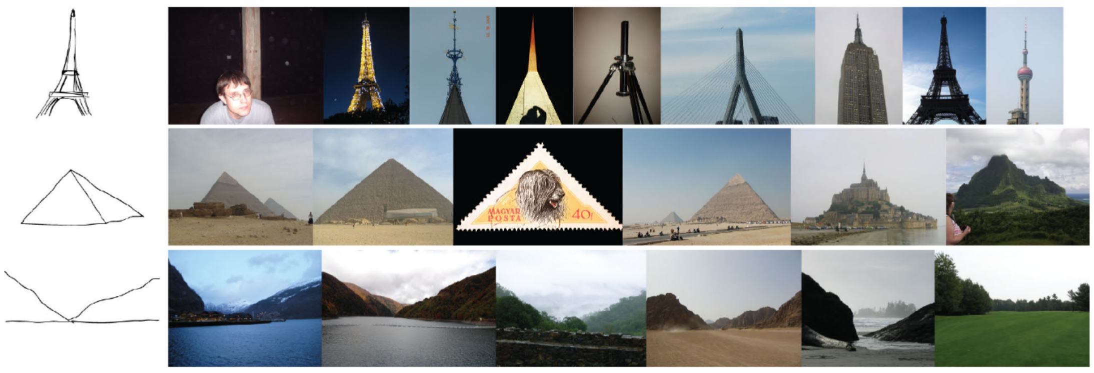


Figure 1.3: The sketch-based image retrieval examples [34].

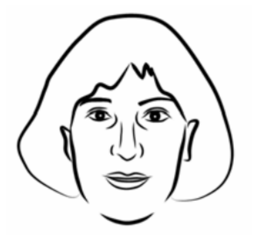
specially to solve this new problem. We employ deformable part-based model as the representation for both domains, counting on its ability to deal with part details and object deformations. Graph matching is used to compare the similarity of different deformable part-based models. Borrowing some sketches from the TU-Berlin dataset and some photos from the PASCAL VOC dataset [36], we propose a new challenging cross-domain dataset to evaluate our proposed framwork, and the results have shown promising future for SBIR towards commercial scenarios.

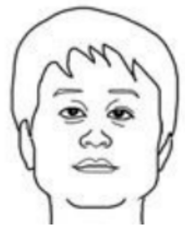
1.3 Sketch Synthesis

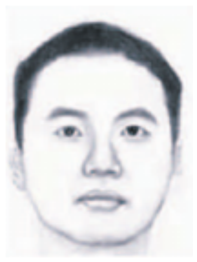
Sketch synthesis is another application being dramatically changed recently. Belonging to the non-photorealistic animation and rendering (NPAR) realm, sketch synthesis renders the photos into sketch-like drawing style while still keeps high similarity between the generated sketches and the original photos.

By utilizing a new source of information coming with the recently proposed sketch datasets – the sketch strokes (the stroke temporal order is also recorded), Berger et al. [8] are able to synthesize free-hand style sketches with natural stroke compositions for the first time. Previous methods [22, 46, 47, 66, 82, 101, 102, 104] have not considered using real human strokes as composing units, so make it easy for us to spot the artifacts in the synthesized sketches. The comparison of [8] and some previous works is shown in Figure 1.4. Although groundbreaking, their work is currently limited to one single category: human faces.

We manage to break the category boundary by studying the stroke data over again. The stroke data is moderately explored by several works [32, 42, 57, 85] previously. Although some insights and usages on stroke data and stroke temporal order are demonstrated, none of them is from a human perceptive point of view. Yet we specifically analyze the relationship between the stroke


(a)


(b)


(c)

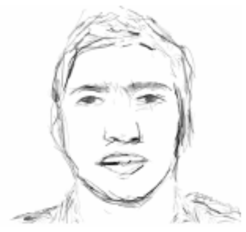

(d)

Figure 1.4: The synthesis results on human faces of (a) Chen et al. [22], (b) Liang et al. [66], (c) Wang and Tang [102] and (d) Berger et al. [8]. It can be clearly observed that (d) has an obviously distinct free-hand style: the sketch is composed of many sketchy strokes rather than a few bold and regulated strokes.

length and the stroke semantics (i.e. within what kind of length a stroke turns to denote an object part), and find out that there does exist a range for the length within which the strokes normally correspond to some object parts. We also find that a lot of consecutively drawn strokes are very likely depicting different portions of the same object part. Based on these observations and by synergizing Gestalt principles [58], we propose an unsupervised method to segment sketches into proper semantic parts. And built on these semantic parts, a generative deformable stroke model can be learned for a specific category which is further refined by iterative learning. Free-hand style sketches can be synthesized for the given images with the learned deformable stroke model and through a detection process at last.

1.4 Contributions

The contributions of this thesis are:

1) For sketch recognition:

We propose a star graph representation that captures both the holistic structure and local features to address the internal structural complexity problem. To further account for the lack of visual cue problem, we employ a multiple kernel learning framework that fuses several popular features known to work with sketches. Extensive experiments on the TU-Berlin dataset show significant improvement over the state-of-the-art, from 61.5% to 65.81% (human accuracy being 73.1%). Over and above that, we for the first time study attributes for sketches, and demonstrate their effectiveness in reducing confusion inside one super-category (a super-category is a set of categories having similar properties, like animal super-category including cat, dog, etc.). Moreover, we show how the high-level semantic nature of the attribute feature allows novel applications such as query by attribute or class-attribute description.

2) For sketch-based image retrieval:

We raise the fine-grained sketch-based image retrieval concept for the first time and define it to reflect the real-life scenario. A system based on deformable part-based models and graph matching is proposed to specifically tackle the newly proposed problem. Furthermore, a challenging real-life dataset with sketch-image pair similarity ratings is proposed to offer an effective benchmark for the fine-grained sketch-based image retrieval problem, on which the efficacy of our proposed system has been evidently proven.

3) For sketch synthesis:

We performed a comprehensive and empirical analysis of sketch stroke data, highlighting the relationship between stroke length and stroke semantics, as well as the reliability of the stroke temporal order. Based on the analysis, a perceptual grouping algorithm is proposed, which for the first time synergistically accounts for multiple cues, notably stroke length and stroke temporal order. By employing our perceptual grouping method, a deformable stroke model is automatically learned in an iterative process. This model encodes both the common topology and the variations in structure and appearance of a given sketch category. Afterwards a novel and general sketch synthesis application is derived from the learned sketch models.

1.5 Outline

The whole thesis is then organized as follows. **Chapter 2** offers a thorough literature review with respect to the three sketch applications. **Chapter 3** demonstrates our proposed sketch recognition method. **Chapter 4** introduces our fine-grained sketch-based image retrieval system. **Chapter 5** shows our sketch synthesis framework and finally **Chapter 6** concludes this thesis.

Chapter 2

Literature Review

This thesis overall looks into three major applications related to free-hand sketches: sketch recognition, sketch-based image retrieval and sketch segmentation. Therefore, this literature review chapter tries to cover as many materials to date as possible related to these three applications. Through the reviewing process, we aim to discover new understandings about the applications, and use them as the starting points to motivate our works. In terms of order, this chapter starts from the sketch recognition works, and then moves onto sketch-based image retrieval works and sketch synthesis works consecutively.

2.1 Sketch Recognition

As a newly proposed application, free-hand sketch recognition rose from the proposal of the large-scale TU-Berlin sketch dataset [32] and has experienced fast improvement during the last 3 years which our work took part in. In this section, we look through the literature of sketch recognition and the background of the techniques that we propose to improve sketch recognition.

The motivations of sketch recognition include: exploring general humans' sketching ability, testing humans' and computers' sketch recognition ability and making human-computer interaction systems with sketch input. The TU-Berlin dataset includes 250 categories with 80 sketches in each category, making it overall 20,000 sketches. This dataset is the largest sketch dataset to date and includes the most complicated and diverse sketches, in terms of number of category, sketch details, sketching style and abstraction level. With the proposal of this data, standard HOG feature [29], bag-of-words (BOW) representation [89] and support vector machine (SVM) classifiers [27] were used for sketch recognition in [32], and the published recognition rates for humans and computers were 73.1% and 56% respectively.

Later on, Schneider and Tuytelaars [85] boosted the sketch recognition performance further to 68.9% by employing Fisher Vectors [81]. By pruning similar categories that were hard to be distinguished by sketches and poor sketches that were too challenging for humans to recognize in TU-Berlin dataset, they proposed a pruned version of TU-Berlin dataset where human recognition rate was 100%. Very soon, Yu et al. [110], empowered with deep learning technique, boosted the sketch recognition performance to a new state-of-the-art 74.9%, that really 'beats human' on sketch recognition. However, that is partially due to the flaws existing in the TU-Berlin dataset and their performance on the new pruned TU-Berlin dataset will be interesting to see.

Right after Eitz et al. [32], but before Schneider and Tuytelaars [85], we also proposed to improve sketch recognition with star graph ensemble matching [12] and multiple kernel learning [79], and achieved a good performance of 65.81%. We also explored using sketch attributes to perform some novel applications, e.g., retrieval sketches with a specific attribute. The proposal of the star graph is to encode the structures of sketches. Thus we provide a review for relevant structured representations below, and explain their limitations in sketch encoding. Comprehensive reviews for multiple kernel learning and attribute learning are also presented.

2.1.1 Structured Feature Representation

The concept of spatially structured feature representation is not new in the computer vision community. Many applications such as category recognition [63] and landmark image retrieval [14] have already proposed the general concept of structured feature representation. Nevertheless, most of these structure encoding methods are quite specific to the problem domain they are designed to address, so are not directly applicable to the sketch encoding problem. Many of them are designed for the image domain and work with BOW representations. For example, the spatial pyramid matching method [63] employs a series of grids over the image with increasingly coarse level. Then the matches at each grid level are summed up with an attached weight to form the final similarity. It is designed for scene categorization and optimized for capturing frequently emerging local patterns in each scene category. However, this scheme is not effective for sketches, due to the large deformations and variations in highly abstract sketches resulting in weak structure information being captured by fixed-position cells. Another spatial BOW method [14] projects the 2D features onto certain lines or circles which are 1D space and then group the features by sectors in the 1D space. This concept of 1D encoding of local 2D features works well for landmark images where a dominant direction(s) may be readily obtained, but this property cannot be found generally in sketches. Only a few works have proposed structured representation of sketches, in which topological relationships between sketch parts were utilized for improving matching accuracy [41, 91]. However, these methods are strictly restricted to some simple CAD and clip-art drawings and are thus not directly applicable to human sketches. On the other hand, a standard star graph has an assigned center with each feature point represented as a node in the graph and connected to the center. A star graph model encodes both direction and distance of each feature point to the graph center, and therefore provides a richer and more flexible representation of spatial structures than other alternative schemes (see the experiments in Section 3.2).

2.1.2 Multiple Kernel Learning

Previous studies either focus on one feature, e.g. HOG [32], or selecting the best performing feature [65] for sketch recognition. However, this ignores the potentially complementary cues contained in other features. SVM multiple-kernel learning (MKL) [4, 45, 97, 99] ([45] offers a good review on MKL) provides a route to discriminative recognition that can exploit multiple complementary features. MKL achieves state-of-the-art performance in a variety of vision areas [45, 79], for example: winning the PASCAL VOC 2009 [35] object detection challenge by balancing dense and sparse textures and self-similarity; or color, shape and texture in recognizing flowers [79]. This is due to discriminatively learning how to weight features according to their informativeness. Moreover, they can automatically fuse multiple kernel metrics, which has also been a subject of comparative evaluation for sketches [53]. Recent MKL optimizers have improved computational efficiency [79], making them applicable for the large scale dataset [32] addressed here. We therefore in Chapter 3 go beyond existing work [32] and use MKL to discover not just the best single feature, but how each cue and kernel metric can be combined for best overall recognition performance.

2.1.3 Attribute Learning

Going beyond traditional structured and unstructured low-level features, attribute learning [62] has recently gained prominence in image [43, 62] and video [43, 68] recognition. Attributes aim to provide a powerful representation by computing a high-level semantic description of images.

For example, bears have fur and claws, while zebras have fur and stripes. Computing this representation involves a category-level annotation of attribute properties, and an additional step of supervised learning where classifiers are trained to predict each attribute, after which the vector classifier posteriors for each attribute becomes the new representation for an instance. This is effective because the resulting representation is low-dimensional and discriminative by design, as human designed attributes are exactly those which humans use to distinguish categories. In Chapter 3 we investigate for the first time the use of attributes for sketch understanding. Not only do attributes provide a novel representation with which sketch categories can be distinguished, but this representation is synergistic with low-level features [68]. Moreover the semantic nature of attribute representation will allow novel tasks that go beyond sketch recognition, such as attribute-query and ranking.

2.2 Sketch-based Image Retrieval

The researches on sketch-based image retrieval (SBIR) can be traced back to the early 1990s. While the basic concepts are generally consistent across the years, the contents are highly affected by the corresponding technologies of each period. As a result, we decide to organize those works in a chronological way (i.e. divided into 3 eras). To be noticed, we do not intend to exhaust all the ever existing SBIR works, but to summarize those works that can represent the trend in each era. In the end, we present critical thoughts for previous SBIR works from multiple perspectives and use them to motivate a new fine-grained sketch-based image retrieval problem. The relevant techniques that we propose to address the fine-grained SBIR are also referred in this section.

2.2.1 A History of Sketch-based Image Retrieval

The 1**st era** came between 1992-1994, when the concept of content-based image retrieval was first raised by Hirata and Kato [48] and Kato et al. [60], in which sketch-based image retrieval was especially focused on. The symbols of the 1st era are the establishment of the basic concepts of the SBIR framework and the using of rigid or coarse similarity metrics. In [48,60], to compare the similarity between a sketch and an image edge map, they firstly divided the sketch and the image into blocks and then compared the block-to-block pixel correlations. The blocks of the sketch were shifted within a small range during the comparison to address small distortions, but the overall corresponding blocks based metric was still quite rigid. Right after this thread of works, the QBIC (Query By Image Content) project [37,77] was published by IBM research. As another early and important work in content-based image retrieval, the QBIC project comprised SBIR, yet went beyond by also emphasizing color and texture. It used global shape features including area, circularity, eccentricity, major axis orientation and moment invariants to encode the shapes in the sketches and the images rather than just raw pixels. The similarity comparison of QBIC on shapes was performed with weighted Euclidean distance on the constructed global shape features, and thus was quite coarse and cannot address the object details. It manually or semi-automatically circled out the objects to facilitate more accurate object-oriented retrieval. And in indexing, it focused on dimension reduction to enable more efficient retrieval and less storage occupation for some standard indexing schemes, e.g., R-trees, grid files, etc.

The 2nd era was between 1994-2000, and the symbols are the considerations of distortion and abstraction for the sketches, and to achieve invariances in several aspects. Bimbo et al. [9–11] employed elastic matching to allow the query sketch to deform according to the given image object, and used both the deformation energy and the matching extent between the deformed sketch and the image to measure the similarity. The elastic matching was expected to approximate human visual perception and be robust to distortion. Multi-object query was also considered by using a signature file to encode the rough spatial relationships between the objects, and individual objects' shapes would be compared only if the signature files of the sketch and the image had matched. In the single object scenario, their method achieved invariances to translation and scaling by cropping out the image object manually and normalizing the sketch and image objects with similar aspect ratios to the same size. Specially, to the best of our knowledge, in [9], Bimbo and Pala for the first time quantitatively evaluated sketches' ability to rank similar images using ranked image list that was constructed from human rated sketch-image pairs. While Bimbo et al. [9–11] utilized edge maps directly obtained from an edge detector, e.g., Canny edge detector, Rajendran and Chang [83] and Chans et al. [20] thought that the edge maps needed to be processed to react to the special characteristics of human sketches. More specifically, Rajendran and Chang [83] employed a multi-scale representation for images to address the variations of level-of-detail of human sketches. Accompanying the multi-scale representation, curvature-directions histograms were used as features to achieve invariances to translation, scaling and rotation. On the other hand, Chans et al. [20] extracted the prominent edges of the images by employing a curvelet model to evolve and encode the image edges, as they believed that the users tended to ignore details when drawing the sketches. Their method was able to cope with global and local translations to some extent and had the similar potential to cope with scaling and rotation. Aligning with Rajendran and Chang [83], Matusiak et al. [72] also tried to achieve total invariances to translation, scaling and rotation by employing the Curvature Scale Space (CSS) representation, which was inherently invariant to translation, scaling and rotation. However, for [72,83] to work, an assumption that the object of the image could be clearly segmented was either implicitly or explicitly made, thus rendering those methods only effective to simple datasets.

The 3rd era started from 2009 and lasts heretofore, and the most significant symbol is working towards large-scale datasets as assumed to be in favor of real-life applications. As a continuous trend of the 2nd era, contour based matching was adopted by Cao et al. [15] and Parui et al. [80]. However, there is an obstacle for contour based methods to scale up to large datasets: the computational cost to match the contour representations while addressing the deformations in the meantime. To increase the computational efficiency, the system's expressive power on the sketch-image similarity has to be compromised. Cao et al. [15] employed Oriented Chamfer Matching as the contour matching method. An indexing scheme that organized images by the individual edge pixels shared by images, was proposed to achieve efficient retrieval. However, as the rigidity of Oriented Chamfer Matching and the normalization needed for the indexing, the system can retrieval only objects at the exact position as the sketch, and only very small local deformations are allowed. Parui et al. [80] opted for another extreme. In their work, only the salient contours (chains) were extracted, and each chain was represented as straight line-like segments and further encoded by the length ratios and the joint angles of the adjacent segments. Finally, a fast Dynamic Programming-based approximate substring matching algorithm was used to match two sets of contours and a hierarchical *k*-medoids based indexing was used to index the images by its contours. Although this representation is invariant to translation, scaling and rotation and facilitates fast retrieval, it has lost dramatically the originally smooth and detailed shape properties of both the sketches and the images. Therefore, the retrieved results generally have rather low visual similarity to the query sketch in terms of shape, which is exactly the biggest merit of SBIR. Concerning about the possible drawbacks of contour based methods, Eitz et al. [33,34] and Hu et al. [51,53,56] have explored popular feature descriptors to scale up SBIR, counting on their convenience for large-scale indexing. In [33], Eitz et al. evaluated histograms of oriented gradients (HOG) [29] and tensor descriptor [61] against angular radial partitioning (ARP) [17] and edge histogram descriptor (EHD) [108] (including a semiglobal variant version), all in a global feature fashion. They also evaluated 3 grid resolution variants for feature sampling, and a masked and an unmasked version for each grid resolution. A *k*-means tree [44] and a best-bin-first strategy [76] were employed for indexing the descriptors. The results concluded that as global features, HOG and tensor significantly outperform ARP and EHD, with tensor slightly better than HOG. Right after, by going beyond the global features and including several popular local features, Eitz et al. [34] proposed a benchmark for SBIR. The local features included shape context (SC) [7], spark feature [34] and HOG (as local feature), and bag-of-words (BOW) representation [89] was used to encode the features. For indexing, the standard inverted index that indexes images by the visual words [113] was directly employed. In their comparison, although the local feature SHOG (HOG computed on edge maps) obtained the best performance, the other local features were outperformed by the global features. Worth noticing, in this work, the sketch's power to discriminate similar images (from both the same category and other categories) is quantitatively evaluated again after Bimbo and Pala [9]. Human ranked image lists that were based on sets of sketch-image pairs manually rated according to visual similarities, were used for retrieval evaluation. In the meantime, Hu et al. [51] was doing similar evaluations for local features and BOW representation, yet with a slightly different set of features. They proposed to compute gradient field images for the edge maps in advance of computing HOG features. The gradient field is composed of interpolations of edge pixels, so physically, it expands the width of the prominent edges and increases the generality of the representation. In a later journal version [53], Hu and Collomosse increased their feature set which finally included gradient field HOG (GF-HOG), HOG, scale-invariant feature transform (SIFT) [70, 71], self-similarity descriptor (SSIM) [86], SC and tensor descriptor (global feature). Their results confirmed that local features, especially GF-HOG and HOG were obviously better than global features in SBIR. However, their evaluation is category level retrieval, which does not take care of the perceptual similarity between the sketch and the image, so it provides relatively limited insight on the similarity discrimination power of different features. *k*-d tree was mentioned in their work for indexing, yet the *k*-d tree excluded some distance metrics, like histogram intersection and chi square, and thus should be used accordingly. Continuing the local feature evaluation, Hu et al. [56] considered the object localization problem as an exceptional work in the 3rd era. A hierarchical segmentation algorithm [3] was used to obtain the regions of the image, and the object's contour was contained in some of those regions. Afterwards, each region was encoded by local features for retrieval.

2.2.2 Sketch-based Video Retrieval

Besides sketch-based image retrieval, there are also a considerable amount of works on sketch-based video retrieval (SBVR). The early works [6,49,87,103] used sketched object trajectories to retrieve video clips containing specific motion patterns. Since the only focus is on the trajectory, the appearance and semantic properties of the videos are entirely ignored by these methods. VideoQ [19] was one of the first works to combine motion cues with appearance cues, but its region segmentation relied majorly on color and thus was quite preliminary. Instead, Collomosse et al. [26] proposed to aggregate super-pixels to sketched objects and did not assume 'ideal' segmentation of the video. Their method has significantly improved the precision yet at the expense of a computationally costly inference procedure which assigns the super-pixels to the sketch objects. Therefore, Hu et al. [52] tried an alternative approach to match tokenized motion trajectories with a trellis-based edit distance. And Hu et al. [54] futher considered to integrate the semantic labels on both the sketch side and the video side, in order to enhance the system's ability to interpret the semantics of the video content. A more robust motion clustering algorithm was employed in [54] as well to improve the retrieval performance. Given all the works above, the segmentation procedures for the video frames were performed offline as a pre-processing step. But uniquely, Hu et al. [55] performed segmentation at query-time with a Markov Random Field (MRF) optimization which could also rank the relevant clips and localize the sketched objects.

2.2.3 Deformable Part-based Model

To encode the sketch appearance with local features as well as addressing abstractions/distortions, we employ deformable part-based model (DPM) as the representation, as it is a two-layer structure encoding both holistic shape and local parts. The DPM [38] is designed for object detection and obtains state-of-the-art performance on the challenging PASCAL VOC dataset [36]. Yang and Ramanan [109] and Sun and Savarese [93] have used strongly supervised DPM for human pose estimation. However, their methods need a pre-defined pose model for each specific category and extensive part annotations are mandatory, which make them non-scalable in the general case for numerous diverse categories. Therefore, we adopt the original DPM [38] to encode the objects in two domains. To bridge the DPMs from different domains, we further employ an effective graph matching method to measure the cross-domain similarity of DPMs.

2.2.4 Graph Matching

Graph matching is widely used in computer vision applications such as object categorization [31], face recognition [106] and tracking [90]. Graph matching has the advantage of flexibly encoding topological object structure, and coping with relatively large structural deformations. There has been a great body of research to date on graph matching. Cho et al. [24] establish matches by performing random graph walk on an association graph whose nodes represent candidate matches, which is later extended to cope with node progression by iteratively examining homography projection errors [25]. Very recently, supervised learning techniques have also shown prominence towards graph matching [23, 50]. Despite offering state-of-the-art results on standard datasets, they require explicit training a priori. Therefore, we employ Cho et al. [24] to match DPM parts.

2.3 Sketch Synthesis

In this section, we review the sketch synthesis, sketch segmentation and stroke analysis literature. The stroke analysis is brought up as we are looking at sketch synthesis with stroke data, and the sketch segmentation serves as a first step for our general sketch synthesis framework.

The non-photorealistic animation and rendering (NPAR) field has intensively studied sketch synthesis, yet majorly focuses on mimicking the sketchy feeling for the lines or the textures. Humans' behaviors (especially amateur drawers') of sketching the central object without irrelevant background and imposing high-level abstraction on the objects are seldom modeled by them. High quality inputs are often desired by these methods as well. Winkenbach and Salesin [104] introduced "stroke textures" to simulate both texture and tone of line drawing and demonstrated quite realistic architectural drawings with sophisticated details. A highly descriptive interactive system was established based on the stroke textures by Salisbury et al. [84], which could greatly alleviate human labor on sketching detailed textures. AlMeraj et al. [1] employed a physically-based mathematical model to render input point data directly into lines that resembled human-drawn style. Their user studies had supported that their generated lines were perceptually indistinguishable from human-drawn lines. Gooch et al. [46] considered human facial illustration and created the sketchy illustrations with a human brightness perception model. They also considered holistic abstraction by converting the sketchy illustrations into caricatures yet with an interactive technique. Another interesting work presented by Winnemöller et al. [105] carefully reviewed the difference-of-Gaussians (DoG) operator and its extensions, and their connections to other image processing techniques. Especially, they demonstrated many new results on a variety of styles just using the extended DoG (XDoG) formulation, including pencil-shading, pastel, hatching and woodcut. Kang et al. also proposed an important extension for DoG [59]. They computed edge tangent flow (ETF) to offer edge direction guidance for the DoG filtering (originally computed isotropically) and named it FDoG.

Other attempts convert images to sketch-like edge maps, which despite being more abstract still closely resemble natural image statistics ([47, 82]). And sketch tokens [112] is a recently proposed contour detection work which utilized human traced contours in images to learn mid-level stroke-like representations. They have achieved very competitive contour detection performance with considerably low computational cost. Data-driven approaches have been introduced to generate more human-like sketches, exclusively for one object category: human faces. [22, 66] took simple exemplar-based approachs to synthesize faces and used holistic training sketches. [101, 102] decompose training image-sketch pairs into patches, and train a patch-level mapping model. All face synthesis systems above work with professional sketches and assume perfect alignment across all training and testing data. As a result, image and patch-level replacement strategies are often sufficient to synthesize sketches.

Moving onto free-hand sketches, [8] directly use strokes of a portrait sketch dataset collected from professional artists, and learn a set of parameters that reflect style and abstraction of different artists. They achieved this by building artist-specific stroke libraries and performing stroke-level study on them with multiple characteristics accounted. Upon synthesis, they first convert image edges into vector curves according to a chosen style, then replace them with human strokes measuring shape, curvature and length. Although these stroke-level operations provided more freedom during synthesis, the assumption of rigorous alignment, in the form of manually fitting a face-specific mesh model to both images and sketches, is still made making extension to wider categories non-trivial. Their work laid a solid foundation for future study on free-hand sketch synthesis, yet extending it to many categories presents three major challenges: (i) sketches with fully annotated parts/feature points are difficult and costly to acquire, especially for more than one category; (ii) intra-category appearance and structure variations are larger in categories other than faces, and (iii) a better means of model fitting is required to account for noisier edges. Yet the model proposed in Chapter 5 is flexible enough to account for all these highlighted problems.

2.3.1 Sketch Segmentation

Sketch segmentation was originally proposed to enhance SBIR by extracting clear sketch object(s) from cluttered sketches or segmenting semantic sketch components out for part-level retrieval. But in the context of this thesis, we employ it to discover semantic parts of a category and thus facilitate sketch synthesis. As still a newly emerged topic, the literature keeps coarse. Sun et al. [94] studied the object-level sketch segmentation using basic perceptual principles, e.g., proximity, similarity, symmetry, direction and closure, and a web-scale clipart database to check the semantic meaningfulness of segmented objects. Huang et al. [57] remains the single study on stroke-level segmentation on free-hand sketches to date. They worked with sketches of 3D objects, assuming that sketches do not possess noise or over-sketching (obvious overlapping strokes). Annotated 3D models are mandatory as guidance, and the specific perspective that suits a query sketch needs to be manually selected. These limitations make their method hard to use in practice. In this thesis, we look at stroke-level/part-level free-hand sketch segmentation when noise and over-sketching are pervasive and needs minimal guidance (the proper stroke length). And a novel perceptual grouping algorithm is proposed to address this.

2.3.2 Stroke Analysis

Similar as the status of sketch segmentation, stroke-level analysis of human sketches remains sparse. Existing studies ([8, 32, 85]) have mentioned stroke ordering, categorizing strokes into types, and the importance of individual strokes for recognition. However, a detailed analysis has been lacking especially towards: (i) level of semantics encoded by human strokes, and (ii) the temporal sequencing of strokes within a given category.

[32] proposed a dataset of 20,000 human sketches and offered anecdotal evidence towards the role of stroke ordering. [42] claimed that human generally sketch in a hierarchical fashion, i.e., contours first, details second. Yet as can be seen later in Chapter 5, we found this does not always hold, especially for non-expert sketches. More recently, [85] touched on stroke importance and demonstrated empirically that certain strokes are more important for sketch recognition.

While interesting, none of the work above provided means of modeling stroke ordering/saliency inside a computational framework, thus making potential applications unclear. [57] was first in actually utilizing temporal ordering of strokes as a soft grouping constraint. Similar to them, we also employ stroke ordering as a cost term in our grouping framework. Yet while they only took the temporal order grouping cue as a hypothesis, we move on to provide solid evidence to support this usage in Chapter 5.

A more comprehensive analysis of strokes was performed by [8] aiming to decode the style and abstraction of different artists. They claimed that stroke length correlates positively with abstraction level, and in turn categorized strokes into several types based on their geometrical characteristics. Although insightful, their analysis was constrained to a dataset of professional portrait sketches, whereas in Chapter 5 we perform an in-depth study into non-expert sketches of many categories as well as the professional portrait dataset and we specifically aim to understand stroke semantics rather than style and abstraction.

2.3.3 Contour Models and Pictorial Structure Analysis

Our deformable stroke model that will be introduced in Chapter 5 is inspired by contour ([28, 40, 78, 88]) and pictorial structure models ([39]). Both have been shown to work well in the image domain, especially in terms of addressing holistic structural variation and noise robustness. The idea behind contour models is learning object parts directly on edge fragments. And a by-product of the contour model is that via detection an instance of the model will be left on the input image. Despite being able to generate sketch-like instances of the model, the main focus of that work is on object detection, therefore synthesized results do not exhibit sufficient aesthetic quality. Major drawbacks of contour models in the context of sketch synthesis are: (i) duplicated parts and missing details as a result of unsupervised learning, (ii) rigid star-graph structure and relatively weak detector are not good at modeling sophisticated topology and enforcing plausible sketch geometry, and (iii) inability to address appearance variations associated with local contour fragments. On the other hand, pictorial structure models are very efficient at explicitly and accurately modeling all mandatory parts and their spatial relationships. They work by using a minimum spanning tree and casting model learning and detection into a statistical maximum a posteriori (MAP) framework. Yet the much desired model accuracy is achieved at the cost of supervised learning that involves intensive labeling a priori.

In Chapter 5, by integrating pictorial structure and contour models, we propose a deformable stroke model that: (i) employs our proposed sketch segmentation algorithm and an iterative learning scheme, and thus yields accurate models with minimum human effort, (ii) customizes the model learning and detection framework of pictorial structure to address more sophisticated topology possessed by sketches and achieve more effective stroke to edge map registration, and (iii) augments contour model parts from just one uniform contour fragment to multiple stroke exemplars in order to capture local appearance variations.

The And-Or graph is a hierarchical-compositional model which has been widely applied for sketch modeling. An And-node indicates a decomposition of a configuration or sub-configuration by its children, while an Or-node serves as a switch among alternative sub-configurations. Both the part appearance and structure variations can be encoded in the And-Or graph. Wu et al. [107] proposed an active basis model, which employed the And-Or graph and could be applied to general categories. The active basis model consists of a set of Gabor wavelet elements which look like short strokes and can slightly perturb their locations and orientations to form different object variations. A shared sketch algorithm and a computational architecture of sum-max maps were employed for model learning and model recognition respectively. Our model in essence is also an And-Or graph with an And-node consisting the parts and Or-nodes encoding stroke exemplars. Our model learning and detection share resemblance to the above work but dramatically differ in that we learn our model from processed real human strokes and do not ask for any part-level supervision.

2.4 Summary

2.4.1 Sketch recogntion

By observing the TU-Berlin sketch dataset [32], we consider the complexity of internal structures as opposed to simple shapes (demonstrated later in Figure 3.1) and obvious visual cue sparsity as opposed to images are the two biggest challenges for the general free-hand sketches. Moreover, sketch attributes could be an interesting exploring direction towards novel applications.

In Chapter 3, we will employ star graph representation and multiple kernel learning to tackle these two big challenges. Thorough evaluations on different features, representations and kernel metrics are also performed. And interesting sketch retrieval by attribute application is presented to demonstrate the efficacy of sketch attributes.

2.4.2 Sketch-based image retrieval

After looking through the 3 past eras, we draw some conclusions for SBIR as follows:

i) The applying scenarios: Two scenarios are possible for SBIR: 1) sketching the scene and 2) sketching the object. In the first scenario, the user is delivering a holistic description of the image (usually coarse), a holistic matching scheme, e.g., global feature and edgel index [15], is sufficient for the purpose. Therefore, it may not be the focus of SBIR research. In the second scenario, focusing on objects, more details and more complex structures could be expected in the sketches. Moreover, when the objects are not the only things in the image, object localization could be helpful for reliable retrieval. The most foreseeable real-life application for SBIR is commodity retrieval, like retrieving clothes and furniture. Therefore, the second scenario, especially towards intra-category fine-grained SBIR, is a more promising and worthwhile direction.

ii) The representations: Two types of representations are existing in the SBIR literature. Contour based representation, due to its limited descriptive power and high computational cost to compare, is only effective when the sketches are generally simple and/or the gallery is not large. But its ability to address abstraction and distortion is profound especially when an elastic matching scheme is employed. On the contrary, descriptor based representation, especially local feature based representation, can capture rich details and is easy to scale up. But its ability to address abstraction and distortion is limited due to the hand-crafted and rigid design. Some descriptors are designed to address distortion, like SC and SSIM, but they did not perform well in the SBIR evaluations, comparing to more rigid descriptor like HOG. The possible explanation is that there is a trade-off between the ability to address abstraction and distortion and the ability to be discriminative. And for large-scale SBIR, the ability to be discriminative seems more important. This is however intuitive, because if it goes to higher abstraction level, many things will look alike. Above all, local feature based representation is more suitable for future SBIR, yet an extra scheme to address the unrealistic abstractions/distortions due to unlimited drawing styles is still desirable.

iii) The dataset and the evaluation metric: As concluded above to proceed to intra-category fine-grained SBIR, a dataset that can evaluate the intra-category fine details and with multiple categories and moderate size, is necessary to facilitate future research. The human ranked image list evaluation strategy [9, 33] should be adopted for this dataset, as it can accurately evaluate the fine-grained discrimination power of the approaches.

In Chapter 4, we raise up the fine-grained sketch-based image retrieval concept for the first time and employ deformable part-based model and graph matching to solve it.

2.4.3 Sketch synthesis

Free-hand sketch synthesis that resembles human habits of using sketchy strokes, abstracting the objects and tending to ignore the background, is still challenging to apply on general categories other than human faces. We argue that a detection model learned on sketch strokes and accounting object localization and object deformation variations could possibly address these issues and simulate the human sketching behavior well.

In Chapter 5, a deformable stroke model which is inspired by the pictorial structure and the contour models is learned and utilized for the general category free-hand sketch synthesis. The model supervision cost is alleviated by a novel sketch segmentation method called perceptual grouping, and this segmentation method is benefited from a thorough stroke analysis on stroke semantics and stroke temporal order.

Chapter 3

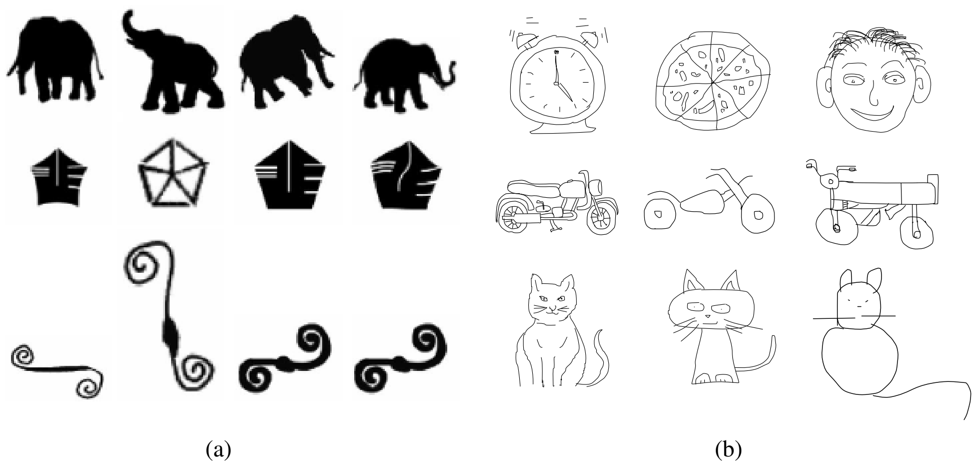
Sketch Recognition

Given the design of the TU-Berlin sketch dataset [32], we consider that the two biggest challenges for the general free-hand sketches are the complexity of internal structures as opposed to simple shapes (see Figure 3.1) and obvious visual cue sparsity as opposed to images.

To address the complexity of internal structures, we propose a mid-level representation to capture the holistic structure of sketches. More specifically, we employ a star graph to encode both local features and holistic structure of a sketch and exploit ensemble matching as a similarity measure. With the star graph, popular local features are carefully evaluated and compared for sketch recognition, and star graph has the best performance.

Furthermore, although different features or representations have different levels of performance, we argue that all features contain some potentially complementary information, at least for some classes, and should ideally be used together. We therefore address the cue sparsity problem via Multiple Kernel Learning (MKL), aiming at fully utilizing the discriminative power of all features and eliminating the both bias imposed by any single feature. Our experiment confirms much better performance of MKL on sketch recognition over Eitz et al. [32], and different kernel metrics are fairly compared.

The TU-Berlin dataset has as many as 250 categories. In order to show how different representations benefit certain categories more clearly, we introduce the concept of super-categories, which is defined as a superset of basic categories that share a higher-level semantic property (e.g., animal, plant). We found that although the star graph is generally best, different representations tend to favor different super-categories. An interesting finding from the super-category analysis is



(a) (b)

Figure 3.1: (a) Typical inputs of shape matching are generally silhouettes with quite simple internal structures, and are quite photo-realistic (extracted from [7]): three rows each corresponding to one object category; (b) Free-hand sketches are likely to have more complicated internal structures. Some sketches can exhibit similar silhouettes, but their internal structures can be quite different, e.g., the alarm clock, the pizza, and the face shown in the first row. In addition, the abstraction/distortion level and the diversity of drawing styles of free-hand sketches can both be higher than shapes.

that the confusions inside super-categories are much bigger than those between super-categories. This is especially true for large super-categories such as animal and vehicle. It is hypothesized that higher-level semantic properties shared among categories (e.g., spots on the body of a giraffe or butterfly) could help to remove ambiguity within a super-category – a hypothesis that was found to be successful in the image domain [62, 64]. Inspired by Lampert et al. [62], this work performs a preliminary study on how sketch attributes can benefit sketch recognition by constructing an attribute kernel within the MKL framework. The experiment is carried out on the animal super-category with classic animal attributes from [62] as well as additional attributes obtained from WordNet [74]. Experimental results show attributes to be effective in improving recognition performance inside super-categories.

Finally, going beyond simple recognition of sketch categories, we show how the high-level semantic nature of attribute features can be used to enable novel applications. We demonstrate attribute-based sketch retrieval (query by description rather than category; e.g., stripy), and joint category-attribute sketch retrieval (find a long-leg ant, etc).

3.1 Methodology

This section introduces the features, representations and classification models utilized for our sketch recognition study.

3.1.1 Features

A. ***Histogram of Oriented Gradients (HOG)***

HOG was first proposed by [29] for pedestrian detection. The gradients in each cell on a dense uniform grid are quantized into orientation bins that are then formed into a histogram. This feature is commonly reported to have best performance with sketches [32–34, 51].

B. ***Self-Similarity (SSIM)***

SSIM was proposed by [86]. For a feature point p, it compares a patch centered at p to nearby patches within a local region also centered at p, thus extract the "local self-similarity" for p. Then, the local SSIM descriptors are formed into a star graph model, and ensemble matching is employed to match the star graph models. SSIM has already been used on some very simple colored sketches [86], thus it is worthwhile to evaluate it with human sketches.

C. ***Daisy***

Daisy is based on histograms of gradients, like SIFT and GLOH [73], but utilizes a Gaussian weighting and circularly symmetrical kernel. It is very fast and efficient to compute densely [96]. Recent work of sketch tokens [67] has shown its effectiveness with sketches.

D. ***Attributes***

Unlike the previous features, the high-level attribute representation is itself the output of a supervised learning procedure. Attribute ground truth is defined by a binary class-attribute association matrix $\mathbf{A}$ (Figure 3.6), where each column specifies the attributes for that class. Given this matrix, a bank of M binary SVM attribute classifiers are independently trained to predict the presence or absence of each attribute. That is, for each attribute m, sketches from all categories with $a_m = 1$ are positive and sketches from categories with $a_m = 0$ are negative. The posterior $p(a_m|\mathbf{x})$ then reports the probability of a given sketch $\mathbf{x}$ having attribute m. The attribute representation of a sketch is then the M dimensional vector stacking the posterior probabilities for the presence of each attribute $A(\mathbf{x}) = [p(a_1|\mathbf{x}), \ldots, p(a_M|\mathbf{x})]$. Rather than utilizing these posteriors directly to predict the category as in [62], we use $A(\mathbf{x})$ as a new representation to be combined with the previous features by MKL. Details of SVM and MKL are described in Section 3.1.3.

3.1.2 Representations

A. ***Bag-of-words Representation***

Bag-of-words BOW is used as the baseline for sketch recognition and multiple features are employed to evaluate its performance, including HOG, SSIM and Daisy. We apply normalization to all the sketches by scaling them into a fixed size. The features of a sketch are extracted on local patches. And the patches are centered in the intersections of a regular grid on top of the sketch. We use relatively large-sized patches, due to the limited information contained by the sketch, thus those patches have overlapping areas. To construct the BOW, we first collect a large set of n features by random sampling. Those n features are clustered into H clusters via k-means. The mean values of the clusters are used to form a visual codebook : $\mathcal{U} = \{\mathbf{u}_i\}_{i=1}^{k}$. After the codebook is obtained, a feature $\mathbf{f}$ is then represented by a vector of the weights of $\mathbf{f}$ to all the words $\mathbf{u}_i$. The weight is the distance between the $\mathbf{f}$ and a word $\mathbf{u}_i$ measured by Gaussian kernel, with parameter σ.

B. ***Star Graph and Ensemble Matching***

For the star graph representation, we apply the same normalization and grid as for BOW. The nodes of the graph are grid intersections close to the sketch strokes, so they can depict the structure of the sketch and different sketches have different numbers of nodes. In practice, we choose the grid intersections that have valid SSIM features, as they are just the intersections close to the strokes. Those grid intersections are applied to other features afterwards. The center of the star graph is the center of mass of those nodes.

We denote star graph as $G = (V,E,A)$, where V,E,A represent the graph nodes, edges and properties respectively. More specifically, $V = \{v_i\}_{i=1}^{N_s} \cup c$ denotes all N_s sample points $\{v_i\}_{i=1}^{N_s}$ and the graph center c, while $e_i \in E$ is the edge between v_i and c. Besides, $\mathbf{a}_{ic} \in A$ is a relative location vector describing e_i, and $\mathbf{a}_i \in A$ denotes the feature descriptor corresponding to node v_i.

Ensemble Matching is the similarity metric employed here to compare star graphs. Like in Boiman and Irani [12], we assume that given a dataset ensemble patch with a descriptor $\mathbf{a}_i^t$ and a relative location $\mathbf{a}_{ic}^t$, the corresponding query patch descriptor $\mathbf{a}_i^q$ is independent to other query patch descriptors and the corresponding query patch relative location $\mathbf{a}_{ic}^q$ is independent to other query patch locations. The assumption on descriptor independence although is not valid in case of overlapping patches, it is still a useful approximation [12]. And the assumption on location independence offers the flexibility to accommodate small variations on translation, scale and rotation. We also adopt [12]'s strategy to model the relation between a patch descriptor $\mathbf{a}_i^t$ and its location $\mathbf{a}_{ic}^t$ from the dataset ensemble non-parametrically using examples from the dataset to achieve generality:

$$P(\mathbf{a}_i^t|\mathbf{a}_{jc}^t) = \begin{cases} 1, & \text{i=j, i.e. this pair exists in the dataset.} \\ 0, & \text{otherwise.} \end{cases}.$$

We then formulate the similarity between two star graphs q (query) and t (target) as below:

$$P(G^q|G^t) = \prod_i \Big(P(\mathbf{a}_i^q|\mathbf{a}_i^t)P(\mathbf{a}_{ic}^q|\mathbf{a}_{ic}^t)P(\mathbf{a}_i^t|\mathbf{a}_{ic}^t)\Big) = \prod_i \Big(P(\mathbf{a}_i^q|\mathbf{a}_i^t)P(\mathbf{a}_{ic}^q|\mathbf{a}_{ic}^t))\Big). \tag{3.1}$$

where $G^q = (V^q, E^q, A^q)$ and $G^t = (V^t, E^t, A^t)$ are the corresponding star graphs q and t.

$P(\mathbf{a}_i^q|\mathbf{a}_i^t)$ calculates the similarity between feature descriptors $\mathbf{a}_i^q$ and $\mathbf{a}_i^t$ using a sigmoid function [86]:

$$P(\mathbf{a}_i^q|\mathbf{a}_i^t) = \frac{1}{1 + \|\mathbf{a}_i^q - \mathbf{a}_i^t\|_1}. \tag{3.2}$$

$P(\mathbf{a}_{ic}^q|\mathbf{a}_{ic}^t)$ computes the similarity of relative location vectors $\mathbf{a}_{ic}^q$ and $\mathbf{a}_{ic}^t$ using a Gaussian function [12]:

$$P(\mathbf{a}_{ic}^q|\mathbf{a}_{ic}^t) \propto \exp(-(\mathbf{a}_{ic}^q - \mathbf{a}_{ic}^t)^T S_L^{-1} (\mathbf{a}_{ic}^q - \mathbf{a}_{ic}^t)) \tag{3.3}$$

where S_L is a covariance matrix to allow for some deviations in the node locations. Figure 3.2 illustrates the nodes and edges of the star graph and the ensemble matching concept.

We modify traditional ensemble matching in several ways to accelerate the matching process and improve the matching performance. First, a two step algorithm is employed to find the best match in the target for each node in the query. Multiple nodes in the query are allowed to match to the same node in the target, which we found to be better than a one-to-one matching policy. For a query node, the algorithm first finds the most similar D target features $\{\mathbf{a}_j^t\}_{j=1}^D$ in terms of feature descriptors (D is much smaller than the total feature amount in the target) among all the nodes by exhaustive searching. Then, calculate the location correlations only for these D features and select the node having the maximum overall similarity score. Second, to penalize the

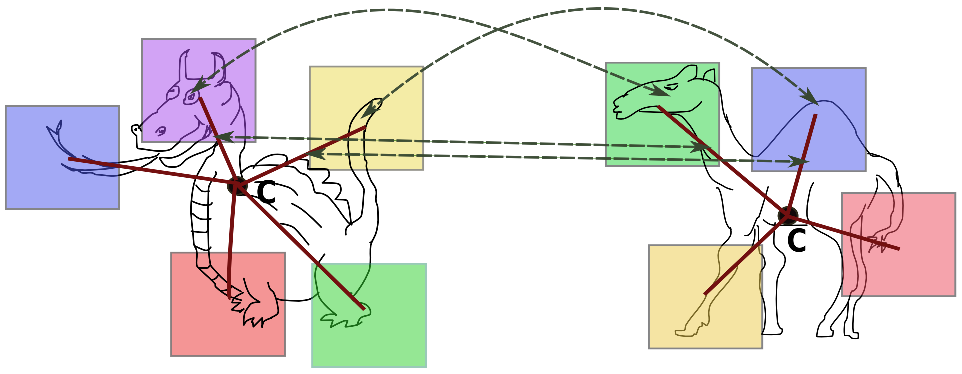


Figure 3.2: An illustration of star graphs for sketches and their ensemble matching. Patches are extracted from the sketch to construct the star graph, and each patch is connected to the graph center. The connections represent the patches' relative locations to the center. Different sketches can have different patch numbers. For two star graphs, both the patches and their corresponding relative locations are matched. Note that for illustrative clarity, only a few patches and matchings are shown for demonstration.

points not matched in the target, a penalty factor w is added which is defined as the proportion of the matched points in the target. Third, to generate the overall matching score above each node's matching score, we obtain the logarithm for computational convenience. The final score is normalized by the number of nodes N_s in the query star graph. The new function for ensemble matching is then:

$$P(G^q|G^t) = \frac{w\sum_{i\in N_s}\log\left(\max_j P(\mathbf{a}_j^q|\mathbf{a}_i^t)P(\mathbf{a}_{jc}^q|\mathbf{a}_{ic}^t)\right)}{N_s}. \tag{3.4}$$

Using Equation 3.4, the matching scores from G^q to G^t and from G^t to G^q are often different. To improve the stability of the final score, we average the scores for both directions:

$$P^f(G^q,G^t) = (P(G^t|G^q)+P(G^q|G^t))/2 \tag{3.5}$$

It is worth noting that if considered as a kernel function, Equation 3.5 is symmetric, and empirically we found it was always positive semi-definite in each cross-validation fold of our experiments. If we assume that there are in average n sampling points in each star graph, the computational complexity of the ensemble matching between two star graphs is then $O(2n(n+D))$. Several detailed examples of ensemble matching are shown in Figure 3.3. Only those points having both similar features and similar locations will be matched. It can be seen that ensemble matching addresses the holistic structure similarity well in the successful cases, and finds similar

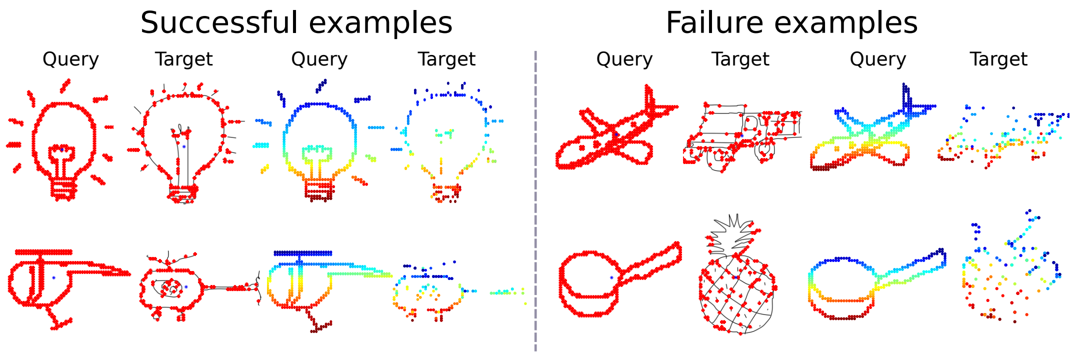


Figure 3.3: A visualization of ensemble matching. The left column includes successful examples and the right column includes failure cases. The image pair with red points indicates matched points. Multi-colored pairs indicate the detailed matching correspondence, where points with the same color are matched.

object parts in terms of structure in the failure cases.

3.1.3 Classification methods

A. *Support Vector Machines*

The SVM classifiers are trained for each category to classify sketches. For category l with $\mathbf{x}$ being the sketch representation, the score function used to decide the class of a query $\mathbf{x}$ is:

$$c^l(\mathbf{x}) = \sum_{s=1}^{S} \alpha_s^l k(\mathbf{x}_s^l, \mathbf{x}) + b^l \tag{3.6}$$

where α_s are the coefficients, b is the bias, k is a kernel function and s indexes support vectors $\mathbf{x}_s$. The response $c^l(\mathbf{x})$ measures how likely the query belongs to the lth category.

Radial basis function (RBF) kernel is used for k in the case of BOW representation:

$$k(\mathbf{x}_s^l, \mathbf{x}) = \exp(-\gamma \|\mathbf{x}_s^l - \mathbf{x}\|_2^2). \tag{3.7}$$

In the star graph case, the RBF kernel is replaced with Equation 3.5:

$$k(\mathbf{x}_s^l, \mathbf{x}) = P^f(\mathbf{x}_s^l, \mathbf{x}). \tag{3.8}$$

And one-vs-all approach is employed for the multiclass classification task.

B. Multiple Kernel Learning

Different features and representations have varying values for each category. In conventional SVMs, the kernel is defined on one feature type. Some features are more informative, but each feature may provide some complementary information. A weighted sum of kernels is therefore desirable to best utilize the discriminative power of each feature and representation. If we have a few kernels : $k_1, k_2, ...k_M$, using the same notation as Section A., their weighted linear combination is then formulated as:

$$k(\mathbf{x}_s^l, \mathbf{x}) = \sum_{m=1}^{M} \beta_m k_m(\mathbf{x}_s^l, \mathbf{x}) \tag{3.9}$$

where $\{\beta_m\}_{m=1}^M$ are weights reflecting the contribution of each kernel. The classifier score function is then:

$$f^l(\mathbf{x}) = \sum_{m=1}^{M} \beta_m c_m^l(\mathbf{x}). \tag{3.10}$$

$c_m(\mathbf{x})$ are the score functions for m different features and defined in Equation 3.6. MKL is used to select the inter-kernel weights $\{\beta_m\}_{m=1}^M$ and the coefficients $\{\alpha_s\}$ for each feature kernel which maximize the accuracy using Equation 3.10.

We use four kinds of kernels as described below. Besides ensemble matching, each kernel is applied to all BOW features. And the dimension of $\mathbf{x}$ is t.

1. Linear kernel

$$k(\mathbf{x}_s^l, \mathbf{x}) = \langle \mathbf{x}_s^l, \mathbf{x} \rangle \tag{3.11}$$

2. RBF kernel, described in Equation 3.7.

3. Chi square kernel

$$k(\mathbf{x}_s^l, \mathbf{x}) = \sum_t \frac{2x_{st}^l x_t}{x_{st}^l + x_t} \tag{3.12}$$

4. Histogram intersection kernel

$$k(\mathbf{x}_s^l, \mathbf{x}) = \sum_t \min(x_{st}^l, x_t) \tag{3.13}$$

5. Ensemble matching kernel, described in Equation 3.8 (for star graph only).

3.2 Experiments

This sections first evaluates SVM's performance with different features and representations. Then a thorough evaluation of fusing the features and representations using a MKL model follows, which delivers the state-of-the-art performance at the time of publication. Finally, the attribute experiments and applications are demonstrated.

3.2.1 Dataset and General Settings

Dataset We evaluate our methods on the sketch dataset with the most categories to date proposed by Eitz et al. [32], which has 250 categories and 20,000 sketches. The dataset was collected on Amazon Mechanical Turk from 1,350 non-expert subjects, thus the drawing style and sophistication level are diverse. Following [32], the sketches are normalized to 256×256 pixels.

Super-categories To define super-categories, we refer to WordNet [74]. The original category names is used to search their inherited hypernyms in WordNet. Then a few hypernyms that are representative and intuitive are chosen to be the super-categories' names, and each of them is used to group several original categories. Finally, 14 super-categories are defined to analyze different representations' performances. The specific super-categories' names and the number of categories in each are shown in Table 3.3.

Features Three basic types of features are evaluated in the proposed methods, including HOG, SSIM and Daisy. SSIM is computed with VGG's implementation [21]. The 'var_noise' parameter is set to 50,000, and the radical bins and angular bins are set to 5 and 12 respectively. On top of the 256×256 pixels sketch, a 51×51 grid is used to extract the sample points, and the local patch size is 90×90. VGG's saliency checking, homogeneity checking and second-nearest neighbor checking are all disabled, as they are not suitable for sketches. We sample the feature points along the sketch contours and use the same sampling points for all the features. HOG is computed using the VLFeat [98] implementation with each patch divided into 4×4 cells and the orientation is set to 4. Daisy is computed with CVLAB's implementation [96] with all the default settings as well.

Bag-of-words representation For all the mentioned features, a codebook of 1,000 visual words are used to obtain the BOW representation. 1,000,000 features are randomly sampled to generate the codebook via k-means clustering. The σ parameter for the Gaussian kernel is searched between [0.001,1].

Star graph representation As described in Section 3.1.2, SSIM is used to decide which grid intersections are used to construct the star graph, and other features will adopt these intersections. The center of the star graph is the center of mass of these intersections.

Parameter searching and training data size We employ 4-fold cross validation scheme to search for parameters. For both SVM and MKL, the γ and C parameters are searched between $[2^{-2}, 2^{8}]$. A coarse grid search is performed with an interval of 2^2 to find a best value X, followed by a fine grid search with an interval of $2^{0.25}$ among $[2^{-1}X, 2X]$. To evaluate the impact of training dataset size, the dataset is also separated into growing subsets (i.e., 20,40,60,80 sketches per category), and on each of the subset, the average 4-fold cross validation accuracy is calculated.

Support Vector Machines and Multiple Kernel Learning For single-kernel experiments we use the libsvm optimiser [18], and for multi-kernel experiments we use the UFOMKL optimiser [79].

3.2.2 Comparing Different Features' Performances on SVM

We compare the BOW representation of HOG, SSIM and Daisy, and the star graph representation on HOG (due to the computational cost of ensemble matching, we just select HOG to work with star graph as it is the reported best performing feature), with SVM classifiers. Figure 3.4 shows their performance with incrementally increased training set size. Table 3.1 shows the recognition accuracies of each feature when using the full training set. It can be seen that star graph representation performs better than BOW representation, and HOG is still the best performing feature.

Table 3.1: Classification performance of different features with SVM using the full training set.

	HOG	SSIM	Daisy	Star graph(HOG)
Acc.	55.12%	27.99%	43.80%	**56.42%**

3.2.3 Fusing Features and Kernel Metrics using Multiple Kernel Learning

Given the varying informativeness of each feature on BOW, we next investigate whether MKL can fuse these features in a complementary way. We train an RBF kernel MKL classifier with three features including BOW representation of HOG, SSIM and Daisy. The star graph kernel is also included and computed with Equation 3.8. With the complementary cues of multiple representations on multiple features, recognition performance reaches 62.61% (RBF Intermediate fusion result in Table 3.2(c); Intermediate fusion is explained later) Additionally, to show that

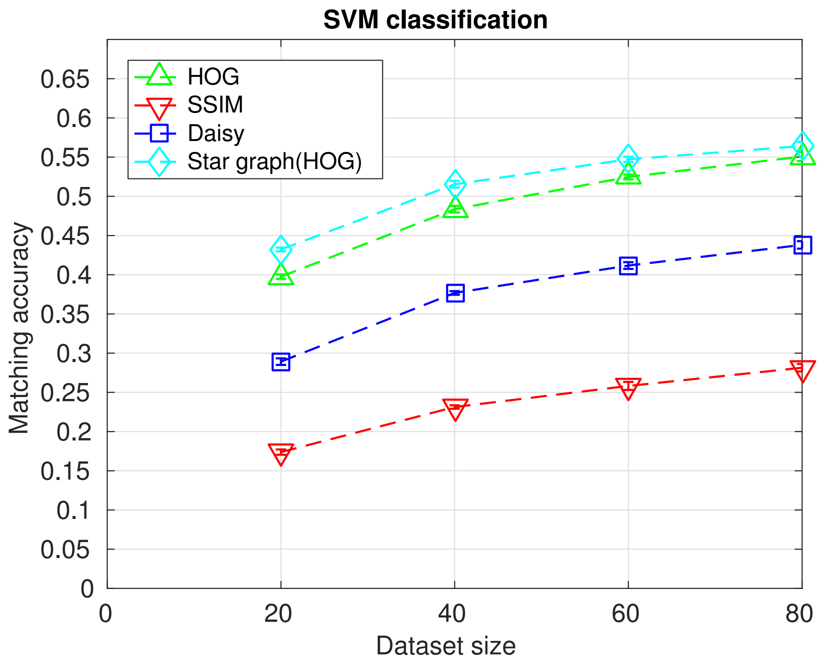


Figure 3.4: Performance comparison for varying data volume. The x axis indicates the number of the sketches used in training. Error bars are added to indicate one standard deviation, and the deviations are generally around 0.3% similar as in Table 3.2.

each feature has contributed to the overall result, we computed the MKL results without one feature at a time. The results are shown in Table 3.2(a). We also show the weight β_m from Equation 3.10 in Table 3.2(b) to help illustrating each feature's contribution, and the weights are generally consistent with the accuracy in Table 3.2(a), highlighting the contribution of star graph and HOG.

Beyond feature type, a pervasive design question in conventional sketch recognition is what is the right kernel metric to use for comparing images. Within the MKL framework, this question can be sidestepped as all kernel metrics can be used together synergistically. To demonstrate this, we further evaluated 3 additional kernel functions beyond RBF used thus far: linear, chi square (Chi2) and histogram intersection (HI) on all the features (star graph kernel is always included when using each kernel function, computed with Equation 3.8). The performance of all kernel functions is shown in Table 3.2(c) with HI kernel yielding the highest accuracy of 65.45%. Then, we compute all the kernels for each feature, and use them all in MKL, which yields an even better result of 65.81% (also shown in Table 3.2(c)). This performance significantly exceeds the state-of-the-art performances in [32], which are compared in Table 3.2(d). Importantly, these experiments show that not only does using all kernels and all features yield the best performance, but that tuning the choice of features and kernels [33, 34] is not necessary – the simple strategy of using them all together is best.

Table 3.2: (a) Recognition accuracy of MKL using all the features but excluding one feature each time to see the contribution of each feature (on RBF kernel). (b) The weight β_m (see Equation 3.10) is also shown for each feature to illustrate its relative importance in MKL (on RBF kernel). (c) The accuracies of early, intermediate and late fusions of the features with different kernel functions. The intermediate fusion (MKL) with all the features and all the kernel functions yields the best performance. (d) The performance comparison with previous works. The standard errors are also provided for all the accuracies when available.

(a)

Excluded	*HOG*	*SSIM*	*Daisy*	*Star graph(HOG)*
Acc.	$58.85 \pm 0.11\%$	$62.01 \pm 0.28\%$	$60.86 \pm 0.29\%$	$60.46 \pm 0.28\%$

(b)

Feature	*HOG*	*SSIM*	*Daisy*	*Star graph(HOG)*
Weight	0.0054	0.0047	0.0043	0.0098

(c)

Kernel	RBF	Chi2	HI	Linear	All
Intermediate fusion Acc.	$62.61 \pm 0.34\%$	$63.78 \pm 0.48\%$	$65.45 \pm 0.61\%$	$55.09 \pm 0.45\%$	$\mathbf{65.81 \pm 0.58}\%$
Early fusion Acc.	$61.20 \pm 0.44\%$	$63.45 \pm 0.45\%$	$64.82 \pm 0.59\%$	$57.41 \pm 0.42\%$	$64.38 \pm 0.48\%$
Late fusion Acc.	$61.48 \pm 0.31\%$	$56.75 \pm 0.31\%$	$63.74 \pm 0.49\%$	$60.90 \pm 0.16\%$	$56.14 \pm 0.25\%$

(d)

Methods	SVM [32]	MKL(All)
Acc.	56%	**65.81%**

Table 3.3: Comparison of SVM recognition performance grouped by super-category, using BOW and Star graph (Star) on HOG . The number of categories in each super-category is in parentheses. MKL results are also stated.

	Animal(48)	Plant(18)	Vehicle(27)	Electronics(27)	Clothing(8)
BOW	43.01%	61.62%	51.06%	55.42%	65.94%
Star	45.31%	63.97%	53.06%	58.38%	74.38%
MKL	53.47%	73.01%	58.94%	63.84%	75.78%
	Furniture(14)	Body part(20)	Building(10)	Sport(6)	Food(9)
BOW	47.32%	63.69%	53.63%	61.25%	59.86%
Star	50.63%	68.19%	57.75%	62.92%	56.67%
MKL	58.21%	73.25%	66.75%	71.46%	70.42%
	Instrument(7)	Commodity(45)	Weapon(6)	Nature(5)	
BOW	57.14%	59.97%	58.33%	78.75%	
Star	58.39%	56.42%	61.88%	67.25%	
MKL	68.04%	68.86%	70.00%	89.25%	

To provide a complete analysis on feature fusion, we also included results of two alternative strategies for fusion: a) the “early fusion” by feature stacking, and b) the “late fusion” by classifier voting. Our MKL method is referred to as the intermediate fusion, because it learns weights for the distance matricies. For the early fusion, we concatenate the BOW of the 3 basic features and use the chosen kernel function to obtain the kernel matrix, which is then averaged with the star graph kernel (computed with Equation 3.8)) without weighting. This averaged kernel matrix is then used in SVM for classification. For the late fusion, we make a kernel for each feature with the chosen kernel function (star graph kernel is computed with Equation 3.8) and train an SVM classifier for each of them. The output of this bank of SVMs is combined with majority voting to obtain the final classification result. Those results are also shown in Table 3.2(c).

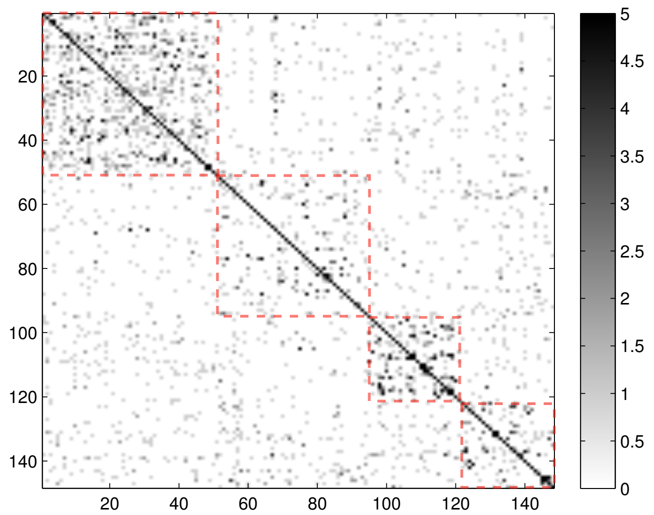
To offer insight into what types of sketches each representation is better at, the per super-category performance of SVM on star graph and BOW is provided in Table 3.3. Although for the overall result, star graph is only slightly better, star graph is evidently better at 11 super-categories, while BOW is better at only 3 super-categories. The per super-category performance of MKL is also shown in Table 3.3. After employing both representations, MKL achieves the best of both, with top results on every super-category. For the super-category analysis, we also show the confusion matrices of the 4 biggest super-categories (animal, commodity, vehicle and electronics) in Figure 3.5. It can be seen that the inside super-category confusions are much bigger than the between super-category confusions, especially for the animal and the vehicle super-categories, as the categories inside these two super-categories have more similar topology structures.

3.2.4 Attributes for Classification

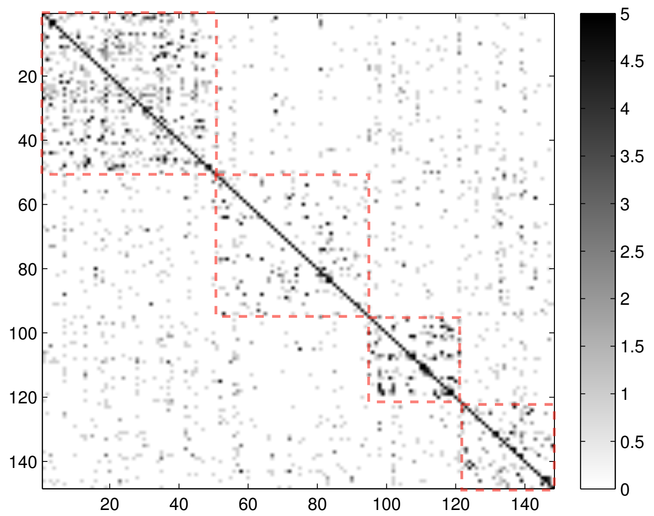
To see how attributes can help improving the recognition inside a super-category, we pick the animal super-category to perform a preliminary experiment. We borrowed some attributes from [62] and defined a few more attributes by searching the category names' inherited hypernyms in WordNet. Finally we selected 29 attributes for the animal set. The category/attribute table is shown in Figure 3.6.

To demonstrate the contribution of the attributes, we use the best MKL result with all the features as the baseline, and compare with the MKL result when adding the attribute feature. We also use SVM to test how attributes perform alone. A different evaluation scheme is adopted compared to the previous sections, as two loops of training are needed for a fair comparison: the attribute classifiers and the MKL/SVM classifiers. We divide the 80 sketches of each category into 2 subsets : s_1, s_2, with s_1 for training and testing attribute classifiers on HOG, s_2 for training and testing the MKL/SVM classifiers. On s_1, for each attribute classifier, we select its parameters γ and C by 4-fold cross validation also among $[2^{-2}, 2^{8}]$. When the attribute classifiers are obtained, they are used to compute the attribute features for s_2. Then s_2 is used to train and test the MKL/SVM classifiers through the same type of 4-fold cross validation. s_1 and s_2 are both set 40 in our experiment.

The recognition rate of each attribute classifier is shown in Table 3.4(a) and demonstrates



(a)



(b)

Figure 3.5: The confusion matrix of BOW and Star graph on HOG for 4 major super-categories: animal, commodity, vehicle, electronics. The matrices are sorted by category. Red dotted rectangles highlight within-category versus across-category confusion. The matrices are exaggerated via mapping values from 1 to 5 to the whole color range so numbers above 5 are shown the same color as 5 and small values are more clearly observed. The confusions inside most super-categories are much higher than the confusions between super-categories.

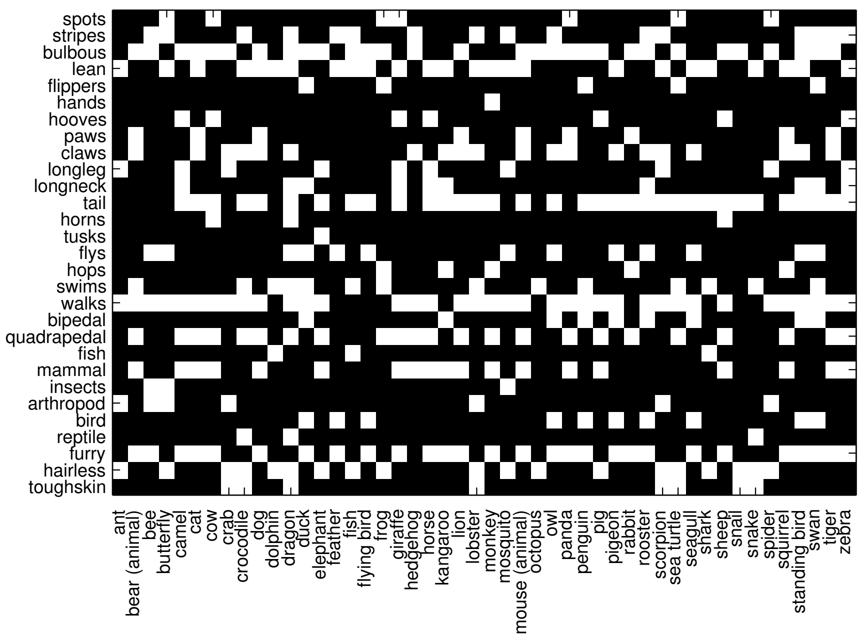


Figure 3.6: The ground-truth class-attribute matrix used in our experiments. X axis (horizontal) indicates categories and the Y axis (vertical) indicates attributes (The attributes are borrowed from [62]).

that, despite the sparsity of features available in sketches, attributes are quite reliably detected. Table 3.4(b) offers a comparison of the recognition accuracy of MKL without attributes, MKL with attributes, and SVM solely with attributes. Evidently attributes can further improve the recognition of sketches. This is because they provide a representation which is discriminative by design – highlighting individual semantic properties that are useful for distinguishing categories. It is thus reasonable to expect that attribute definitions for other super-categories besides animals should also provide solid improvements in results.

Table 3.4: (a) The classification accuracies of the attribute classifiers. (b) The comparison of recognition accuracies by using MKL on all the features, MKL on all the features with attributes and SVM on attributes.

(a)

Attribute	spots	stripes	bulbous	lean	flippers	hands	hooves	paws
Accuracy	86.30%	74.84%	71.93%	69.58%	91.04%	97.92%	91.67%	84.17%
Attribute	longneck	tail	horns	tusks	flys	hops	swims	walks
Accuracy	86.04%	79.48%	94.48%	98.59%	81.46%	90.78%	77.81%	85.94%
Attribute	fish	mammal	insects	arthropod	bird	reptile	furry	hairless
Accuracy	96.15%	82.40%	95.36%	88.91%	88.54%	95.47%	76.20%	76.20%
Attribute	claws	longleg	bipedal	quadrapedal	toughskin			
Accuracy	72.81%	84.64%	88.75%	80.00%	87.76%			

(b)

	MKL(All)&Attributes	MKL(All)	Attributes
Accuracy	**52.39%**	50.63%	36.77%

3.2.5 Exploiting Sketch Attributes

A possible interesting future direction for sketch attributes is to provide the opportunity for novel sketch-understanding based applications.

- A first application is to allow the user to retrieve sketches by attribute rather than by category, by sorting sketches via attribute classifier rather than category classifier (e.g., spotty or stripy). The first results in the sorted list possess the attribute with high probability and the last few results possess the attribute with low probability (or equivalently the inverse attribute with high probability, e.g., long legs versus short legs). Figure 3.7 illustrates this for three attributes in the top row of each section.

- A second application is to allow the user to retrieve sketches based on a combination of category and attribute. There are various potential ways to achieve this, but we illustrate the concept by querying the category first and then sorting by attributes. In the second row of each section in Figure 3.7 we show the results of sorting attributes within ground truth categories (thus separating categorization errors from attribute-sorting errors). In the third rows, we show the results for a fully automated query which retrieves the top 20 confident sketches for the specified category, and then sort these by the attribute scores. In each case, both the top 5 results and bottom 5 results for each category are shown to illustrate the contrast.

3.2.6 Computational Cost

Our experiments were performed on a CentOS 7 system with 3.40 GHz Intel Core i7 processor and 16 GB main memory. We used precomputed kernels (pairwise similarities between all the training sketch pairs) to accelerate the MKL training process. When we used all the 80 sketches in each category to train, for the star graph kernel, it took around 1 hour to compute the full kernel matrix. And for each BOW kernel, it took around 10 minutes to compute the full kernel matrix. The cross-validation process normally took 10-40 hours to finish depending on the number of combined kernels.

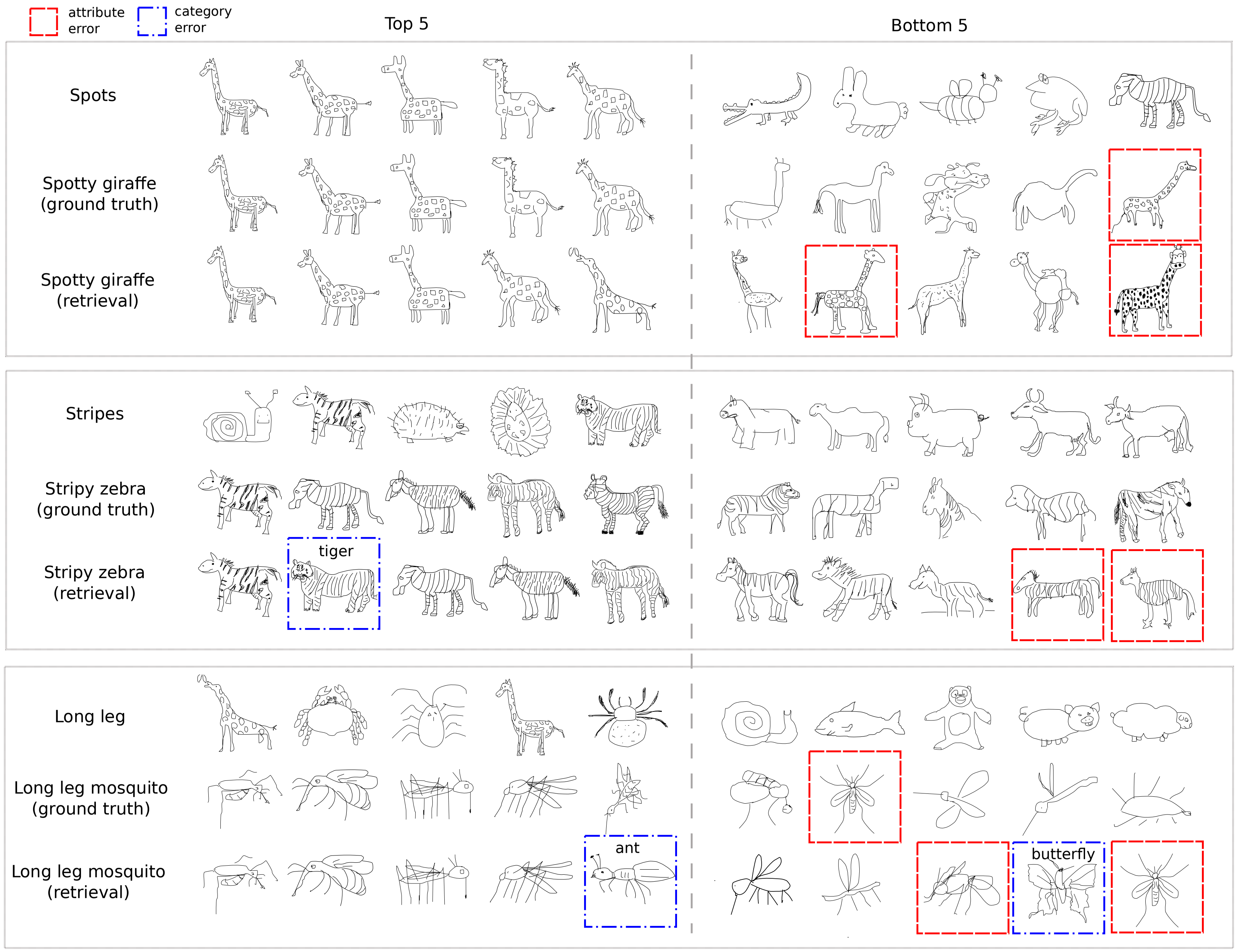

attribute error
category error
Top 5
Bottom 5
Spots
Spotty giraffe (ground truth)
Spotty giraffe (retrieval)
Stripes
Stripy zebra (ground truth)
Stripy zebra (retrieval)
tiger
Long leg
Long leg mosquito (ground truth)
Long leg mosquito (retrieval)
ant
butterfly

Figure 3.7: The results of unconditional attribute query (top rows), class-attribute query with ground truth class (middle rows) and automatically recognised class (bottom rows). Both of the top 5 and last 5 results are selected here to demonstrate the contrast. Rectangles refer to mistakes.

3.3 Discussions

After the experiments, we can see that the proposed star graph representation performs evidently better than the BOW representation, and HOG is confirmed to be the best among several popular features for sketch recognition, which is the same as sketch-based image retrieval. By fusing multiple features and representations, MKL demonstrated significant improvement over the previous state-of-the-art, from 56% to 65.81%. And histogram intersection kernel is rated the best kernel metric. Over and above that, we for the first time study attributes for sketches, and demonstrate their effectiveness in reducing confusion inside one super-category. Moreover, we show how the high-level semantic nature of the attribute feature allows novel applications such as query by attribute or category-attribute description.

Chapter 4

Fine-Grained Sketch-Based Image Retrieval

As discussed in Section 2.2.1, intra-category fine-grained sketch-based image retrieval is the most promising direction for the sketch-based image retrieval (SBIR) problem. We show the contrasts of text-based image retrieval and conventional SBIR with our proposed fine-grained SBIR in Figure 4.1, in which the advantage of fine-grained SBIR to differentiate fine-grained variations of objects can be clearly seen.

The fine-grained SBIR should look at sketching the object scenario, as its potentially applicable situation would be commodity retrieval. And object localization is necessary to ensure credible object-oriented retrieval. Besides, a scheme that utilizes local feature descriptor and has the ability to address abstractions/distortions is needed. A new dataset that is composed of real-life images and has sketch-image similarity ratings is also compulsory for evaluation. In this section, we propose a novel fine-grained SBIR framework which specially addresses the above problems, and its corresponding evaluation benchmark is also presented.

To realize our framework, we use deformable part-based model (DPM) both as an object detector and cross-domain representation to bridge the sketch and image domains. Similarity comparison of DPMs is performed via graph matching, taking account both geometry and appearance information encoded in the DPMs. Specifically, we achieve fine-grained SBIR by first training per category DPMs independently for each domain, then aligning DPM-mixture components across the domains to obtain a component correspondence via graph matching. At retrieval time, we use the trained DPMs to detect both the probe sketch and all gallery images, and use the learned component alignment mapping to rank the images for the first round. Then we per-

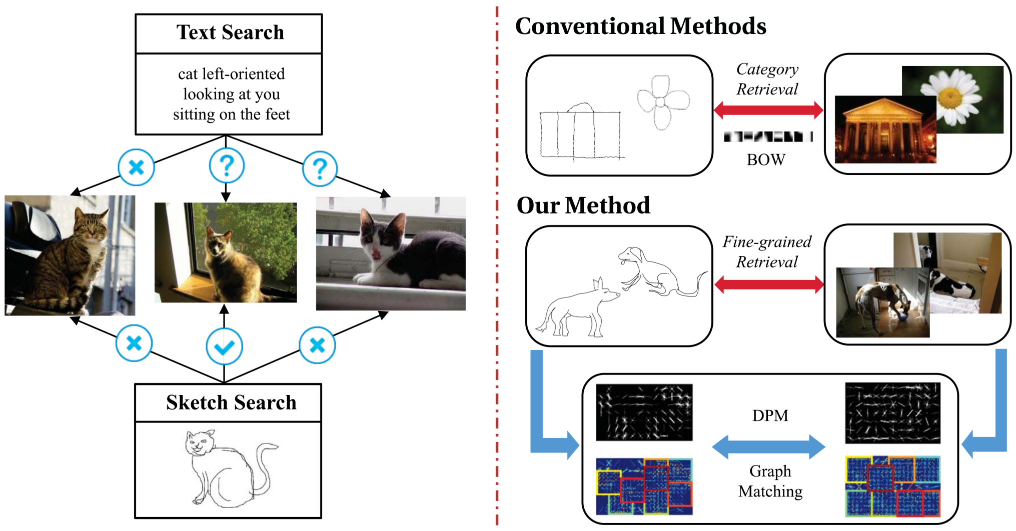


Figure 4.1: The advantages of SBIR over text-based image retrieval and fine-grained SBIR over conventional SBIR.

form finer part alignment on the DPM detections via graph matching to rank the images for the second round. Intuitively, the component-level matching ensures retrieved objects are in broadly the same appearance/pose as the sketch. The detection-level matching enables matching fine-grained details such as body configuration (e.g., limb position) and part appearances (e.g., the shapes of claws). We demonstrate our proposed system's performance quantitatively and qualitatively against previous bag-of-words [89] and spatial-pyramid [63] based methods. To perform the evaluation, we create a real-life SBIR dataset for fine-grained retrieval by sampling sketches from the TU-Berlin dataset [32] and images from the challenging PASCAL VOC dataset [36]. Ground-truth for sketch-image pairwise similarities within each category is carefully labeled according to four criteria for fine-grained similarity on a portion of the proposed dataset used for testing. This ground-truth then provides overall criterion for performance evaluation.

4.1 Methodology

We start this section with introducing basic notations for deformable part-based model, followed by the formulation of our graph matching method. Given those, we finally illustrate our overall framework in detail.

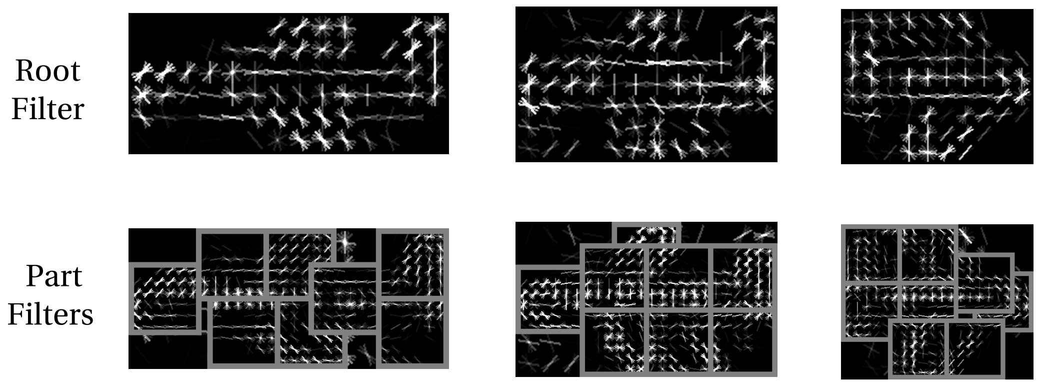


Figure 4.2: A mixture of DPMs for the airplane category. There are three learned components. For each component, the first row shows the root filter and the second row shows the part filters.

4.1.1 Deformable Part-based Model and Notations

To use deformable part-based model (DPM), a mixture of DPMs is trained from a set of images, which comprises several components and is used for detection. During detection, only one component will be triggered for one object in the image, and a corresponding DPM detection is obtained for that object. Both DPM components and detections are in the form of a two-layer structure composed of a root filter and a set of N part filters connected as a star graph (part filter represents a small portion of the root filter and has twice the resolution of the root filter; all part filters have the same size). One example for the mixture of DPMs of the airplane category is illustrated in Figure 4.2.

We denote this two-layer structure as $M = (r, G)$ and refer it as DPM, where $r = (w, h, \mathbf{f})$ specifies the width w, height h and global appearance feature (HOG is employed) of the root filter; and $G = (V, E, A)$ represents the star graph composed of the part filters. For the star graph G, V represents a set of nodes, E, edges, and A, attributes. More specifically, $V = \{v_i\}_{i=1}^{N} \cup c$ represents all N parts v_i and the center c that is the center of r. Each node v_i has an associated attribute $\mathbf{a}_i \in A$ describing appearance feature (also HOG) of v_i, and an associated edge $\mathbf{e}_{ic} \in E$ describing the geometrical relationship between the center of v_i and c in terms of relative coordinate offset.

4.1.2 Graph Matching for Deformable Part-based Model

The key challenge for matching sketch with images in our approach is the computation of the similarity metric between the DPMs across domains, including both DPM model components and DPM image detections. In this section, we introduce our similarity measure $S(M^R|M^T)$ between two DPMs, M^R and M^T.

Our matching objective accounts for both appearance and geometric information encoded in the DPMs, as well as both layers of representation, i.e., root filter r and part filter star graph G. The similarity function is defined as:

$$S(M^R|M^T) = \gamma S_{root}(M^R|M^T) + (1-\gamma)S_{part}(M^R|M^T) \tag{4.1}$$

where S_{root} is the root similarity and S_{part} is the part similarity; γ is a weighting factor balancing root and part similarities.

Root Similarity (S_{root}) Given that all part filters of a DPM share a common size, differences in root size and aspect ratio implicitly reflects pose variations. Therefore, we introduce a term to represent root filter similarity based on appearance features, sizes and aspect ratios of the root filters of M^R and M^T. We denote the root filters as r^R and r^T, the widths as w^R and w^T, the heights as h^R and h^T, and the appearance features as $\mathbf{f}^R$ and $\mathbf{f}^T$ respectively. Then, the root similarity metric can be written as I:

$$S_{root}(M^R|M^T) = \delta(\mathbf{f}^R\mathbf{f}^T) + (1-\delta)\exp\left(-|\frac{w^R}{h^R} - \frac{w^T}{h^T}|\frac{\max(h^R,h^T)}{\min(h^R,h^T)}\right), \tag{4.2}$$

where the first term represents appearance similarity (dot product is inherited from [38]), and the second term accounts for size and aspect ratio variations of the root filters. δ is a linear weighting factor balancing the significance of both terms. The appearance feature $\mathbf{f}^R$ and $\mathbf{f}^T$ are extracted after normalizing r^R and r^T to the same size.

Part Similarity (S_{part}) The part-level similarity between two DPMs depends on the unknown mapping of the parts from one DPM to another. We achieve this by finding the mapping that maximizes the overall geometrical and appearance consistency between the two DPMs' part filters. Since the part filters are organized as a star graph, we formalize this mapping task as a graph-matching problem between the part filter star graphs and an illustration of this process is shown in Figure 4.3.

Given two DPMs M^R and M^T, their part filters are represented as star graphs $G^R = (V^R, E^R, A^R)$ and $G^T = (V^T, E^T, A^T)$. We are going to find out a set of one-to-one matchings from all the nodes in V^R to all the nodes in V^T that maximizes the overall geometrical and appearance consistency of G^R and G^T. The mutual consistency of geometrical and appearance attributes between one pair of matching candidates (v_i^R, v_a^T) and (v_j^R, v_b^T) can be described by an affinity function

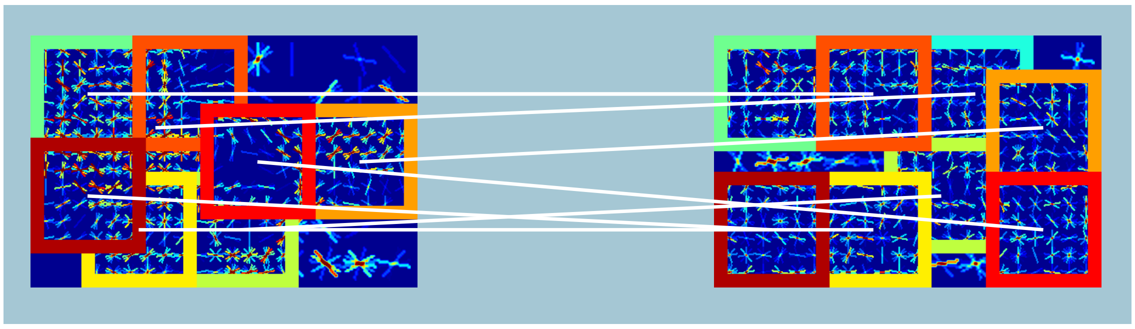


Figure 4.3: The part correspondences of two cat DPM components are sorted by graph matching. The left component is from the sketch domain while the right one is from the image domain. The parts with the same color are matched parts, which are also connected by a white line. Some part may be overlapped by other part.

$\mathbf{W}_{ia;jb} = f(\mathbf{a}_i^R, \mathbf{a}_j^R, \mathbf{e}_{ic}^R, \mathbf{e}_{jc}^R, \mathbf{a}_a^T, \mathbf{a}_b^T, \mathbf{e}_{ac}^T, \mathbf{e}_{bc}^T)$. It follows that we can construct an affinity matrix $\mathbf{W}$, whose non-diagonal element $\mathbf{W}_{ia;jb}$ contains a pair-wise affinity between two matching candidates (v_i^R, v_a^T) and (v_j^R, v_b^T) and whose diagonal element $\mathbf{W}_{ia;ia}$ denotes a unary affinity of one matching candidate (v_i^R, v_a^T).

If the number of parts of DPM is N, the correspondence between the parts of two DPMs can be represented by an assignment matrix $\mathbf{X} \in \{0,1\}^{N \times N}$, where $\mathbf{X}_{ia} = 1$ states that node v_i^R corresponds to node v_a^T. It can then be further substituted by its column-wise vectorized replica $\mathbf{x} \in \{0,1\}^{N \cdot N}$. Finally, the graph matching problem can be formulated as seeking an assignment $\mathbf{x}^*$ that maximizes the quadratic score function I:

$$\begin{aligned} \mathbf{x}^* &= \arg\max_{\mathbf{x}} (\mathbf{x}^T \mathbf{W} \mathbf{x}) \\ s.t.\ \ \mathbf{x} &\in \{0,1\}^{N \cdot N}, \forall i \sum_{a=1}^{N} \mathbf{x}_{ia} \leq 1, \forall a \sum_{i=1}^{N} \mathbf{x}_{ia} \leq 1, \end{aligned} \tag{4.3}$$

where the two-way constrains define a one-to-one matching from G^R to G^T. It follows that the part similarity can be calculated by :

$$S_{part}(M^R | M^T) = \mathbf{x}^{*T} \mathbf{W} \mathbf{x}^* \tag{4.4}$$

where $\mathbf{W}$ is the affinity matrix given by:

$$\mathbf{W}_{ia;jb} = \max(s_{app}(m_{ia}) s_{geo}(m_{ia}) + s_{app}(m_{jb}) s_{geo}(m_{jb}), 0) \tag{4.5}$$

where $m_{ia} = (\mathbf{a}_i^R, \mathbf{a}_a^T, \mathbf{e}_{ic}^R, \mathbf{e}_{ac}^T)$ and $m_{jb} = (\mathbf{a}_j^R, \mathbf{b}_b^T, \mathbf{e}_{jc}^R, \mathbf{e}_{bc}^T)$ represent matching pair (v_i^R, v_a^T) and

(v_j^R, v_b^T), respectively. $\mathbf{W}_{ia;jb}$ denotes the overall similarity between such pairs, in which $s_{app}(m_{ia})$ denotes feature similarity, $s_{geo}(m_{ia})$ represents geometrical similarity, and they can be computed as follows:

$$s_{app}(m_{ia}) = \mathbf{a}_i^R \mathbf{a}_a^T \tag{4.6}$$

$$s_{geo}(m_{ia}) = \exp(-(\mathbf{e}_{ic}^R - \mathbf{e}_{ac}^T)^T S_D^{-1} (\mathbf{e}_{ic}^R - \mathbf{e}_{ac}^T)) \tag{4.7}$$

where S_D is a constant covariance matrix controlling the allowed deviation of the matched cross-domain parts and is empirically set to the normalized side length of the part of DPM. $s_{geo}(m_{jb})$ and $s_{app}(m_{jb})$ can also be calculated as above.

In principle, any graph matching algorithm that is capable of solving a binary quadratic maximisation function can be used to solve Equation 4.3. In work, we employ the method of [24] that delivers good performance for our purpose.

4.1.3 Algorithm Design

The desired input of our proposed method is a sketch query q with known category, and the output is a sequence of images from the same category ordered by their similarities with the query q in terms of pose/appearance details. Achieving this *fine-grained* SBIR requires two major steps: (i) Training: DPM training and component alignment; (ii) Retrieval: *fine-grained* retrieval based on matching a query sketch DPM detection with image DPM detections. Below, we refer to DPM component as M_c, and DPM detection as M_d, and offer the detailed algorithms for the two steps:

Algorithm 1 DPM Training and component alignment

```
1: Training sketch mixture DPM $L^s = \{M_c^i\}_{i=1}^Q$ on sketch domain
2: Training image mixture DPM $L^p = \{M_c^j\}_{j=1}^P$ on image domain
3: for $i = 1 \to Q$ do
4:     for $j = 1 \to P$ do
5:         $\mathbf{t}(j) = S(M_c^i | M_c^j)$        ▷ Eq. (4.1), calculating the similarity of two components
6:     end for
7:     Rearrange $\{M_c^j\}$ in descending order into $\{M_c^j\}^i$ according to similarities in $\mathbf{t}$
8:     Store $\{M_c^j\}^i$
9:   ▷ For each sketch component $M_c^i$, its similarity order (from more similar to less similar) to
     the image components $\{M_c^j\}$ are saved here as $\{M_c^j\}^i$
10: end for
11:   ▷ As each component represents a coarse pose category (e.g., left, right or 45° views), this
     step will establish a consistent coarse pose mapping across domains
```

Algorithm 2 Fine-grained Retrieval

```
1:  Input query sketch $q$
2:  Detect on $q$ with mixture DPM $L^s$, obtaining $M_d^q$ and triggering component $M_c^i$
3:  Detect on all the images with mixture DPM $L^p$, obtaining detections $\{M_d^k\}_{k=1}^F$
4:  Group the images into $P$ groups $\{G^j\}_{j=1}^P$ according to which component $M_c^j$ detected it
    ($G^j = \{M_d^{k^j}\}_{k^j=1}^{F^j}, \sum_{j=1}^P F^j = F$)
5:  Sort $\{G^j\}$ into the same order of $\{M_c^j\}^i$ obtained in the first step
6:                                        ▷ This will ensure the consistency of the coarse pose
7:  for $j = 1 \to P$ do
8:      for $k^j = 1 \to F^j$ do
9:          $\mathbf{t}(k^j) = S(M_d^q | M_d^{k^j})$   ▷ Eq. (4.1), calculating the similarity of two detections
10:     end for
11:     Rearrange $\{M_d^{k^j}\}$ in descending order according to similarities in $\mathbf{t}$
12:                   ▷ This will ensure the consistency of the detailed part shape and appearance
13: end for
```

4.2 Experiments

In this section, we first introduce a challenging SBIR dataset with human labels that enables *fine-grained* SBIR performance to be quantified. We then use this dataset to evaluate performance of the proposed *fine-grained* SBIR framework compared to conventional baselines [15, 16, 34, 51, 53, 56, 100] employing bag-of-words (BOW) and spatial pyramid (SP).

4.2.1 SBIR Dataset and Annotation

We create our SBIR dataset by intersecting 14 common categories from the TU-Berlin sketch dataset and PASCAL VOC dataset [36], resulting in a new dataset of $14 \times 80 = 1,120$ sketches and $7,267$ images (made up of $14 \times n_i$ images, where n_i is the total number of images in the corresponding PASCAL category).

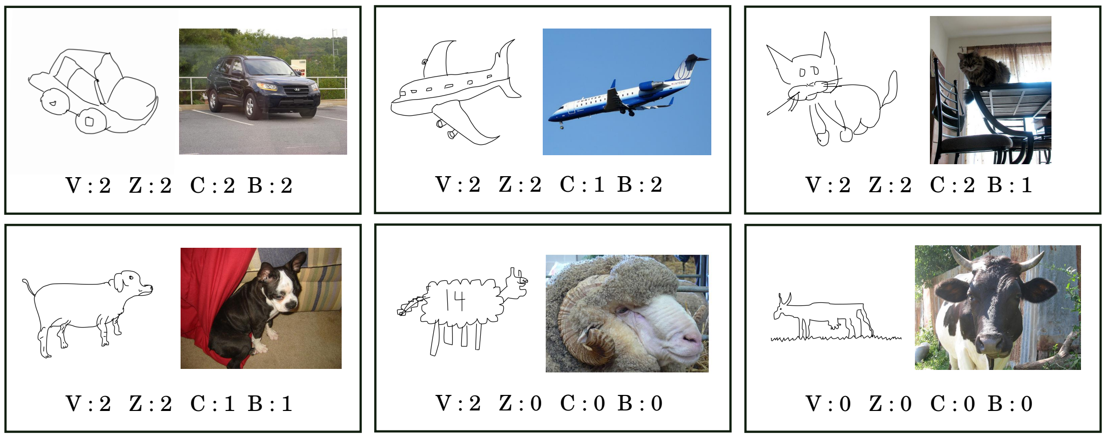


Figure 4.4: Examples of sketch-image pairs and their human rated similarity scores from our dataset.

We divide the whole dataset into testing and training sets of the equal size. To enable quantitative evaluation, we manually annotate a subset of the testing set with exhaustive pairwise similarity ground-truth. Specifically, 6 sketches and 60 images from each category are sampled from the full testing set, and sketch-image pair has its similarity manually annotated. For each sketch-image pair ($14 \times 6 \times 60 = 5,040$ pairs in total), we score their similarity in terms of four independent criteria: (i) viewpoint (V), e.g., left or right, (ii) zoom (Z), e.g., head only or whole body; (iii) configuration (C), e.g., position and shape of the limbs; (iv) body feature (B), e.g., fat or thin. For each criterion, we annotate ($5,040 \times 4 = 20,160$ annotations in total) three levels of similarity: 0 for not similar, 1 for similar and 2 for very similar. Figure 4.4 includes some example annotations. Previously, Bimbo and Pala [9] and Eitz et al. [34] have done similar sketch-image pair similarity ratings. Not only is our dataset's size significantly larger than them (5,040 pairs comparing to 66 pairs in [9] and 1,240 pairs in [34]), we also employ 4 criteria for similarity rating that is more accurate and practical in annotation.

4.2.2 Experimental Settings

We compare our framework to HOG Bag-of-Words and Spatial Pyramid baselines. The settings for each model are given as follows.

Bag-of-Words Following common practice [32, 34], to compute the BOW representation, images are first converted into edge maps using Canny edge detector [13]. Both images and sketches are then scaled into a fixed size of 256×256 pixels. HOG features are generated from sketch/image patches of the size 90×90 pixels. A 51×51 grid is applied to each sketch/image, and the patches are centered in the grid intersections. A large set of n features are randomly sampled from all HOG features extracted (including both sketch and image features). Afterwards, k-means clustering is employed to cluster those n features into H clusters. A code book $\mathcal{U} = \{\mathbf{u}_i\}_{i=1}^{H}$ is formed using the mean values of the clusters. After obtaining the codebook, a feature $\mathbf{f}$ is represented by its distance to all the words $\mathbf{u}_i$. The distance is measured by Gaussian kernel with parameter σ. We set $n = 1,000,000$, $H = 2000$, $\sigma = 0.1$ for our experiments.

Spatial Pyramid The spatial pyramid strategy [63] aims to encode the geometrical structure of BOW by partitioning the image into increasingly finer equal sub-regions (i.e., in level 1 the image has 1×1 region, and in level 2 the image has 2×2 regions) and compute the BOW for each sub-region. The final representation is a concatenated vector of weighted BOW from all the sub-regions. In our experiment, we use 2 levels of pyramid, and adopted the implementation of [63].

Gradient field HOG Gradient field HOG (GFHOG) was proposed by Hu et al. [51] for SBIR and obtained the best performance in their evaluation comparing to several popular features [51, 53]. To compute the GFHOG descriptor, a gradient field is firstly constructed for the sketch or the edge map, and then HOG features are extracted along the strokes or the edges on multiple scales. Finally, the BOW representation is employed to encode the set of GFHOG features on each sketch or image. We also compare to this feature descriptor, and employ Hu et al. [53]'s implementation. We set the vocabulary size $H = 3500$ and employ the histogram intersection distance, according to the best setting of [53].

DPM training and detection We train DPMs on the full training set of sketches/images in each domain for each category, using the implementation of [38]. Each DPM is set to 3 mixture components and 8 parts per component. For each category, the sketches/images of that category are used as positive training examples while those from all remaining categories are employed as negative examples. During training, bounding boxes provided by PASCAL VOC pascal-voc-2010 are used to crop image objects, and sketch bounding box is extracted from the borders of the sketch object. During detection, we choose the DPM detection with the largest probability in each image.

Graph Matching Our graph matching works both on the obtained DPM components and detections. Two parameters, the root-part weight γ and the root filter appearance-geometry weight δ, are optimized by searching among $[0, 1]$ with interval of 0.1 on half of the annotated dataset, and applied to the other half upon testing.

4.2.3 SBIR Performance Evaluation

Bimbo and Pala [9] uses ranking agreement between the humans and the algorithm for evaluation and only looks at the top 6 results. As some images may be equally similar to the sketch, their local rankings are not very important. So when the size of the annotated images is big (e.g., 60 in our case), the ranking agreement may be too rigid for evaluation. Eitz et al. [34] also has this problem and they do not care if the annotated images are in the top results or not. Therefore to overcome their weaknesses, we perform quantitative evaluation on the ground-truth dataset based on rated scores. Given a probe sketch, we retrieve K images, and accumulate the ground-truth similarity scores of those K images as the performance metric (the larger the better). Table 4.1 summarizes our results when $K = 5$ and $K = 10$. The per-category score is the average over all

Table 4.1: SBIR performance comparison for top $K = 5, 10$ retrievals: Ours, Spatial Pyramid (SP), Bag-of-Words (BOW) and gradient field HOG (GFHOG).

(a) $K = 5$

Top 5	Ours	SP	BOW	GFHOG
airplane	22.00	20.33	18.83	13.67
bicycle	11.67	13.83	13.67	9.00
standing bird	14.67	13.50	11.33	10.33
bus	24.67	10.50	10.50	15.00
car (sedan)	18.83	14.50	13.50	6.67
cat	12.17	7.67	7.50	11.67
chair	20.00	20.33	19.50	9.67
cow	19.67	14.00	13.17	15.00
table	8.67	3.33	4.33	6.00
dog	9.50	6.83	5.50	3.00
horse	31.67	7.33	4.67	6.33
motorbike	22.50	9.00	11.50	4.00
sheep	17.67	5.00	6.17	9.67
train	12.50	10.33	11.50	11.00
Average	**17.58**	11.18	10.83	9.36

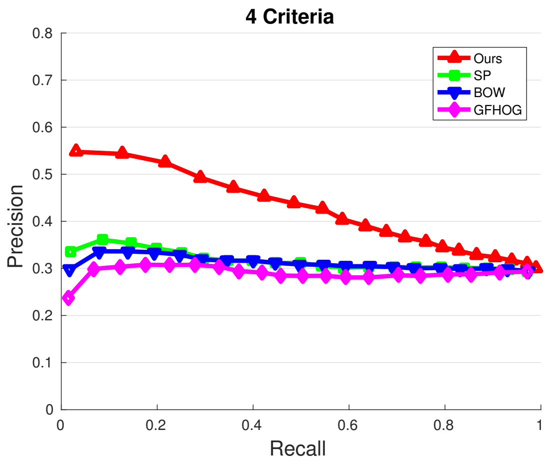
(b) $K = 10$

Top 10	Ours	SP	BOW	GFHOG
airplane	48.17	34.00	32.33	18.67
bicycle	25.50	26.67	25.00	21.33
standing bird	26.33	25.83	25.50	20.67
bus	37.67	19.17	20.00	29.00
car (sedan)	36.50	27.00	26.33	18.00
cat	20.33	16.17	15.17	23.33
chair	38.50	33.50	31.67	26.00
cow	27.17	26.50	25.33	29.00
table	12.33	9.00	9.33	11.00
dog	20.33	11.17	11.00	6.00
horse	57.33	14.50	13.33	13.00
motorbike	38.17	20.17	20.50	9.67
sheep	23.67	11.50	12.33	19.00
train	26.67	25.33	23.50	20.67
Average	**31.33**	21.46	20.81	18.95

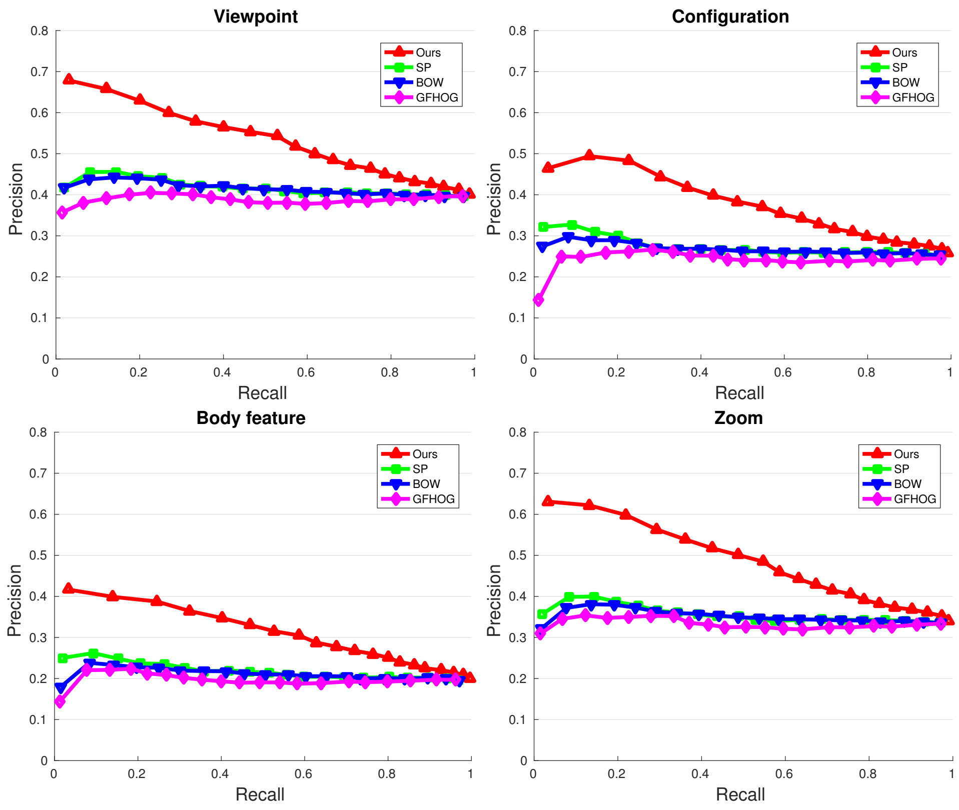
3 query sketches in that category. It can be seen that our method significantly outperforms the conventional alternatives on most categories. Although GFHOG has outperformed other features in previous evaluations, it obtained less favorable results in the fine-grained SBIR task. Our possible explanation for this is that the interpolation process for constructing the gradient field has a blurring effect on the images. This effect could elevate the holistic structure of the object yet diminish the fine details. Thus, the GFHOG may suit the inter-category SBIR better than the intra-category fine-grained SBIR.

In Figure 4.5 we offer precision-recall curves computed over the full available range $K = 1 : 60$, utilizing all four criteria combined (Figure 4.5(a)) and each criterion alone (Figure 4.5(b)). Given an image with retrieval score A, we compute its precision as $precision = A/Y$, where Y is the maximum score an image can have (8 in our case), and recall as $recall = A/U$, where U is the accumulative image score of the entire category. The results show that our SBIR framework provides the biggest margin in its ability to perform at high-precision, suggesting that it has a much better chance of retrieving the most relevant images in the first few results. Moreover, our framework is more effective at *fine-grained* SBIR under all individual criteria.

Qualitative retrieval results with ground-truth annotation are provided in Figure 4.6. It can be seen that our SBIR framework generally retrieves images having the same pose as the sketch query. This is because the DPM training has summarized and encoded the representative poses in the category as components, and our matching has corresponded similar representative poses



(a)



(b)

Figure 4.5: Precision-recall curves comparing bag-of-words (BOW), spatial pyramid (SP), gradient field HOG (GFHOG) and our method (Ours), using: (a) all 4 criteria, (b) criterion viewpoint, configuration, body feature, zoom separately.

from two domains. Figure 4.7 shows the intra-category SBIR using different sketches with our proposed system. It can be clearly seen that the convenience to distinguish intra-category variations with sketches, which is hard to achieve in conventional text-based image retrieval.

To provide further insight into the mechanism of our model, in particular graph matching, we also demonstrate retrieval using only root similarity versus both root and part similarities.

Figure 4.8 shows a qualitative comparison, in this case querying the entire test set rather than just the subset with ground-truth similarity annotation, as more sufficient images available for evaluation. Part-level graph-matching is illustrated in the second row by way of color coding the parts based on their sketch-image correspondence. Part similarity helps our method retrieve images with more similar *fine-grained* details (e.g., the bent legs of the running horse). Although not all the parts are perfectly aligned, their cumulative impact still helps to retrieve better matches than using the root similarity alone.

4.2.4 Computational Cost

Our experiments were performed on a CentOS 7 system with 3.40 GHz Intel Core i7 processor and 16 GB main memory. The major computational cost stayed in the DPM training stage. It generally took half an hour to train a sketch mixture DPM and 12 hours to train an image mixture DPM. Once the DPM mixtures are obtained, a detection procedure on one sketch or image takes less than 10 seconds. Because DPM detection on images is performed offline, the retrieval for one query on our test dataset takes around 20 seconds.

4.3 Discussions

Over the experiments, we can see that a specially designed representation and corresponding similarity metric, i.e. DPM and graph matching, can significantly improve the fine-grained retrieval ability of SBIR. The key for the success of the DPM is the hierarchical structure that encodes both the holistic shape and part appearances. The detection/object localization function of DPM also opens SBIR's access to the challenge real-life images with cluttered background. Furthermore, our proposed dataset and evaluation scheme have provided an effective benchmark to evaluate fine-grained SBIR system's performance. However, some issues still exist for our system, e.g., the high computational cost of the DPM detection and the graph matching process and the moderate detection accuracy of DPM, which need to be further tackled before our system can be applied to daily life. Further discussion about our system and future fine-grained SBIR will be continued in Chapter 6.

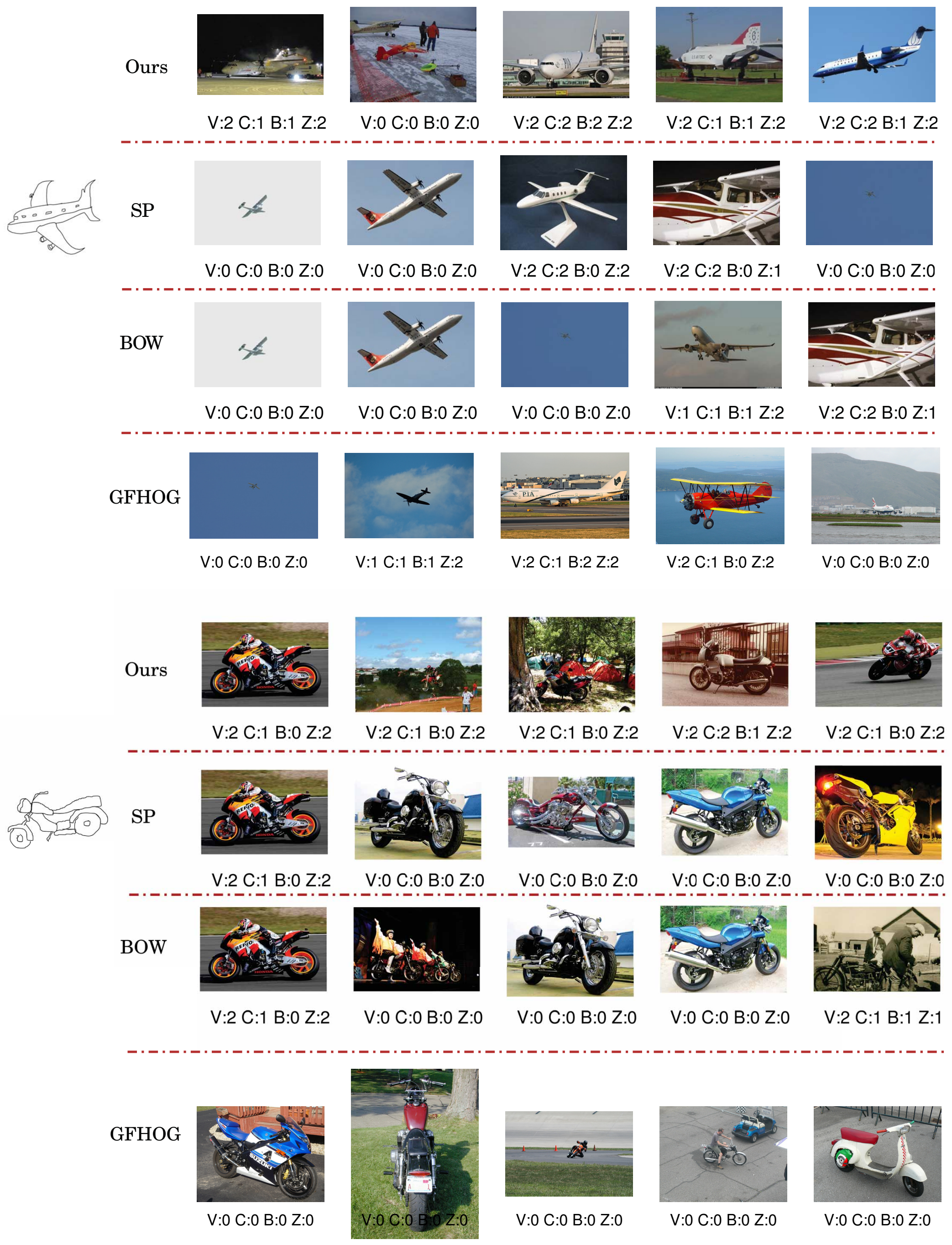


Figure 4.6: Two example retrievals of our method (Ours), spatial pyramid (SP), bag-of-words (BOW) and gradient field HOG (GFHOG). Ground truth similarity is also illustrated with the decomposition of viewpoint (V), configuration (C), body (B) and zoom (Z).

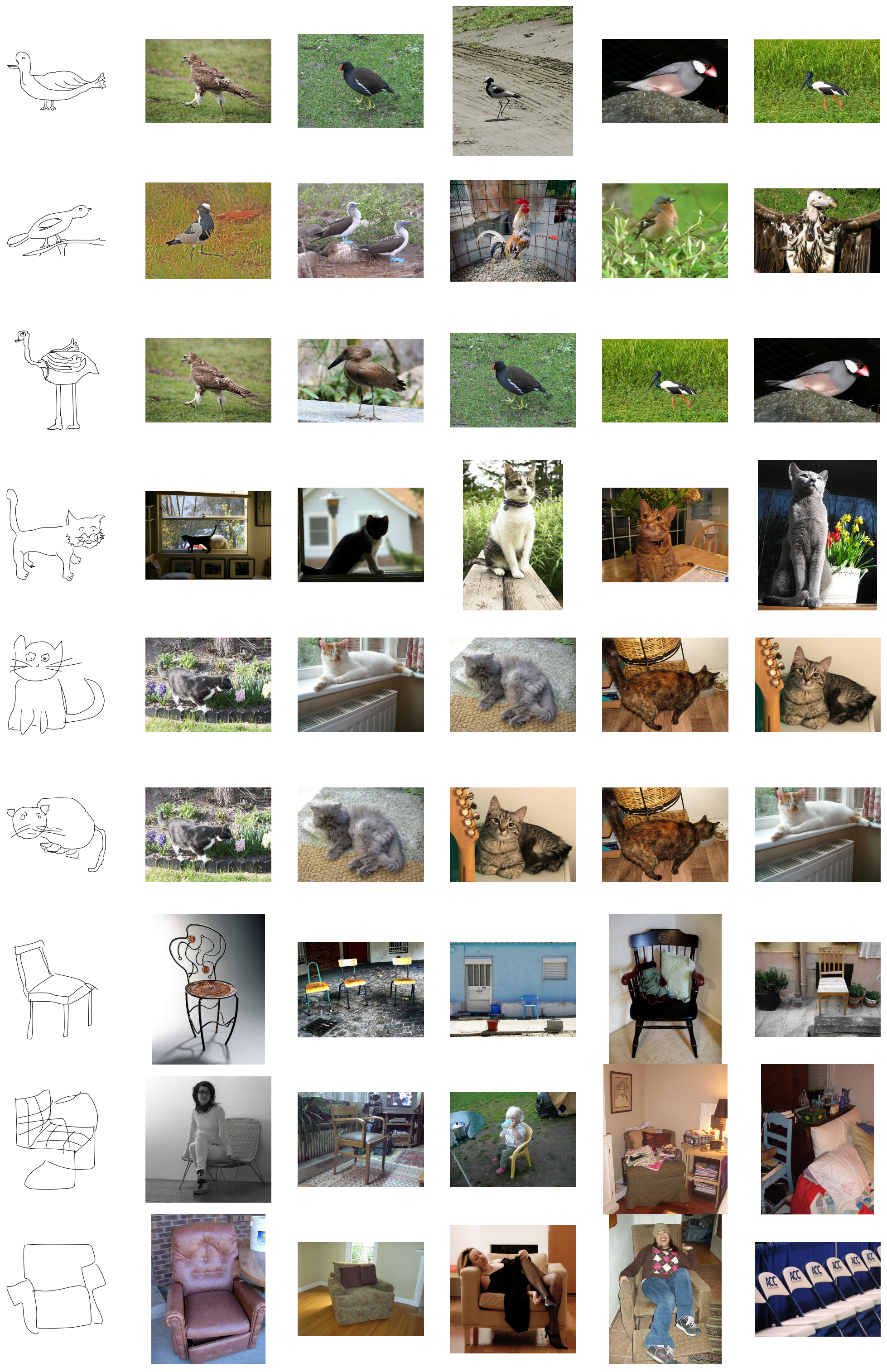


Figure 4.7: Using sketches to retrieval different object variations in the same category (from top to bottom: bird, cat and chair categories).

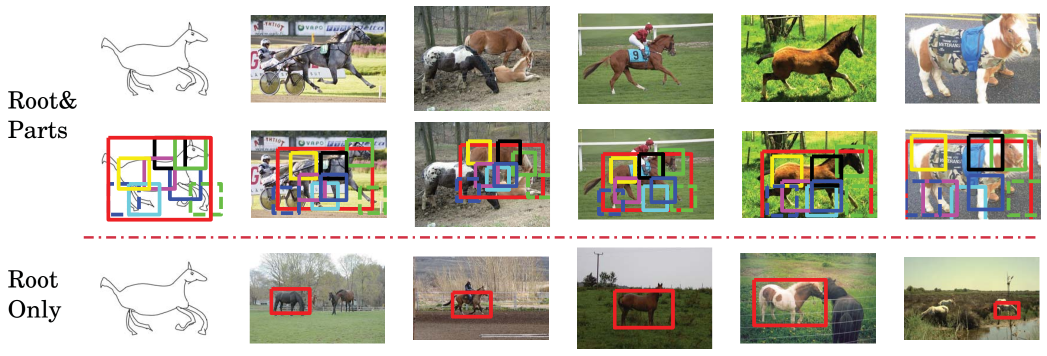


Figure 4.8: Comparison of retrievals using root similarity only (Root Only) and root and part similarities (Root&Parts) in graph matching.

Chapter 5

Sketch Synthesis

As introduced in Section 2.3, previous free-hand style sketch synthesis is restricted to a single category: human faces [8]. The difficulties that impede sketch synthesis to reach other categories include the cluttered edges and diverse deformation variations of the object. To tackle these problems, we propose a deformable stroke model (DSM) which forms the sketch synthesis into a detection process and solves the aforementioned difficulties. Normally, to learn such a deformable model with clearly defined parts, intensive part-level supervision is mandatory. To avoid this tedious work, we propose a perceptual grouping method to segment the sketches into proper semantic parts automatically. An iterative learning scheme is further proposed to refine the learned model gradually. At the first iteration, the initial DSM is learned on the first iteration perceptual grouping results, and the learned DSM will in turn guide the perceptual grouping of the next iteration. The iterations will go on until some convergence criterion is met. During the iterations, duplicated parts or improperly formed parts are largely corrected. Finally, the learned DSM covers almost all the semantic parts of the object without duplication, and appearance variations of each part are well encoded.

The proposed perceptual grouping method tackles the stroke-level sketch segmentation problem. The most advanced stroke-level sketch segmentation work to date [57] uses supervised 3D model with manually selected viewpoint for each input sketch. However, a sufficient number of 3D models may not always be handy and the selection of viewpoint will be tedious for users. So, they make the supervised method impractical for real-life usage. In this chapter, instead, we introduce an unsupervised perceptual grouping method for stroke-level sketch segmentation. It

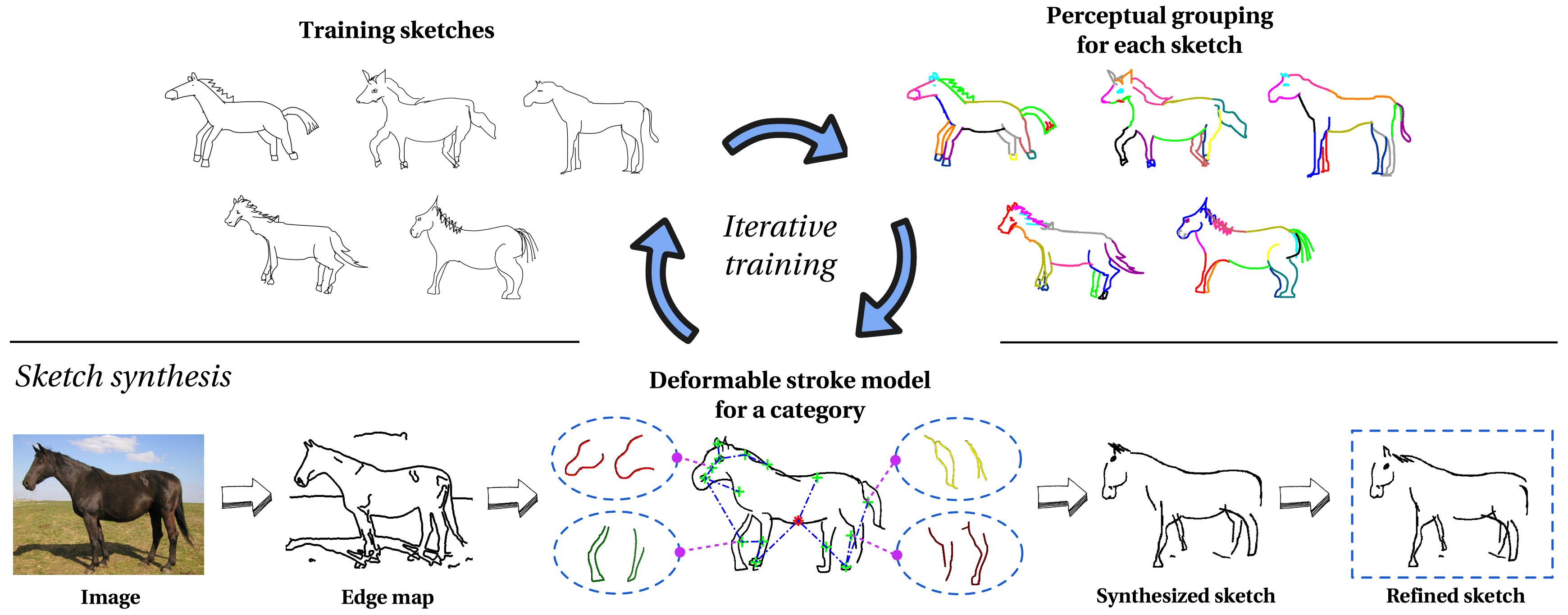


Figure 5.1: An overview of our framework, encompassing deformable stroke model (DSM) learning and free-hand sketch synthesis for given images. To learn a DSM, i) raw sketch strokes are grouped into semantic parts by perceptual grouping (semantic parts are not totally consistent across sketches); ii) category-level DSM is learned upon those semantic parts (category-level semantic parts are summarized and encoded); iii) the learned DSM is used to guide the perceptual grouping in the next iteration until convergence. When the DSM is obtained, we can synthesize sketches for given images, and the synthesized sketches from this model are highly similar to the original images and of a clear free-hand style.

takes into account the relationship between the stroke length and stroke semantics, and utilizes the most semantically meaningful stroke length from observation as guidance to perform segmentation. Unlike the supervised method using a top-down model fitting strategy, a bottom-up greedy grouping algorithm is employed to form the parts gradually. Popular Gestalt principles are also considered, and the local temporal order is used in a similar soft constraint fashion as [57].

To support our usage of stroke length as the criterion to measure the stroke semantics and to reveal the myth of stroke temporal order which is claimed arbitrarily in different works [42, 57], we perform an evident stroke analysis regarding the relationship between the stroke length and the stroke semantics as well as the stroke temporal order on two different sketch dataset: the amateur TU-Berlin dataset and the professional Disney portrait dataset [8]. In the analysis, intuitive results are presented to support our motivations.

Finally, we evaluate our framework via user studies and experiments on two publicly available sketch datasets: (i) six diverse categories of non-expert sketches from the TU-Berlin dataset including: *horse, shark, duck, bicycle, teapot* and *face*, and (ii) professional sketches of two *abstraction levels* (90 seconds and 30 seconds, and we will refer them as 90s and 30s in the rest of this thesis) of two artists in the Disney portrait dataset.

5.1 Stroke Analysis

In this section we perform a full analysis on how stroke-level information can be best used to locate semantic parts of sketches. In particular, we look into (i) the correlation between stroke length and its semantics as an object part, i.e., what kind of strokes do object parts correspond to, and (ii) the reliability of temporal ordering of strokes as a grouping cue, i.e., to what degree can we rely on temporal information of strokes. We conduct our study on both non-expert and professional sketches: (i) six diverse categories from non-expert sketches from the TU-Berlin dataset ([32]) including: *horse, shark, duck, bicycle, teapot* and *face*, and (ii) professional sketches of two *abstraction levels (90s and 30s)* of *artist A* and *artist E* in the Disney portrait dataset ([8]).

Semantics of strokes On the TU-Berlin dataset, we first measure stroke length statistics (quantified by pixel count) of all six chosen categories. Histograms of each category are provided in Figure 5.2. It can be observed that despite minor cross-category variations, distributions are always long-tailed: most strokes being shorter than 1000 pixels, with a small proportion exceeding 2000 pixels. We further divide strokes into 3 groups based on length, illustrated by examples of

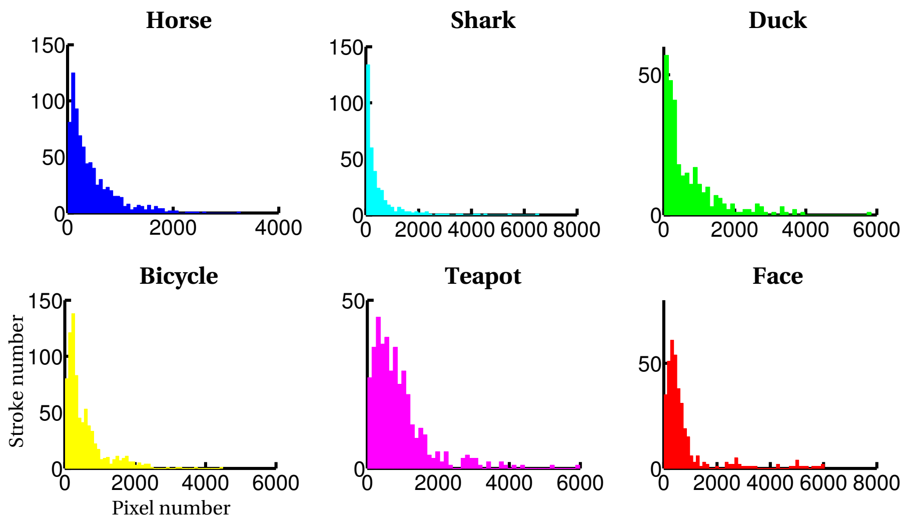
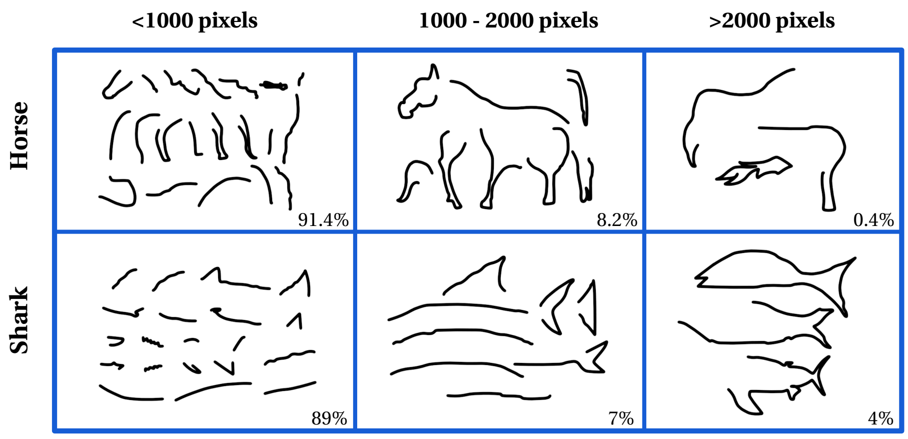


Figure 5.2: Histograms of stroke lengths of six non-expert sketch categories. (x-axis: the size of stroke in pixels; y-axis: number of strokes in the category).

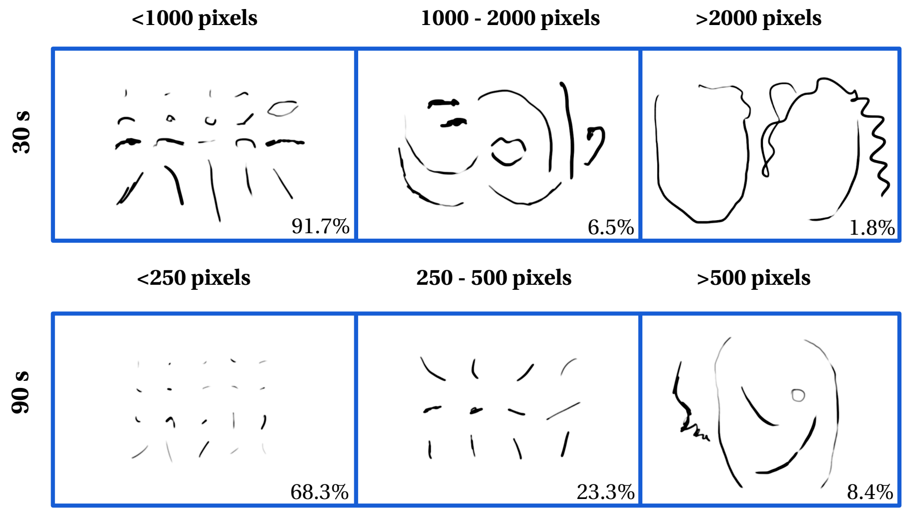
2 categories in Figure 5.3(a). We can see that (i) medium-sized strokes tend to exhibit semantic parts of objects, (ii) the majority of short strokes (e.g., < 1000 pixels) are too small to correspond to a clear part, and (iii) long strokes (e.g., > 2000 pixels) lose clear meaning by encompassing more than one semantic part.

These observations indicate that, ideally, a stroke model can be directly learned on strokes from the medium length range. However, in practice, we further observe that people tend to draw very few medium-sized strokes (length correlates negatively with quantity as seen in Figure 5.2), making them statistically insignificant for model learning. This is apparent when we look at percentages of strokes in each range, shown towards bottom right of each cell in Figure 5.2. We are therefore motivated to propose a perceptual grouping mechanism that counters this problem by grouping short strokes into longer chains that constitute object parts (e.g., towards the medium range in the TU-Berlin sketch dataset). We call the grouped strokes representing semantic parts as semantic strokes. Meanwhile, a cutting mechanism is also employed to process the few very long strokes into segments of short and/or medium length, which can be processed by perceptual grouping afterwards.

On the Disney portrait dataset, a statistical analysis of strokes similar to Figure 5.2 was already conducted by the original authors and the stroke length distributions are quite similar to ours. From example strokes in each range in Figure 5.3(b), we can see for sketches of the 30s level the situation is similar to the TU-Berlin dataset where most semantic strokes are clustered within the middle length range (i.e., $1000 - 2000$ pixels) and the largest group is still the short



(a)



(b)

Figure 5.3: Example strokes of each size group. (a) 2 categories in TU-Berlin dataset. (b) 2 levels of abstraction from artist A in Disney portrait dataset. The proportion of each size group in the given category is indicated in the bottom-right corner of each cell.

strokes. As already claimed in [8] and also reflected in the bottom row of Figure 5.3(b), stroke lengths across the board reduce significantly as abstraction level goes down to 90s. This suggests that, for the purpose of extracting semantic parts, a grouping framework is even more necessary for professional sketches where individual strokes convey less semantic meaning.

Stroke ordering Another previously under-studied cue for sketch understanding is the temporal ordering of strokes, with only a few studies exploring this ([42, 57]). Yet these authors only hypothesized the benefits of temporal ordering without critical analysis a priori. In order to examine if there is a consistent trend in holistic stroke ordering (e.g., if long strokes are drawn first followed by short strokes), we color-code the length of each stroke in Figure 5.4 where: each sketch is represented by a row of colored cells, ordering along the x-axis reflects drawing order, and sketches (rows) are sorted in ascending order of the number of constituent strokes. For ease of interpretation, only 2 colors are used for the color-coding. Strokes with above average length are encoded as yellow and those with below average as cyan.

From Figure 5.4 (1st and 2nd rows), we can see that non-expert sketches with fewer strokes tend to contain a bigger proportion of longer strokes (greater yellow proportion in the upper rows), which matches the claim made by [8]. However, there is not a clear trend in the ordering of long and short strokes across all the categories. Although clearer trend of short strokes following long strokes can be observed in few categories, e.g., *shark* and *face*, and this is due to these categories' contours can be depicted by very few long and simple strokes. In most cases, long and short strokes appear interchangeably at random. Only in the more abstract sketches (upper rows), we can see a slight trend of long strokes being used more towards the beginning (more yellow on the left). This indicates that average humans draw sketches with a random order of strokes of various lengths, instead of a coherent global order in the form of a hierarchy (such as long strokes first, short ones second). In Figure 5.4 (3rd row), we can see that artistic sketches exhibit a clearer pattern of a long stroke followed by several short strokes (the barcode pattern in the figure). However, there is still not a dominant trend that long strokes in general are finished before short strokes. This is different from the claim made by [42], that most drawers, both amateurs and professionals, depict objects hierarchically. In fact, it can also be observed from Figure 5.5 that average people often sketch objects part by part other than hierarchically. However the ordering of how parts are drawn appears to be random.

Although stroke ordering shows no global trend, we found that local stroke ordering (i.e.,

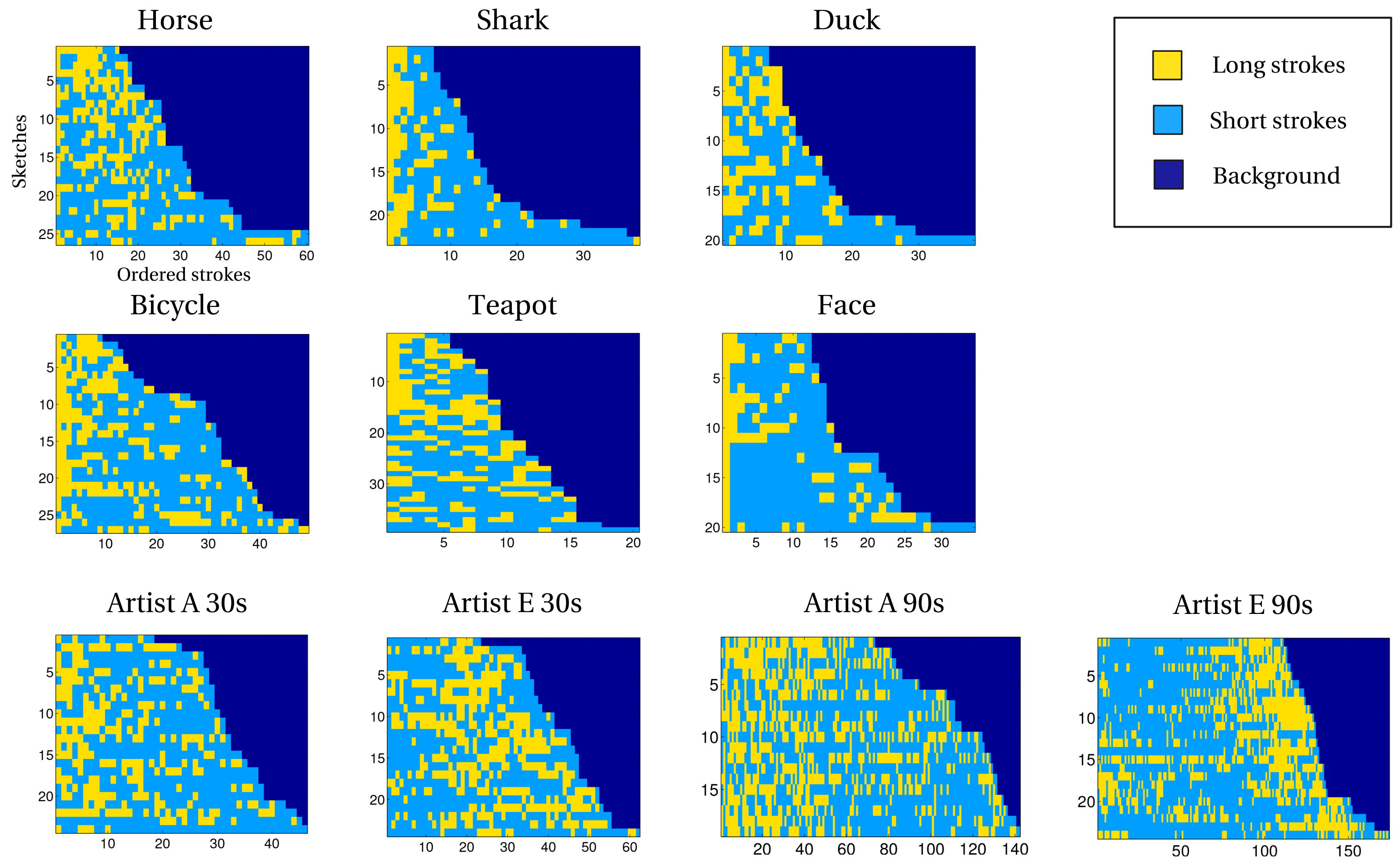


Figure 5.4: Exploration of stroke temporal order. Subplots represent 10 categories: *horse*, *shark*, *duck*, *bicycle*, *teapot* and *face* of TU-Berlin dataset and 30s and 90s levels of *artist A* and *artist E* in Disney portrait dataset. x-axis shows stroke order and y-axis sketch samples, so each cell of the matrices is a stroke. Sketch samples are sorted by their number of strokes (abstraction). Shorter than average strokes are yellow, longer than average strokes are cyan.

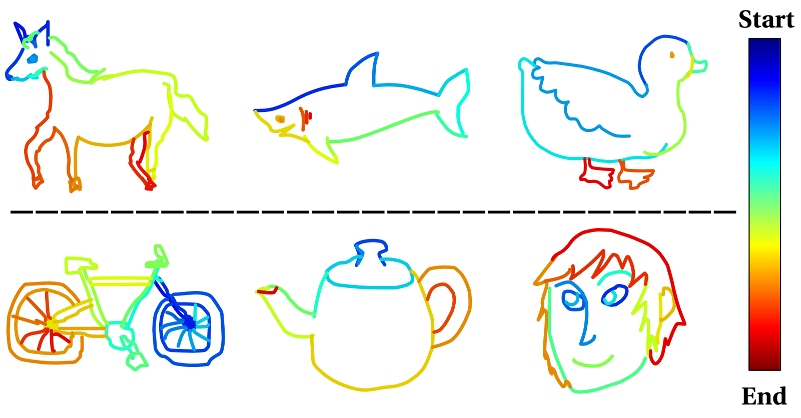


Figure 5.5: Stroke drawing order encoded by color (starts from blue and ends at red). Object parts tend to be drawn with sequential strokes.

strokes depicted within a short timeframe) does possess a level of consistency that could be useful for semantic stroke grouping. Specifically, we observe that people tend to draw a series of consecutive strokes to depict one semantic part, as seen in Figure 5.5. The same hypothesis was also made by [57], but without clear stroke-level analysis beforehand. Later, we will demonstrate via our grouper how local temporal ordering of strokes can be modeled and help to form semantic strokes.

5.2 Deformable Stroke Models

From a collection of sketches of similar poses within one category, we can learn a generative deformable stroke model (DSM). In this section, we first formally define DSM. Then, we introduce the perceptual grouping which groups raw strokes into semantic strokes/parts, and we illustrate how a DSM is learned on those semantic parts and how to use DSM to detect on sketches/images. Finally, the iterative process of performing these three steps interchangeably is well demonstrated with concrete examples.

5.2.1 Model Definition

Our DSM is an undirected graph of n semantic part clusters: $G = (V,E)$. The vertices $V = \{v_1,...,v_n\}$ represent category-level semantic part clusters, and pairs of semantic part clusters are connected by an edge $(v_i,v_j) \in E$ if their locations are closely related. The model is parameterized by $\theta = (u,E,c)$, where $u = \{u_1,...,u_n\}$, with $u_i = \{\mathbf{s}_i^a\}_{a=1}^{m_i}$ representing m_i semantic stroke exemplars of the semantic part cluster v_i; E encodes pairwise part connectivity; and $c = \{c_{ij}|(v_i,v_j) \in E\}$ encodes the spatial relation between connected part clusters. An example *shark* DSM illustration with full part clusters is shown in Figure 5.11 (and a partial example for *horse* is already shown in Figure 5.1), where the green crosses are the vertices V and the blue dashed lines are the edges E. The part exemplars u_i are highlighted in blue dashed ovals.

5.2.2 Sketch segmentation by Perceptual Grouping

Perceptual grouping segments the sketches into *semantic strokes/parts* based on *raw stroke* input. There are many factors that need to be considered in perceptual grouping. As demonstrated in Section 5.1, small strokes need to be grouped to be semantically meaningful, and local temporal order is helpful to decide whether strokes are semantically related. Equally important to the above, conventional perceptual grouping principles (Gestalt principles, e.g. proximity, continuity, similarity) are also required to decide if a stroke set should be grouped. Furthermore, after the first iteration, the learned DSM model is able to assign a group label for each stroke, which can be used in the next grouping iteration.

Algorithmically, our perceptual grouping approach is inspired by [5], who iteratively and greedily group pairs of lines with minimum error. However, their cost function includes only proximity and continuity; and their purpose is line simplification, so grouped lines are replaced by new combined lines. We adopt the idea of iterative grouping but change and expand their error metric to suit our task. For grouped strokes, each stroke is still treated independently, but the stroke length is updated with the group length.

More specifically, for each pair of strokes $\mathbf{s}_1, \mathbf{s}_2$, grouping error is calculated based on 6 aspects: proximity, continuity, similarity, stroke length, local temporal order and model label (only used from second iteration), and the error metric function is defined as:

$$Z(\mathbf{s}_i,\mathbf{s}_j) = (\omega_{pro} * D_{pro}(\mathbf{s}_i,\mathbf{s}_j) + \omega_{con} * D_{con}(\mathbf{s}_i,\mathbf{s}_j) + \omega_{len} * D_{len}(\mathbf{s}_i,\mathbf{s}_j) - \omega_{sim} * B_{sim}(\mathbf{s}_i,\mathbf{s}_j)) \\ * F_{temp}(\mathbf{s}_i,\mathbf{s}_j) * F_{mod}(\mathbf{s}_i,\mathbf{s}_j), \quad (5.1)$$

where proximity D_{pro}, continuity D_{con} and stroke length D_{len} are treated as cost/distance which increase the error, while similarity B_{sim} decreases the error. Local temporal order F_{temp} and model label F_{mod} further modulate the overall error. All the terms have corresponding weights $\{\omega\}$, which make the algorithm cutomizable for different datasets. Detailed definitions and explanations for the 6 terms are as follows (to be noticed, as our perceptual grouping method is an unsupervised and greedy algorithm, the colors for the perceptual grouping results are just for differentiating grouped semantic strokes in individual sketches and have no correspondence between sketches):

Proximity Proximity employs the modified Hausdorff distance (MHD) ([30]) $d_H(\cdot)$ between two strokes, which represents the average closest distance between two sets of edge points. We define $D_{pro}(\mathbf{s}_i,\mathbf{s}_j) = d_H(\mathbf{s}_i,\mathbf{s}_j)/\varepsilon_{pro}$, dividing the calculated MHD with a factor ε_{pro} to control the scale of the expected proximity. Given the image size ϕ and the average semantic stroke number η_{avg} of the previous iteration (the average raw stroke number for the first iteration), we use $\varepsilon_{pro} = \sqrt{\phi/\eta_{avg}}/2$, which roughly indicates how closely two semantically correlated strokes should be located.

Continuity To compute continuity, we first find the closest endpoints **x**,**y** of the two strokes. For the endpoints **x**,**y**, another two points $\mathbf{x}'$,$\mathbf{y}'$ on the corresponding strokes with very close distance (e.g., 10 pixels) to **x**,**y** are also extracted to compute the connection angle. Finally, the continuity is computed as:

$$D_{con}(s_i,s_j) = \|\mathbf{x}-\mathbf{y}\| * (1+angle(\overrightarrow{\mathbf{x}'\mathbf{x}},\overrightarrow{\mathbf{y}'\mathbf{y}}))/\varepsilon_{con},$$

where ε_{con} is used for scaling, and set to $\varepsilon_{pro}/4$, as continuity should have more strict requirement than the proximity.

Stroke length Stroke length cost is the sum of the length of the two strokes: $D_{len}(\mathbf{s}_i,\mathbf{s}_j) = (P(\mathbf{s}_i)+P(\mathbf{s}_j))/\lambda$, where $P(\mathbf{s}_i)$ is the length (pixel number) of raw stroke $\mathbf{s}_i$; or if $\mathbf{s}_i$ is already within a grouped semantic stroke, it is the stroke group length. The normalization factor is computed as $\lambda = \tau * \eta_{sem}$, where η_{sem} is the estimated average number of strokes composing a semantic group in a dataset (from the analysis). When $\eta_{sem} = 1$, τ is the proper length for a stroke to be semantically meaningful (e.g. around 1500 pixels in Figure 5.3(a)), and when $\eta_{sem} > 1$, τ is the maximum length of all the strokes.

The effect of changing λ to control the semantic stroke length is demonstrated in Figure 5.6.

Similarity In some sketches, repetitive short strokes are used to draw texture like hair or mustache. Those strokes convey a complete semantic stroke, yet can be clustered into different groups by continuity. To correct this, we introduce a similarity bonus. We extract strokes $\mathbf{s}_1$ and $\mathbf{s}_2$'s shape context descriptor and calculate their matching cost $K(\mathbf{s}_i,\mathbf{s}_j)$ according to [7]. The similarity bonus is then $B_{sim}(\mathbf{s}_i,\mathbf{s}_j) = exp(-K(\mathbf{s}_i,\mathbf{s}_j)^2/\sigma^2)$ where σ is a scale factor. Examples in Figure 5.7 demonstrate the effect of this term.

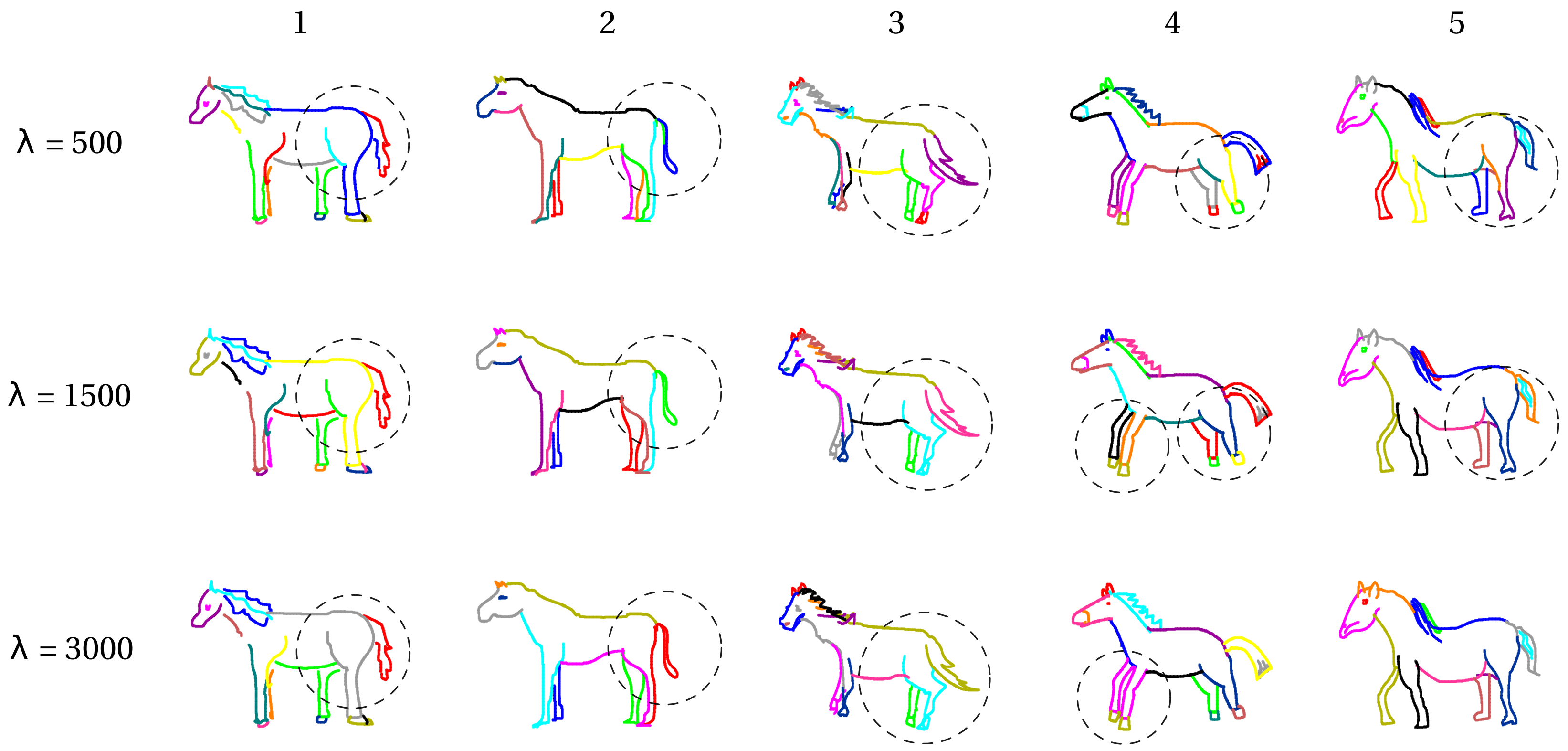


Figure 5.6: The effect of changing λ to control the semantic stroke length (measured in pixels). We can see as λ increases, the semantic strokes' lengths increase as well. And generally speaking, when a proper semantic length is set, the groupings of the strokes are more semantically proper (neither over-segmented or over-grouped). More specifically, we can see that when $\lambda = 500$, many tails and back legs are fragmented. But when $\lambda = 1500$, those tails and back legs are grouped much better. Beyond that, when $\lambda = 3000$, two more semantic parts tend to be grouped together improperly, e.g., one back leg and the tail (column 2), the tail and the back (column 3), or two front legs (column 4). Yet it can also be noticed that when a horse is relatively well drawn (each part is very distinguishable), the stroke length term will influence less, e.g., column 5.

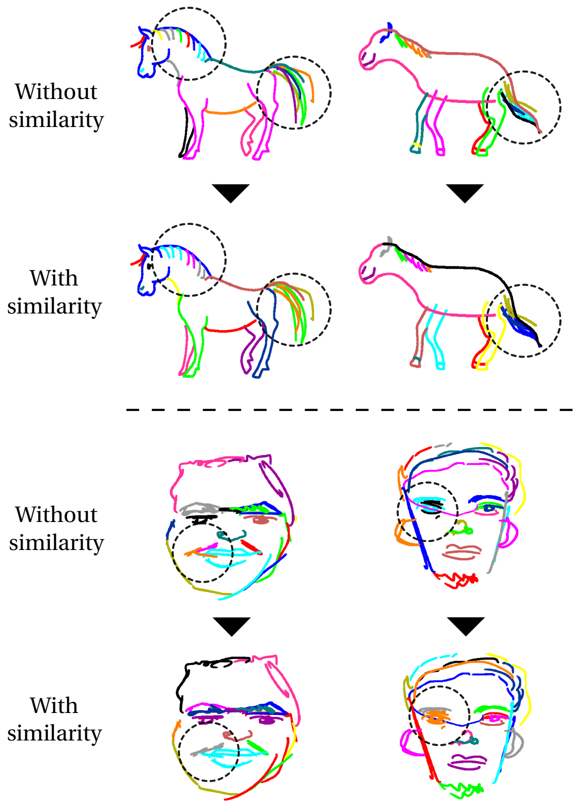


Figure 5.7: The effect of employing the similarity term. Many separate strokes or wrongly grouped strokes are correctly grouped into properer semantic strokes when exploiting similarity.

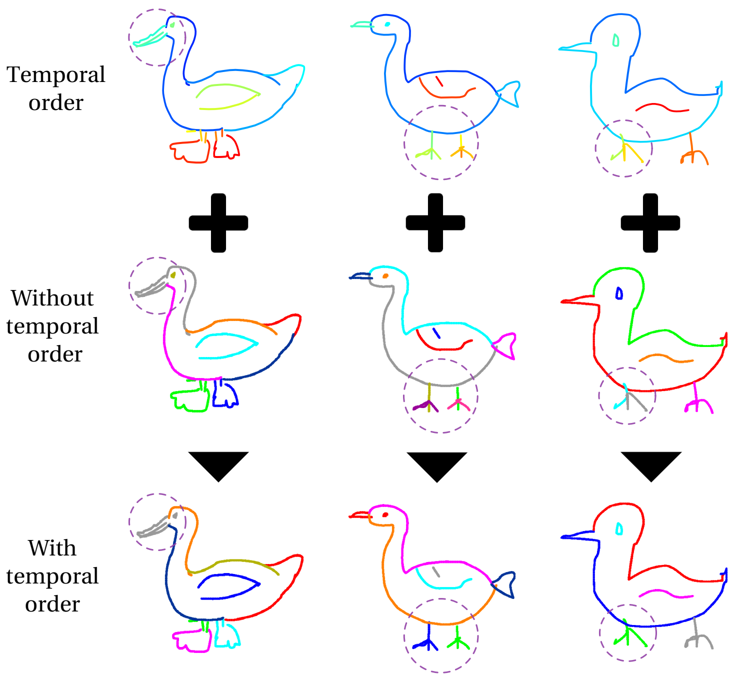


Figure 5.8: The effect of employing stroke temporal order. We can see many errors made to the beak and feet (wrongly grouped with other semantic part or separated into several parts) are corrected as a result.

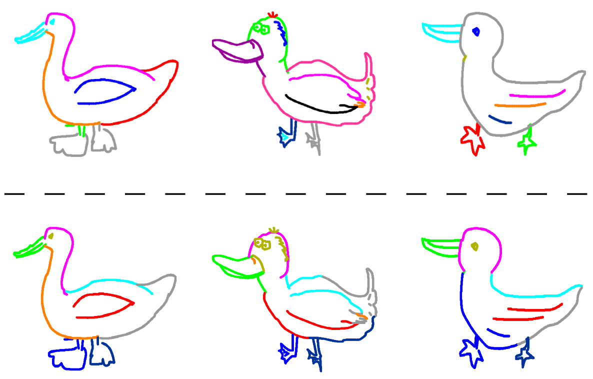


Figure 5.9: The model label after the perceptual grouping of the first iteration. Above: first iteration perceptual groupings. Below: model labels. It can be observed that the first iteration perceptual groupings have different number of semantic strokes, and the divisions over the eyes, head and body are quite different across sketches. However, after a category-level DSM is learned, the model labels the sketches in a very similar fashion, roughly dividing the duck into beak(green), head(purple), eyes(gold), back(cyan), tail(grey), wing(red), belly(orange), left foot(light blue), right foot(dark blue). But some errors still exist in the model label, e.g., missing parts and wrongly labeled part, which will be further corrected in the future iterations.

Local temporal order The local temporal order provides an adjustment factor F_{temp} to the previously computed error $Z(\mathbf{s}_i, \mathbf{s}_j)$ based on how close the drawing orders of the two strokes are:

$$F_{temp}(\mathbf{s}_i, \mathbf{s}_j) = \begin{cases} 1 - \mu_{temp}, & \text{if } |T(\mathbf{s}_i) - T(\mathbf{s}_j)| < \delta. \\ 1 + \mu_{temp}, & \text{otherwise.} \end{cases},$$

where $T(\mathbf{s})$ is the order number of stroke $\mathbf{s}$. $\delta = \eta_{all}/\eta_{avg}$ is the estimated maximum order difference in stroke order within a semantic stroke, where η_{all} is the overall stroke number in the current sketch. μ_{temp} is the adjustment factor. The effect by this term is demonstrated in Figure 5.8.

Model label The DSM model label provides a second adjustment factor according to whether two strokes have the same label or not.

$$F_{mod}(\mathbf{s}_i, \mathbf{s}_j) = \begin{cases} 1 - \mu_{mod}, & \text{if } W(\mathbf{s}_i) == W(\mathbf{s}_j). \\ 1 + \mu_{mod}, & \text{otherwise.} \end{cases}, \tag{5.2}$$

where $W(\mathbf{s})$ is the model label for stroke $\mathbf{s}$, and μ_{mod} is the adjustment factor. The model label obtained after first iteration of perceptual grouping is shown in Figure 5.9.

Pseudo code for our perceptual grouping algorithm is shown in Algorithm 3. More results produced by first iteration perceptual grouping are illustrated in Figure 5.10. As can be seen,

Algorithm 3 Perceptual grouping algorithm

```
1: Input $t$ strokes $\{\mathbf{s}_i\}_{i=1}^{t}$
2: Set the maximum error threshold to $h$
3: for $i, j = 1 \rightarrow t$ do
4:     $ErrorMx(i, j) = Z(\mathbf{s}_i, \mathbf{s}_j)$          ▷ Pairwise error matrix
5: end for
6: while 1 do
7:     $[a, b, minError] = \min(ErrorMx)$
8:                                   ▷ Find $\mathbf{s}_a, \mathbf{s}_b$ with the smallest error
9:     if $minError == h$ then
10:        $break$
11:    end if
12:    $ErrorMx(a, b) \leftarrow h$
13:    if None of $\mathbf{s}_a, \mathbf{s}_b$ is grouped yet then
14:        Make a new group and group $\mathbf{s}_a, \mathbf{s}_b$
15:    else if One of $\mathbf{s}_a, \mathbf{s}_b$ is not grouped yet then
16:        Group $\mathbf{s}_a, \mathbf{s}_b$ to the existing group
17:    else
18:        $continue$
19:    end if
20:    Update $ErrorMx$ cells that are related to strokes in the current group according to the
       new group length
21: end while
22: Assign each orphan stroke a unique group id
```

every sketch is grouped into a similar number of parts, and there is reasonable group correspondence among the sketches in terms of appearance and geometry. However, obvious disagreement also can be observed, e.g., the tails of the sharks are grouped quite differently, as the same to the lips. This is due to the different ways of drawing one semantic stroke that are used by different sketches. And this kind of intra-category semantic stroke variations are further addressed by our iterative learning scheme introduced in Section 5.2.5.

5.2.3 Model Learning

DSM learning is now based on the semantic strokes output by the perceptual grouping step. Putting the semantic strokes from all training sketches into one pool (we use the sketches of mirrored pose to increase the training sketch number and flip them to the same direction), we use spectral clustering ([111]) to form category-level semantic stroke clusters. Semantic strokes in one cluster possess common appearance and geometry characteristics. Subsequently, unlike the conventional pictorial structure/deformable part-based model approach of learning parameters by optimizing on images, we follow contour model methods by learning model parameters from semantic stroke clusters.

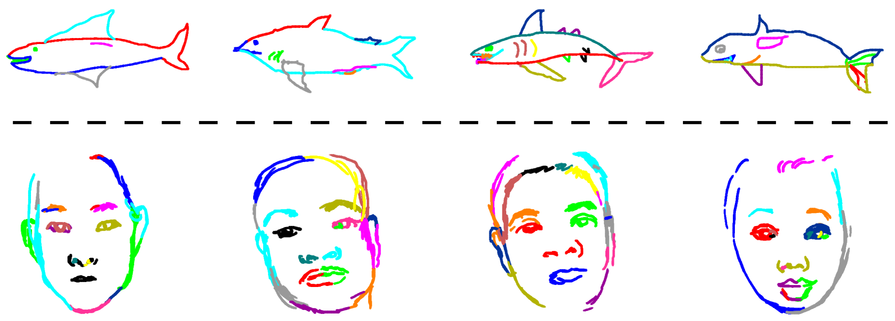


Figure 5.10: Perceptual grouping results. For each sketch, a semantic stroke are represented by one color.

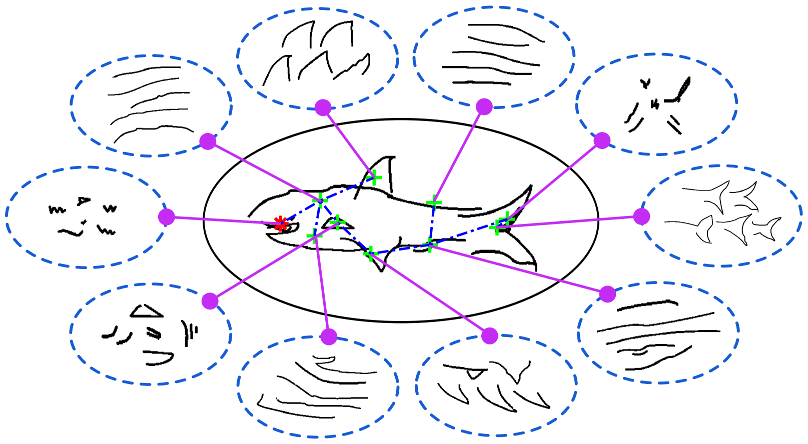


Figure 5.11: An example of *shark* deformable stroke model with demonstration of the part exemplars in all the semantic part clusters (the blue dashed ovals), and the minimum spanning tree structure (the green crosses for tree nodes and the dash-dot lines for tree edges).

A. *Spectral Clustering on Semantic Strokes*

The clustering step forms semantic strokes into semantic stroke clusters, which will be the basic elements of the DSM. We employ spectral clustering, since it takes an arbitrary pairwise affinity matrix as input. Exploiting this, we define our own affinity measure $\mathbf{A}_{ij}$ for semantic strokes $\mathbf{s}_i, \mathbf{s}_j$ whose geometrical centers are $\boldsymbol{l}_i, \boldsymbol{l}_j$ as

$$A_{ij} = exp(\frac{-K(\mathbf{s}_i, \mathbf{s}_j) \cdot \|\boldsymbol{l}_i - \boldsymbol{l}_j\|}{\rho_{\mathbf{s}_i} \rho_{\mathbf{s}_j}}), \tag{5.3}$$

where $K(\cdot)$ is the shape context matching cost and $\rho_{\mathbf{s}_i}$ is the local scale at each stroke $\mathbf{s}_i$ ([111]).

The number of clusters discovered for each category is decided by the mean number of semantic strokes obtained by the perceptual grouper in each sketch. After spectral clustering, in each cluster, the semantic strokes generally agree on the appearance and location. Some cluster examples can be seen in Figure 5.11.

B. Model Parameter Learning

When the semantic stroke clusters are obtained, we need to obtain the parameters θ of the model (exemplars u, connectivity E and spatial relations c) to form the stroke clusters into a functional DSM.

Stroke exemplars We choose the m strokes with the lowest average shape context matching cost to the others in each cluster v_i as the stroke exemplars $u_i = \{\mathbf{s}_i^a\}_{a=1}^{m_i}$ ([88]). The exemplar number m_i is set to a fraction of the overall stroke number in the obtained semantic stroke cluster v_i according to the quality of the training data, i.e., the better the quality, the bigger the fraction. Besides, we augment the stroke exemplars with their rotation variations to achieve more precise fitting. Some learned exemplar strokes of the *shark* category are shown in Figure 5.11.

Spatial Parameters Following the pictorial structure framework ([39]), we treat the spatial parameters (E and c) learning as a maximum likelihood estimation (MLE) problem and assume E forms a minimum spanning tree (MST) structure. However we optimize the parameters on semantic stroke clusters rather than training images. Letting $L_i = \{\boldsymbol{l}_i^a\}_{a=1}^{m_i}$ be the locations of m_i strokes for cluster v_i and $p(L_1,...,L_n|E,c)$ be the probability of the obtained stroke clusters' locations given the model parameters, we get:

$$E^*, c^* = \arg\max_{E,c} p(L_1,...,L_n|E,c). \tag{5.4}$$

As E is assumed to be a tree structure, the probability can be factorized by E:

$$p(L_1,...,L_n|E,c) = \prod_{(L_i,L_j)\in E} p(L_i,L_j|c_{ij}), \tag{5.5}$$

$$p(L_i,L_j|c_{ij}) = \prod_{k=1}^{m_{ij}} p(\boldsymbol{l}_i^k,\boldsymbol{l}_j^k|c_{ij}), \tag{5.6}$$

where k indexes such stroke pairs that one stroke is from cluster v_i and the other from cluster v_j and they are from the same sketch. It can be seen that the spatial relations c_{ij} between two clusters are independent to the edge structure E. Then, we can solve this MLE problem by the following 2 steps.

Learning the Graph Structure To learn such a MST structure for E, we first need to calculate the weights of all the possible connections/edges between the clusters (the smaller the weight, the more closely correlated). We define edge (v_i, v_j)'s weight as:

$$w(v_i, v_j) = \prod_{k=1}^{m_{ij}} \frac{\|\boldsymbol{l}_i^k - \boldsymbol{l}_j^k\|}{max(height, width)}. \tag{5.7}$$

where $height, width$ are the average dimensions of sketches. This metric ensures that stroke clusters are connected to nearby clusters, making the local spatial relations well encoded. Now we can determine the MST edge structure by minimizing

$$E^* = \arg\min_{E} \sum_{(v_i,v_j)\in E} w(v_i, v_j). \tag{5.8}$$

which is solved by Kruskal's algorithm. And the obtained MST is a tree that connects all the vertices and has minimum edge weights.

Spatial Relations After the MST is learned, we can learn the spatial relations of the connected clusters. To obtain relative location parameter c_{ij} for a given edge, we assume that offsets are normally distributed $p(\boldsymbol{l}_i^k, \boldsymbol{l}_j^k | c_{ij}) = \mathcal{N}(\boldsymbol{l}_i^k - \boldsymbol{l}_j^k | \boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij})$. Then MLE result of:

$$(\boldsymbol{\mu}_{ij}^*, \boldsymbol{\Sigma}_{ij}^*) = \arg\max_{\boldsymbol{\mu}_{ij}^*, \boldsymbol{\Sigma}_{ij}^*} \prod_{k=1}^{m_{ij}} \mathcal{N}(\boldsymbol{l}_i^k - \boldsymbol{l}_j^k | \boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}), \tag{5.9}$$

straightforwardly obtains parameters $c_{ij}^* = (\boldsymbol{\mu}_{ij}^*, \boldsymbol{\Sigma}_{ij}^*)$.

The learned model and edge structure is illustrated in Figures 5.1 and 5.11.

5.2.4 Model Matching

As discussed in [39], matching DSM to sketches or images should include two steps: model configuration sampling and configuration energy minimization. Here, we employ fast directional chamfer matching (FDCM, [69]) as the basic operation of stroke registration for these two steps, which is proved both efficient and robust at edge/stroke template matching ([95]). In our framework, automatic sketch model matching is used in both iterative model training and image-sketch synthesis. This section explains this process.

A. Configuration Sampling

A configuration of the model $F = \{(\mathbf{s}_i, \boldsymbol{l}_i)\}_{i=1}^n$ is a model instance registered on an image. In one configuration, exactly one stroke exemplar $\mathbf{s}_i$ is selected in each cluster and placed at location $\boldsymbol{l}_i$. Later, the configuration will be optimized by energy minimization to achieve best balance between (edge map) appearance and (model prior) geometry. Multiple configurations can be sampled, among which the best fitting can be chosen after energy minimization.

To achieve this, on a given image $\mathbf{I}$ and for the cluster v_i, we first sample possible locations for all the stroke exemplars $\{\mathbf{s}_i^a\}_{a=1}^{m_i}$ with FDCM (one stroke exemplar may have multiple possible positions). A sampling region is set based on v_i's average bounding box to increase efficiency, and only positions within this region will be returned by FDCM. All the obtained stroke exemplars and corresponding locations form a set $H_m(v_i) = \{(\mathbf{s}_i^z, \boldsymbol{l}_i^z)\}_{z=1}^{h_i} (h_i \geq m_i)$. For each $(\mathbf{s}_i^z, \boldsymbol{l}_i^z)$, a chamfer matching cost $D_{cham}(\mathbf{s}_i^z, \boldsymbol{l}_i^z, \mathbf{I})$ will also be returned, and only the matchings with a cost under a predefined threshold will be considered by us.

The posterior probability of a configuration F, according to the Bayes's rule, can be formed as:

$$p(F|\mathbf{I}, \theta) \propto p(\mathbf{I}|F, \theta) p(F|\theta), \tag{5.10}$$

Expanding Equation 5.10 on a stroke exemplar basis, we obtain:

$$p(F|\mathbf{I}, \theta) \propto \prod_{i=1}^{n} p(\mathbf{I}|\mathbf{s}_i, \boldsymbol{l}_i) \prod_{(v_i, v_j) \in E} p(\boldsymbol{l}_i, \boldsymbol{l}_j | c_{ij}), \tag{5.11}$$

where $p(\mathbf{I}|\mathbf{s}_i, \boldsymbol{l}_i)$ denotes the appearance fitness for a stroke exemplar and $p(\boldsymbol{l}_i, \boldsymbol{l}_j | c_{ij})$ denotes the spatial relation fitness of two related stroke exemplars.

As the graph edges E forms a MST structure, each node is dependent on a parent node except the root node which is leading the whole tree. Letting v_r denote the root node, C_i denote child nodes of v_i, we can firstly sample the posterior probability $p(\mathbf{s}_r, \boldsymbol{l}_r | \mathbf{I}, \theta)$ for the root, and then sample the probability $p(\mathbf{s}_c, \boldsymbol{l}_c | \mathbf{s}_r, \boldsymbol{l}_r, \mathbf{I}, \theta)$ for its children $\{v_c | v_c \in C_r\}$ until we reach all the leaf nodes. And we can write the marginal distribution for the root as:

$$p(\mathbf{s}_r, \boldsymbol{l}_r | \mathbf{I}, \theta) \propto p(\mathbf{I}|\mathbf{s}_r, \boldsymbol{l}_r) \prod_{v_c \in C_r} S_c(\boldsymbol{l}_r), \tag{5.12}$$

$$S_j(\boldsymbol{l}_i) \propto \sum_{(\mathbf{s}_j, \boldsymbol{l}_j) \in H_m(v_j)} \left(p(\mathbf{I}|\mathbf{s}_j, \boldsymbol{l}_j) p(\boldsymbol{l}_i, \boldsymbol{l}_j | c_{ij}) \prod_{v_c \in C_j} S_c(\boldsymbol{l}_j) \right). \tag{5.13}$$

$p(\boldsymbol{l}_i, \boldsymbol{l}_j | c_{ij})$ is the learned Gaussian offset distribution and $p(\mathbf{I}|\mathbf{s}_i, \boldsymbol{l}_i)$ is computed from the chamfer matching cost: $p(\mathbf{I}|\mathbf{s}_i, \boldsymbol{l}_i) = \exp(-D_{cham}(\mathbf{s}_i, \boldsymbol{l}_i, \mathbf{I}))$.

In computation, the solution for the posterior probability of a configuration F is in a dynamic programming fashion. Firstly, all the S functions are computed once in a bottom-up order from the leaves to the root. Secondly, following a top-down order, we select the top f probabilities $p(\mathbf{s}_r, \boldsymbol{l}_r | \mathbf{I}, \theta)$ for the root with corresponding f configurations $\{(\mathbf{s}_r^b, \boldsymbol{l}_r^b)\}_{b=1}^f$ for the root. For each root configuration $(\mathbf{s}_r^b, \boldsymbol{l}_r^b)$, we then sample a configuration for its children that have the maximum posterior probability, and we continue recursively until we reach the leaves. From this, we obtain f configurations $\{F_b\}_{b=1}^f$ for the model.

B. Energy Minimization

Energy minimization can be considered a refinement for a configuration F according to both appearance and geometry correspondences of the stroke exemplars in the input image. It is solved similarly to configuration sampling with dynamic programming. But instead working with the posterior, it works with the energy function:

$$L^* = \arg\min_L \left(\sum_{i=1}^n D_{cham}(\mathbf{s}_i, \boldsymbol{l}_i, \mathbf{I}) + \sum_{(v_i,v_j)\in E} D_{def}(\boldsymbol{l}_i, \boldsymbol{l}_j) \right), \tag{5.14}$$

where $D_{def}(\boldsymbol{l}_i, \boldsymbol{l}_j) = -\log p(\boldsymbol{l}_i, \boldsymbol{l}_j | c_{ij})$ is the deformation cost between each stroke exemplar and its parent exemplar, and $L = \{\boldsymbol{l}_i\}_{i=1}^n$ are the locations for the selected stroke exemplars in F. The searching space for each $\boldsymbol{l}_i$ is also returned by FDCM. Comparing to configuration sampling, we set a higher threshold for FDCM, and for each stroke exemplar $\mathbf{s}_i$ in F, a new series of locations $\{(\mathbf{s}_i, \boldsymbol{l}_i^k)\}$ are returned by FDCM. And a new $\boldsymbol{l}_i$ is then chosen from those candidate locations $\{\boldsymbol{l}_i^k\}$. To make this solvable by dynamic programming, we define:

$$Q_j(\boldsymbol{l}_i) = \min_{\boldsymbol{l}_j \in \{\boldsymbol{l}_j^k\}} (D_{cham}(\mathbf{s}_j, \boldsymbol{l}_j, \mathbf{I}) + D_{def}(\boldsymbol{l}_i, \boldsymbol{l}_j) + \sum_{v_c \in C_j} Q_c(\boldsymbol{l}_j)), \tag{5.15}$$

And by combining Equations 5.14 and 5.15 and exploit the MST structure again, we can formalize the energy objective function of the root node as:

$$\boldsymbol{l}_r^* = \arg\min_{\boldsymbol{l}_r \in \{\boldsymbol{l}_r^k\}} \left(D_{cham}(\mathbf{s}_r, \boldsymbol{l}_r, \mathbf{I}) + \sum_{v_c \in C_r} Q_c(\boldsymbol{l}_j) \right). \tag{5.16}$$

Through the same bottom-up routine to calculate all the Q functions and the same top-down routine to find the best locations from the root to the leaves, we can find the best locations L^* for all the exemplars. As mentioned before, we sampled multiple configurations and each will have a cost after energy minimization. We choose the one with lowest cost as our final detection result.

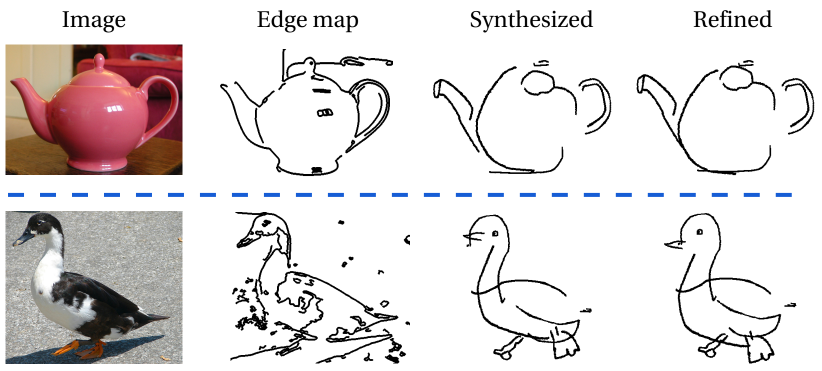


Figure 5.12: Refinement results illustration.

Aesthetic refinement The obtained detection results sometimes will have unreasonable placement for the stroke exemplar due to the edge noise. To correct this kind of error, we perform another round of energy minimization, with appearance terms D_{cham} switched off, and rather than use chamfer matching to select the locations, we let the stroke exemplar to shift around its detection position within a quite small region. Some refinement results are shown for the image-sketch synthesis process in Figure 5.12.

5.2.5 Iterative Learning

As stated before, the model learned with one pass through the described pipeline is not satisfactory – with duplicated and missing semantic strokes. To improve the quality of the model, we introduce an iterative process of: 1) perceptual grouping, 2) model learning and 3) model matching on training data in turns. The learned model will assign cluster labels for raw strokes during detection according to which stroke exemplar the raw stroke overlaps the most with or has the closest distance to. And the model labels are used in the perceptual grouping in the next iteration (Equation 5.2). If an overly-long stroke crosses several stroke exemplars, it will be cut into several strokes to fit the corresponding stroke exemplars.

We employ the variance of semantic stroke numbers at each iteration as convergence metric. Over iterations, the variance decreases gradually, and we choose the semantic strokes from the iteration with the smallest variance to train the final DSM. Figure 5.13(a) demonstrates the convergence process of the semantic stroke numbers during the model training. Different from Figure 5.4, we use 3 colors here to represent the short strokes (cyan), medium strokes (red) and long strokes (yellow). As can be seen in the figure, accompanying the convergence of stroke number variance, strokes are formed into medium strokes with properer semantics as well. Fig-

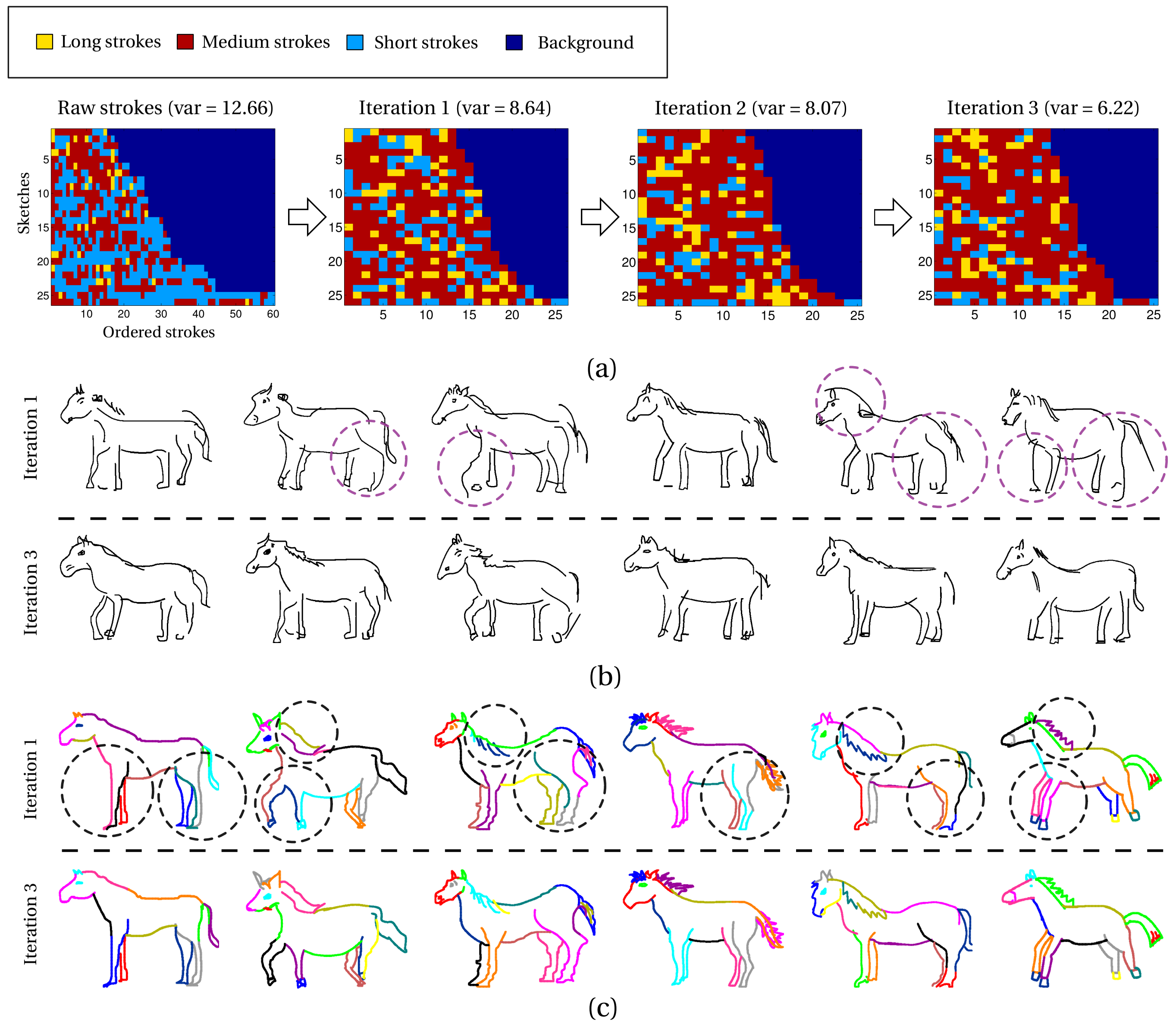


Figure 5.13: The convergence process during model training (horse category): (a) Semantic stroke number converging process (*var* denotes variance); (b) Learned horse models at iteration 1 and 3 (We pick one stroke exemplar from every stroke cluster each time to construct a horse model instance, totally 6 stroke exemplars being chosen and resulting 6 horse model instances); (c) Perceptual grouping results at iteration 1 and 3. Comparing to iteration 1, a much better consensus on the legs and the neck of the horse is observed on iteration 3 (flaws in iteration 1 are highlighted with dashed circles). And this is due to the increased quality of the model of iteration 3, especially on the legs and the neck parts.

ure 5.13(b) illustrates the evolution of the stroke model during the training, and Figure 5.13(c) shows the evolution of the perceptual grouping results.

5.2.6 Image-sketch Synthesis

After the final DSM is obtained from the iterative learning, it can directly be used for image-sketch synthesis through model matching on an image edge map – where we avoid the localization challenge by assuming an approximate object bounding box has been given. Also the correct DSM (category) has to be selected in advance. And these are quite easy to be engineered in practice.

5.3 Experiments

We evaluate our sketch synthesis framework (i) qualitatively by way of showing synthesized results, and (ii) quantitatively via two user studies. We show that our system is able to generate output resembling the input image in plausible free-hand sketch style; and that it works for a number of object categories exhibiting diverse appearance and structural variations.

We conduct experiments on 2 different datasets: (i) TU-Berlin, and (ii) Disney portrait. TU-Berlin dataset is composed of non-expert sketches while Disney portrait dataset is drawn by selected professionals. 10 testing images of each category are obtained from ImageNet, except the face category where we follow [8] to use the Center for Vital Longevity Face Database ([75]). To fully use the training data of the Disney portrait dataset, we did not synthesize face category using images corresponding to training sketches of Disney portrait dataset, but instead selected 10 new testing images to synthesize from. And we normalized the grayscale range of the original sketches to 0 to 1 for the sake of simplifying the model learning process. Specifically, we chose 6 diverse categories from TU-Berlin: *horse*, *shark*, *duck*, *bicycle*, *teapot* and *face*; and the 90*s* and 30*s* abstraction level sketches from *artist A* and *artist E* from Disney portrait (270 level is excluded considering the high computational cost and 15*s* level is due to the presence of many incomplete sketches).

5.3.1 Free-hand Sketch Synthesis Evaluation

In Figure 5.14, we illustrate synthesis results for five categories using models trained on the TU-Berlin dataset. We can see that synthesized sketches are clearly of free-hand style and abstraction while possessing good resemblance to the input images. In particular, (i) major semantic strokes

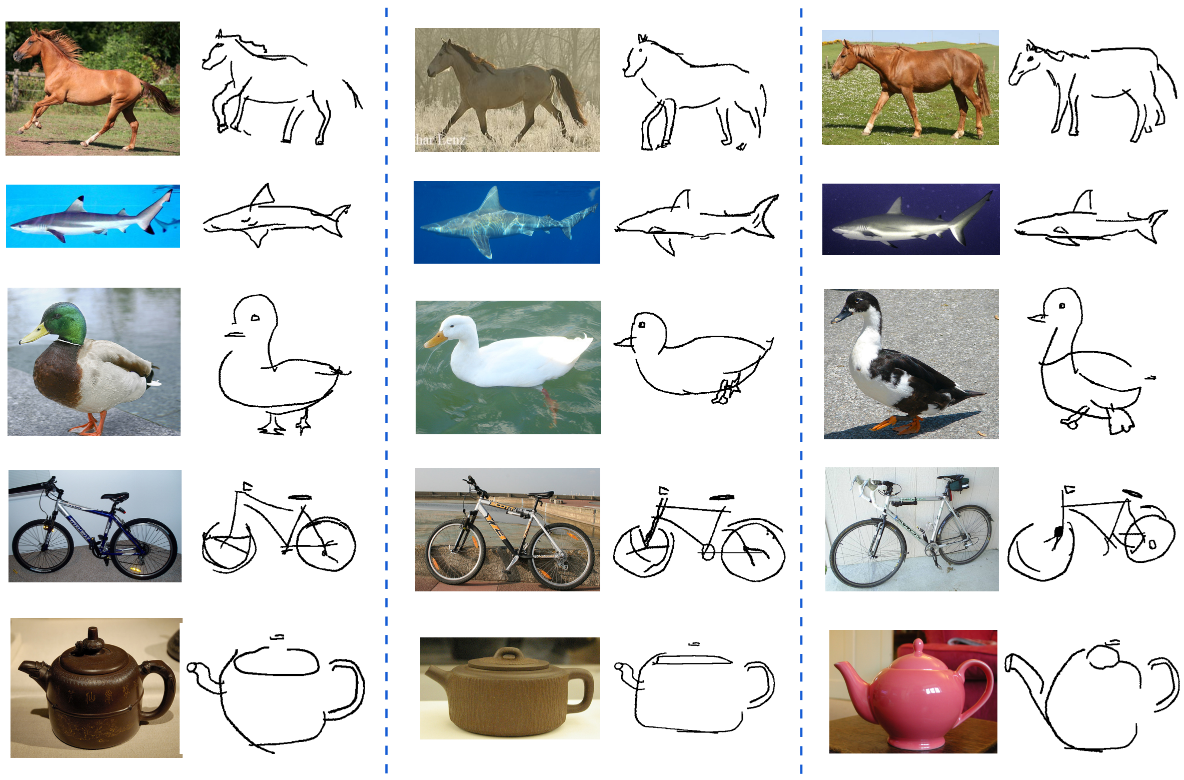


Figure 5.14: Sketch synthesis results of five categories in the TU-Berlin dataset.

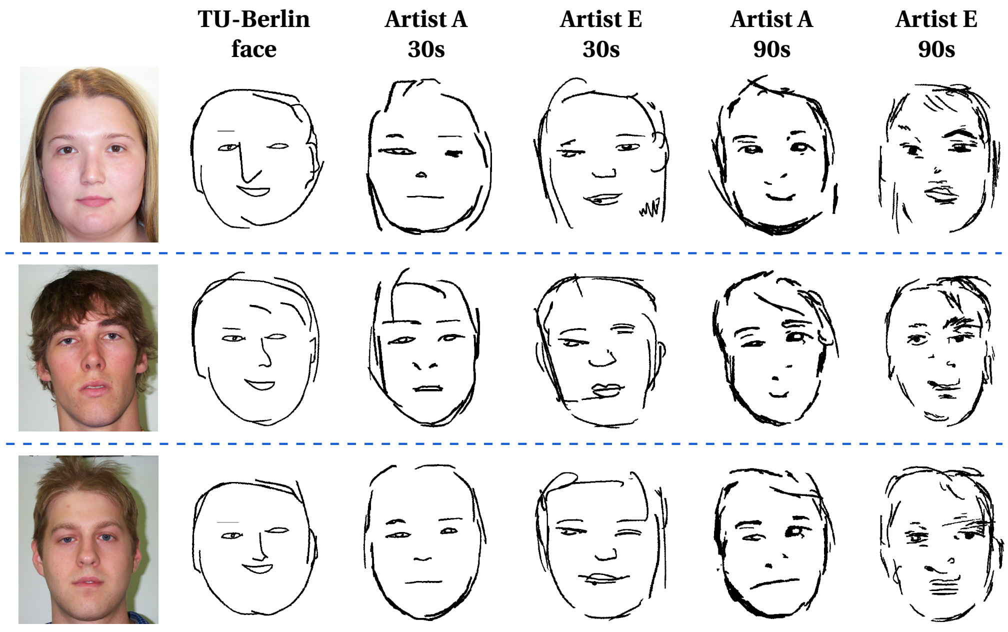


Figure 5.15: A comparison of sketch synthesis results of *face* category using the TU-Berlin dataset and Disney portrait dataset

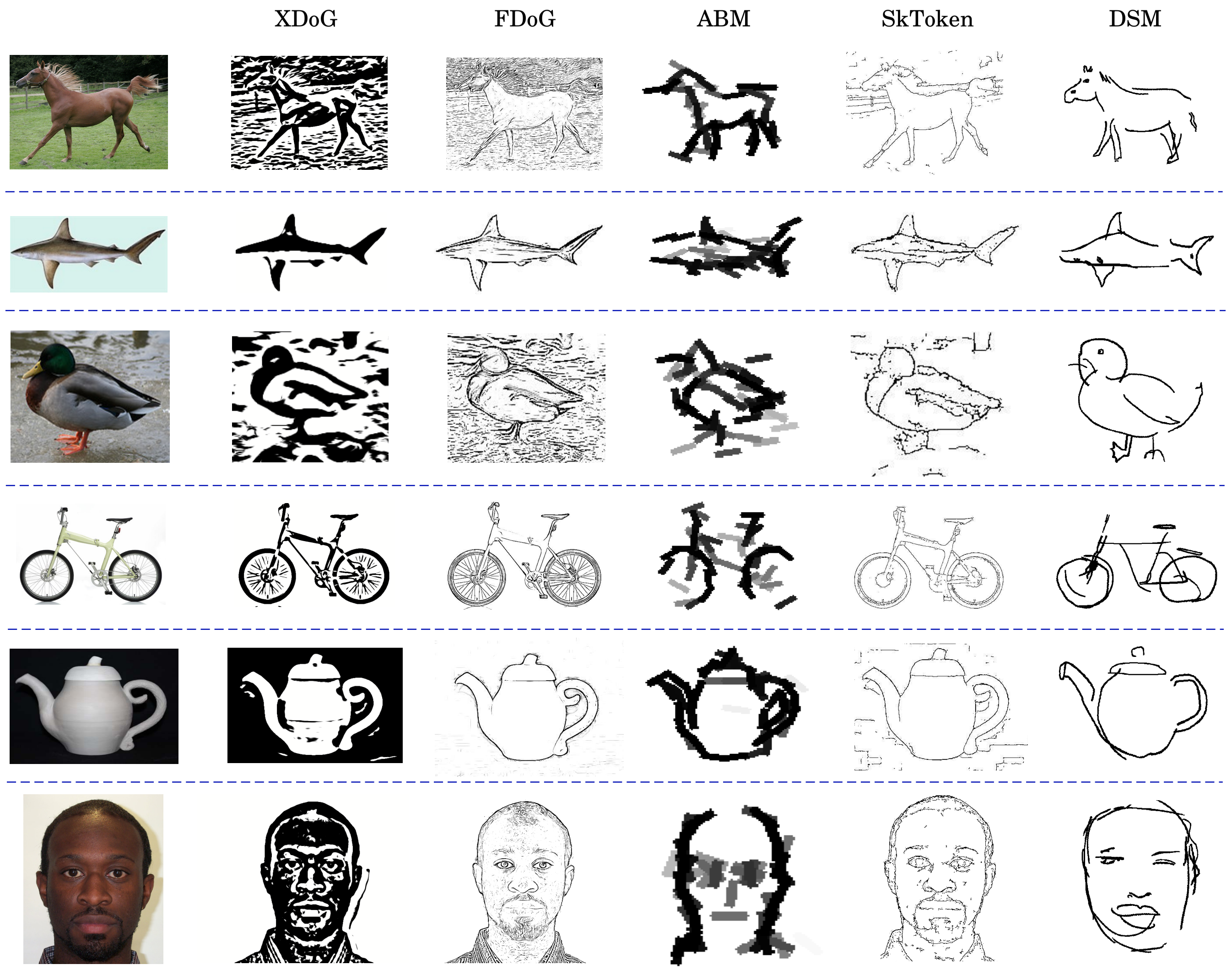


Figure 5.16: Comparison of our DSM to 4 representative works which could also generate sketch-like results, including XDoG [105], FDoG [59], active basis model (ABM) [107] and sketch tokens (SkToken) [112].

are respected in all synthesized sketches, i.e., no missing or duplicated major semantic strokes, (ii) changes in intra-category body configurations are accounted for, e.g., different leg configurations of horses, and (iii) part differences of individual objects are successfully synthesized, e.g., different styles of feet for duck and different body curves of teapots.

Figure 5.15 offers synthesis results for *face* only, with a comparison between these trained on the TU-Berlin dataset and Disney portrait dataset. In addition to the above observations, it can be seen that when professional datasets (e.g., portrait sketches) are used, synthesized faces tend to be more precise and resemble better the input photo. Furthermore, when compared with [8], we can see that although without intense supervision (the fitting of a face-specific mesh model), our model still depicts major facial components with decent precision and plausibility (except for hair which is too diverse to model well), and yields similar synthesized results especially towards more abstract levels (Please refer to [8] for result comparison). We fully acknowledge that the focus of [8] is different as compared to ours, and believe adapting detailed category-specific model alignment supervision could further improve the aesthetic quality of our results, especially towards the less abstract levels.

Figure 5.16 offers a better demonstration of the distinct free-hand style conveyed by our DSM, we select 4 major works that can also generate sketch-like images yet employ different strategies, including XDoG [105], FDoG [59], active basis model (ABM) [107] and sketch tokens [112]. The non-photorealistic animation and rendering (NPAR) works, i.e. XDoG and FDoG, have largely kept the photo details, belonging to both foreground and background. Although the aesthetics of the NPAR results is quite good when the textures of the background and foreground are not too complicated, only moderate abstraction could be expected from the results. Moreover, unpleasant artifacts would be hard to get rid of when complicated texture is present. The ABM could offer more abstract results being closer to sketches and would not keep any background content. In other words, they well simulated the human sketching behavior. However, due to the employment of the Gabor wavelets, the constituent strokes (wavelets) are not similar as natural human strokes, and the level of details is kind of sparse. The sketch tokens results are the closest to human sketches except our DSM results. They possess decent level of abstraction and resemble close characteristics of human strokes. Yet, the affection of background artifacts cannot be totally avoided by them. Distinctly, on the task of free-hand sketch synthesis, our DSM is capable to generate sketch images that have good balance between abstraction and object details and highly resemble the free-hand style almost the same as real free-hand sketches.

5.3.2 Perceptual Study

Two separate user studies were performed to quantitatively evaluate our synthesis results. We employed 10 different participants for each perceptual study (to avoid prior knowledge), making a total of 20. The first user study is on sketch recognition, in which humans are asked to recognize synthesized sketches. This study confirms that our synthesized sketches are semantic enough to be recognizable by human. The second one is on perceptual similarity rating, where human subjects are asked to link the synthesized sketches to their corresponding images. By doing this, we demonstrate the intra-category discrimination power of our synthesized sketches.

Sketch recognition Sketches synthesized using models trained on TU-Berlin dataset are used in this study, so that human recognition performance reported in [32] can be used as comparison. There are 60 synthesized sketches in total, with 10 per category. We equally assign 6 sketches (one from each category) to every participant and ask them to select an object category for each sketch (250 categories are provided in a similar scheme as in [32], thus chance is 0.4%). From Table 5.1, we can observe that our synthesized sketches can be clearly recognized by humans, in some cases offering 100% accuracy. It can be further noted that human recognition performance on our sketches follows a very similar trend across categories to that reported in [32]. The overall higher performance of ours is most likely due to the much smaller scale of our study. The result of this study clearly shows that our synthesized sketches convey enough semantic meaning and are highly recognizable as human-drawn sketches.

Table 5.1: Recognition rate of human users for (S)ynthesised and (R)eal sketches ([32]).

	Horse	**Shark**	**Duck**	**Bicycle**	**Teapot**	**Face**
S	100%	40%	100%	100%	90%	80%
R	86.25%	60%	78.75%	95%	88.75%	73.75%

Image-sketch similarity For the second study, both TU-Berlin dataset and Disney portrait dataset are used. In addition to the 6 models from TU-Berlin, we also included 4 models learned using the 90*s* and 30*s* level sketches from *artist A* and *artist E* from Disney portrait dataset. For each category, we randomly chose 3 image pairs, making 30 pairs (3 pairs $\times$ 10 categories) in total for each participant. Each time, we show the participant one pair of images and their corresponding synthesized sketches, where the order of sketches may be the same or reversed as the image order (Due to the high abstraction nature of the sketches, only a pair of sketch is used and two corresponding images are provided for clues each time). Please refer to Figure 5.14 to see some example image and sketch pairs. The participant is then asked to decide if the sketches are of the same order as the images. We consider a choice to be correct if the participant correctly identified the right ordering. Finally, the accuracy for each category is averaged over 30 pairs and summarized in Table 5.2. A binomial test is applied to the results, and we can see that, except *duck* and *Artist E 90s*, all the rest results are significantly better than random guess (50%), with most $p < 0.01$. The relatively weaker performance for *duck* and *teapot* from TU-Berlin is mainly due to a lack of training sketch variations as opposed to image domain, resulting in the model failing to capture enough appearance variations in images. On Disney portrait dataset, matching accuracy is generally on the same level as TU-Berlin, yet there appears to be a big divide on *artist E 90s*. This is self-explanatory when one compares synthesized sketches of the 90s level from *artist E* (last column of Figure 5.15) with other columns – *artist E 90s* seems to depict a lot more short and detailed strokes making the final result relatively messy. In total, we can see that our synthesized sketches possess sufficient intra-category discrimination power.

Table 5.2: Image-sketch similarity rating experiment results.

	Horse	**Shark**	**Duck**	**Bicycle**	**Teapot**
Acc.	86.67%	73.33%	63.33%	83.33%	66.67%
p	< 0.01	< 0.01	0.10	< 0.01	< 0.05
	Face	**A 30s**	**E 30s**	**A 90s**	**E 90s**
Acc.	76.67%	76.67%	90.00%	73.33%	56.67%
p	< 0.01	< 0.01	< 0.01	< 0.01	0.29

5.3.3 Parameter Tuning

Our model is intuitive to tune, with important parameters constrained within perceptual grouping. There are two sets of parameters affecting model quality: semantic stroke length and weights for different terms in Equation 5.1. Semantic stroke length reflects negatively to the semantic stroke number and it needs to be tuned consistent with the statistical observation of that category. And it is estimated as the λ illustrated in the *stroke length* term in Section 5.2.2. For η_{sem} we used 1-3 for TU-Berlin dataset and the $30s$ level portrait sketches, and for the $90s$ level portrait sketches, η_{sem} is set 8 and 11 respectively for the *90s* level of *artist A* and *artist E*. This is because in the less abstracted sketches artists tend to use more short strokes to form one semantic stroke. For those categories with $\eta_{sem} = 1$, we found 85%-95% of the maximum stroke length is a good range to tune against for τ since our earlier stroke-level study suggests semantic stroke strokes tend to cluster within this range (see Figure 5.2).

Regarding weights for different terms in Equation 5.1, we used the same parameters for both the TU-Berlin dataset and 30s level portrait sketches, and set ω_{pro}, ω_{con} and ω_{len} (for proximity, continuity and stroke length respectively) uniformly to 0.33. For the 90s level sketches, again since too many short strokes are used, we switched off the continuity term, and set ω_{pro} and ω_{len} both to 0.5. The weight ω_{sim} and adjustment factors μ_{temp} and μ_{mod} (corresponding to similarity, local temporal order and model label) are all fixed as 0.33 in all the experiments.

5.4 Discussions

We presented a free-hand sketch synthesis system that for the first time works outside of just one object category. Our model is data-driven and uses publicly available sketch datasets regardless of whether drawn by non-experts or professionals. With minimum supervision, i.e., the user selects a few sketches of similar poses from one category, our model automatically discovers common semantic parts of that category, as well as encoding structural and appearance variations of those parts. By fitting our model to an input image, we automatically generate a free-hand sketch that shares close resemblance to that image. Results provided in the previous section confirms the efficacy of our model. Some key issues of our model are discussed more as follows:

Data alignment: Although our model can address a good amount of variations in the number, appearance and location of parts without the need for well-aligned datasets, a poor model may be learned if the topology diversity (existence, number and layout of parts) of the training sketches is too extreme. This could be alleviated by selecting fine-grained sub-categories of sketches to train on, which would require more constrained collection of training sketches.

Model quality: Due to the unsupervised nature of our model, it has difficulty modelling challenging objects with complex inner structure. For example, buses often exhibit complicated features such as the number and location of windows. We expect that some simple user interaction akin to that used in interactive image segmentation might help to increase model precision, for example by asking the user to scribble an outline to indicate rough object parts.

Another weakness of our model is that the diversity of synthesized results is highly dependent on training data. If there are no similar sketches in the training data that can roughly resemble the input image, it will be hard to generate a good looking free-hand sketch for that image, e.g., some special shaped teapot images. We also share the common drawback of part-based models, that severe noise will affect detection accuracy.

Aesthetic quality: In essence, our model learns a normalized representation for a given category. However, apart from common semantic strokes, some individual sketches will exhibit unique parts not shared by others, e.g., saddle of a horse. To explicitly model those accessory parts can significantly increase the descriptive power of the stroke model, and thus is an interesting direction to explore in the future. Last but not least, as the main aim of this work is to tackle the modeling for category-agnostic sketch synthesis, only very basic aesthetic refinement post-processing was employed. A direct extension of current work will be therefore leveraging advanced rendering techniques from the NPAR domain to further enhance the aesthetic quality of our synthesized sketches.

Chapter 6

Conclusions and Future Work

The studying of more sophisticated sketch datasets has profoundly advanced the free-hand sketch domain. We are glad to be part of it and demonstrate our contributions with this thesis. After the full details of our works in the previous chapters, we hereby offer our summary on the three applications and look into the promising future directions.

6.1 Sketch Recognition

By addressing the internal structure complexity and visual cue sparsity challenges, our proposed star graph representation and multiple kernel learning method have greatly advanced the previous state-of-the-art sketch recognition performance. As a fast pacing area, recent state-of-the-art has been moved forward again by deep learning empowered Sketch-a-Net [110] to 74.9% accuracy as opposed to 73.1% of human accuracy. Respecting the deep learning's superb power, this result may temporally put a period to the sketch recognition race, except some evaluation can be done on the pruned TU-Berlin dataset proposed by Schneider and Tuytelaars [85], where human accuracy is 100%. In this kind of situation, the future works on sketch recognition can focus more on useful applications, especially in the area of preschool education for children. Integrating sketch input into existing preschool education software can make this kind of software more easily accessible for children and increase their interest of using the software to study the world.

6.2 Fine-grained Sketch-based Image Retrieval

In Section 2.2.1, we have summarized the essential understandings for the previous SBIR works. Here, we update them based on the research outcomes of Chapter 4.

i)The applying scenarios As demonstrated by the experiment results of Chapter 4, intra-category fine-grained SBIR focusing on objects is a more promising and worthwhile direction. We can see that with the help of sketches, more customized retrieval could be realized with intuitive sketching rather than composing loads of texts. In the future, this kind of fine-grained SBIR can be promoted to various online sale websites, helping the users to retrieve the desired commodities existing in their minds.

ii)The representations The DPM representation has provided a mid-level abstraction for the objects, which enhances the local descriptor based representation at addressing holistic abstraction/distortions. In the future, how to elaborate this representation to cope with more object variations, to exploit the most discriminative parts and to alleviate the computational cost, are all interesting directions to proceed.

iii)The dataset and the evaluation metric The currently proposed dataset is still quite challenging for the state-of-the-art object detectors to work properly. And if this step cannot be solved well, the afterwards comparisons are not built on a very solid ground. Unfortunately, so far, object detection is still a very challenging area being actively researched. Therefore, commodity datasets, like shoe dataset and furniture dataset, that contain sufficient object variations but have no background clutters are more suitable both to evaluate the core function of fine-grained SBIR systems and to test real-life scenarios. The evaluation metric should still follow the human ranked image list evaluation strategy. More desirably, crowdsourcing practices like employing the Amazon Mechanical Turk, should be adopted to obtain large-scale sketch-image pair similarity ratings for statistically significant evaluation and for long-term evaluation.

6.3 Sketch Synthesis

The demonstrated deformable stroke model has shown promising results in real free-hand style sketch synthesis for general categories. The perceptual grouping method is also quite practical for part-level sketch segmentation due to its unsupervised nature. Two direct extensions for the deformable stroke model work are 1) enhancing the aesthetic quality of the synthesized sketches and 2) introducing simple guidance for the part-level sketch segmentation to improve the segmentation quality. Improving the stroke connectivity is the most straightforward way to enhance aesthetic quality. And the EZ-Sketching work [92] should be a good place to seek inspirations from, which refines user-drawn sketch strokes. Regarding the simple guidance for sketch segmentation, stick-man-like iconic sketches that indicate the desired topology of some category are a good choice. Producing these iconic sketches is easy for the users, and these iconic sketches will help a lot when segmenting some sketch categories with complicated topology, i.e. with intersecting parts. In the long run, simulating professional artists' sketching styles to produce more complicated and aesthetically valuable sketching works from given realistic photos could be the developing direction for sketch synthesis. Hopefully one day, some algorithm can be a famous artist enthusiastically welcomed by the general public.

Bibliography

[1] Z. Almeraj, B. Wyvill, T. Isenberg, A. A. Gooch, and R. Guy. Automatically mimicking unique hand-drawn pencil lines. 33(4):496–508, 2009.

[2] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems*, pages 561–568, 2003.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011.

[4] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *International Conference on Machine Learning*, page 6, 2004.

[5] P. Barla, J. Thollot, and F. X. Sillion. Geometric clustering for line drawing simplification. In *Eurographics Symposium on Rendering*, 2005.

[6] F. I. Bashir, A. A. Khokhar, and D. Schonfeld. Real-time motion trajectory-based indexing and retrieval of video sequences. *IEEE Transactions on Multimedia*, 9(1):58 – 65, 2007.

[7] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[8] I. Berger, A. Shamir, M. Mahler, E. Carter, and J. Hodgins. Style and abstraction in portrait sketching. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):55:1–55:12, 2013.

[9] A. D. Bimbo and P. Pala. Visual image retrieval by elastic matching of user sketches. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):121–132, 1997.

[10] A. D. Bimbo, P. Pala, and S. Santini. Visual image retrieval by elastic deformation of object sketches. In *IEEE Symposium on Visual Languages*, pages 216–223, 1994.

[11] A. D. Bimbo, P. Pala, and S. Santini. Visual image retrieval by elastic deformation of object sketches. In *IEEE International Conference on Multimedia Computing and Systems*, pages 215–218, 1996.

[12] O. Boiman and M. Irani. Detecting irregularities in images and in video. *International Journal of Computer Vision*, 74(1):17–31, 2007.

[13] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[14] Y. Cao, C. Wang, Z. Li, L. Zhang, and L. Zhang. Spatial-bag-of-features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3352–3359, 2010.

[15] Y. Cao, C. Wang, L. Zhang, and L. Zhang. Edgel index for large-scale sketch-based image search. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–768, 2011.

[16] Y. Cao, H. Wang, C. Wang, Z. Li, L. Zhang, and L. Zhang. Mindfinder: interactive sketch-based image search on millions of images. In *International Conference on Multimedia*, pages 1605–1608, 2010.

[17] A. Chalechale, G. Naghdy, and A. Mertins. Sketch-based image matching using angular partitioning. *IEEE Transactions on Systems, Man, and Cybernetics*, 35(1):28–41, 2005.

[18] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[19] S.-f. Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong. VideoQ: An automated content based video search system using visual cues. In *In Proceedings of ACM Multimedia*, pages 313–324, 1997.

[20] Y. Chans, Z. Lei, D. P. Lopresti, and S.-Y. Kung. A feature-based approach for image retrieval by sketch. In *SPIE International Symposium on Voice, Video and Data Communications*, pages 220–231, 1997.

[21] K. Chatfield, J. Philbin, and A. Zisserman. Efficient retrieval of deformable shape classes using local self-similarities. In *Workshop on Non-rigid Shape Analysis and Deformable*

Image Alignment, IEEE International Conference on Computer Vision, pages 264–271, 2009.

[22] H. Chen, N. Zheng, L. Liang, Y. Li, Y. Xu, and H. Shum. Pictoon: A personalized image-based cartoon system. In *ACM International Conference on Multimedia*, pages 171–178, 2002.

[23] M. Cho, K. Alahari, and J. Ponce. Learning graphs to match. In *IEEE International Conference on Computer Vision*, pages 25–32, 2013.

[24] M. Cho, J. Lee, and K. M. Lee. Reweighted random walks for graph matching. In *European Conference on Computer Vision*, pages 492–505, 2010.

[25] M. Cho and K. Lee. Progressive graph matching: Making a move of graphs via probabilistic voting. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 398–405, 2012.

[26] J. P. Collomosse, G. McNeill, and Y. Qian. Storyboard sketches for content based video retrieval. In *IEEE International Conference on Computer Vision*, pages 245 – 252, 2009.

[27] C. Cortes and V. Vapnik. Support-vector networks. *Maching Learning*, 20(3):273–297, 1995.

[28] J. Dai, Y. Wu, J. Zhou, and S. Zhu. Cosegmentation and cosketch by unsupervised learning. In *IEEE International Conference on Computer Vision*, 2013.

[29] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[30] M.-P. Dubuisson and A. K. Jain. A modified hausdorff distance for object matching. In *International Conference on Patter Recognition*, pages 566–568, 1994.

[31] O. Duchenne, A. Joulin, and J. Ponce. A graph-matching kernel for object categorization. In *IEEE International Conference on Computer Vision*, pages 1792–1799, 2011.

[32] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):44:1–44:10, 2012.

[33] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. An evaluation of descriptors for large-scale image retrieval from sketched feature lines. *Computers & Graphics*, 34(5):482–498, 2010.

[34] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa. Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1624–1636, 2011.

[35] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html.

[36] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html.

[37] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3-4):231–262, 1994.

[38] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[39] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

[40] V. Ferrari, F. Jurie, and C. Schmid. From images to shape models for object detection. *International Journal of Computer Vision*, 87(3):284–303, 2010.

[41] M. Fonseca, A. Ferreira, and J. Jorge. Sketch-based retrieval of complex drawings using hierarchical topology and geometry. *Computer-Aided Design*, 41(12):1067–1081, 2009.

[42] H. Fu, S. Zhou, L. Liu, and N. J. Mitra. Animated construction of line drawings. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(6):133:1–133:10, 2011.

[43] Y. Fu, T. Hospedales, T. Xiang, and S. Gong. Learning multi-modal latent attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(2):303–316, 2013.

[44] K. Fukunage and P. M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE Transactions on Computers*, 24(7):750–753, 1975.

[45] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *The Journal of Machine Learning Research*, 12:2211–2268, 2011.

[46] B. Gooch, E. Reinhard, and A. Gooch. Human facial illustrations: Creation and psychophysical evaluation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 23(1):27–44, 2004.

[47] C.-e. Guo, S.-C. Zhu, and Y. N. Wu. Primal sketch: Integrating structure and texture. *Computer Vision and Image Understanding*, 106(1):5–19, 2007.

[48] K. Hirata and T. Kato. Query by visual example - content based image retrieval. In *International Conference on Extending Database Technology: Advances in Database Technology*, pages 56–71, 1992.

[49] J.-W. Hsieh, S.-L. Yu, and Y.-S. Chen. Motion-based video retrieval by trajectory matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(3):396 – 409, 2006.

[50] N. Hu, R. Rustamov, and L. Guibas. Graph matching with anchor nodes: A learning approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2906–2913, 2013.

[51] R. Hu, M. Barnard, and J. Collomosse. Gradient field descriptor for sketch based retrieval and localization. In *IEEE International Conference on Image Processing*, pages 1025–1028, 2010.

[52] R. Hu and J. Collomosse. Motion-sketch based video retrieval using a trellis levenshtein distance. pages 121–124, 2010.

[53] R. Hu and J. Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *Computer Vision and Image Understanding*, 117:790–806, 2013.

[54] R. Hu, S. James, and J. Collomosse. Annotated free-hand sketches for video retrieval using

object semantics and motion. In *International Conference on MultiMedia Modeling*, pages 473–484, 2012.

[55] R. Hu, S. James, T. Wang, and J. Collomosse. Markov random fields for sketch based video retrieval. In *Proceedings of ACM Conference on International Conference on Multimedia Retrieval*, pages 279–286, 2013.

[56] R. Hu, T. Wang, and J. Collomosse. A bag-of-regions approach to sketch based image retrieval. In *IEEE International Conference on Image Processing*, pages 3661–3664, 2011.

[57] Z. Huang, H. Fu, and R. W. H. Lau. Data-driven segmentation and labeling of freehand sketches. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 33(6):175:1–175:10, 2014.

[58] G. Humphrey. The psychology of the gestalt. *Journal of Educational Psychology*, 15(7):401–412, 1924.

[59] H. Kang, S. Lee, and C. K. Chui. Coherent line drawing. In *Proceedings of the International Symposium on Non-photorealistic Animation and Rendering*, pages 43–50, 2007.

[60] T. Kato, T. Kurita, N. Otsu, and H. Kyoji. A sketch retrieval method for full color image database-query by visual example. In *IAPR International Conference on Computer Vision and Applications*, pages 530–533, 1992.

[61] H. Knutsson. Representing local structure using tensors. In *Scandinavian Conference on Image Analysis*, pages 244–251, 1989.

[62] C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 951 – 958, 2009.

[63] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2169–2178, 2006.

[64] L. Li, H. Su, Y. Lim, and L. Fei-Fei. Objects as attributes for scene classification. In *European Conference on Computer Vision*, pages 57–69, 2010.

[65] Y. Li, Y. Song, and S. Gong. Sketch recognition by ensemble matching of structured features. In *British Machine Vision Conference*, pages 35.1–35.11, 2013.

[66] L. Liang, H. Chen, Y. Xu, and H. Shum. Example-based caricature generation with exaggeration. In *Pacific Conference on Computer Graphics and Applications*, pages 386–393, 2002.

[67] J. Lim, C. Zitnick, and P. Dollr. Sketch tokens: A learned mid-level representation for contour and object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3158 – 3165, 2013.

[68] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3337–3344, 2011.

[69] M. Liu, O. Tuzel, A. Veeraraghavan, and R. Chellappa. Fast directional chamfer matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1696–1703, 2010.

[70] D. G. Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision, 20–25 September, 1999, Kerkyra, Corfu, Greece, Proceedings*, volume 2, pages 1150–1157, 1999.

[71] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[72] S. Matusiak, M. Daoudi, T. Blu, and O. Avaro. Sketch-based images database retrieval. In *Workshop on Multimedia Information Systems*, pages 185–191, 1998.

[73] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005.

[74] G. Miller. Wordnet: A lexical database for english. *COMMUNICATIONS OF THE ACM*, 38(11):39–41, 1995.

[75] M. Minear and D. Park. A lifespan database of adult facial stimuli. *Behavior Research Methods, Instruments, & Computers.*, 36(4):630–633, 2004.

[76] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *IEEE International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009.

[77] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. H. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The qbic project: Querying images by content, using color, texture, and shape. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 173–187, 1993.

[78] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *European conference on Computer Vision*, pages 575–588, 2006.

[79] F. Orabona and L. Jie. Ultra-fast optimization algorithm for sparse multi kernel learning. In *International Conference on Machine Learning*, pages 249–256, 2011.

[80] S. Parui and A. Mittal. Similarity-invariant sketch-based image retrieval in large databases. In *European Conference on Computer Vision*, pages 398–414, 2014.

[81] F. Perronnin and C. R. Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[82] Y. Qi, J. Guo, Y. Li, H. Zhang, T. Xiang, and Y. Song. Sketching by perceptual grouping. In *International Conference on Image Processing*, pages 270–274, 2013.

[83] R. K. Rajendran and S.-F. Chang. Image retrieval with sketches and compositions. In *IEEE International Conference on Multimedia and Expo*, pages 717–720, 2000.

[84] M. P. Salisbury, S. E. Anderson, R. Barzel, and D. H. Salesin. Interactive pen-and-ink illustration. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, pages 101–108, 1994.

[85] R. G. Schneider and T. Tuytelaars. Sketch classification and classification-driven analysis using fisher vectors. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 33(6):174:1–174:9, 2014.

[86] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[87] C.-B. Shim and J.-W. Chang. Efficient similar trajectory-based retrieval for moving objects in video databases. In *Proceedings of the International Conference on Image and Video Retrieval*, pages 163–173, 2003.

[88] J. Shotton, A. Blake, and R. Cipolla. Multiscale categorical object recognition using contour fragments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1270–1281, 2008.

[89] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, pages 1470–1477, 2003.

[90] Y. Song, C. Li, L. Wang, P. Hall, and P. Shen. Robust visual tracking using structural region hierarchy and graph matching. *Neurocomputing*, 89:12–20, 2012.

[91] P. Sousa and M. Fonseca. Sketch-based retrieval of drawings using spatial proximity. *Journal of Visual Language and Computing*, 21(2):69–80, 2010.

[92] Q. Su, W. H. A. Li, J. Wang, and H. Fu. Ez-sketching: Three-level optimization for error-tolerant image tracing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*, 33(4):54:1–54:9, 2014.

[93] M. Sun and S. Savarese. Articulated part-based model for joint object detection and pose estimation. In *IEEE International Conference on Computer Vision*, pages 723–730, 2011.

[94] Z. Sun, C. Wang, L. Zhang, and L. Zhang. Free hand-drawn sketch segmentation. In *European Conference on Computer Vision*, pages 626–639, 2012.

[95] A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla. shape context and chamfer matching in cluttered scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 127–133, 2003.

[96] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):815–830, 2010.

[97] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[98] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

[99] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *IEEE International Conference on Computer Vision*, pages 606–613, 2009.

[100] C. Wang, Z. Li, and L. Zhang. Mindfinder: image search by interactive sketching and tagging. In *International Conference on World Wide Web*, 2010.

[101] S. Wang, L. Zhang, Y. Liang, and Q. Pan. Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2216–2223, 2012.

[102] X. Wang and X. Tang. Face photo-sketch synthesis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):1955–1967, 2009.

[103] C. wen Su, H. yuan Mark Liao, H. rong Tyan, C. wen Lin, D. yu Chen, and K. chin Fan. Motion flow-based video retrieval. *IEEE Transactions on Multimedia*, 9(6):1193–1201, 2007.

[104] G. Winkenbach and D. H. Salesin. Computer-generated pen-and-ink illustration. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, pages 91–100, 1994.

[105] H. Winnemöller, J. E. Kyprianidis, and S. C. Olsen. XDoG: An extended difference-of-gaussians compendium including advanced image stylization. *Computers & Graphics*, 36(6):740–753, 2012.

[106] L. Wiskott, J.-M. Fellous, N. Kuiger, and C. Von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):775–779, 1997.

[107] Y. N. Wu, Z. Si, H. Gong, and S.-C. Zhu. Learning active basis model for object detection and recognition. *International Journal of Computer Vision*, 90(2):198–235, 2010.

[108] A. Yamada, M. Pickering, S. Jeannin, L. Cieplinski, R. Ohm, Jens, and M. Kim. Mpeg-7 visual part of experimentation model version 8.0, 2000.

[109] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1385 – 1392, 2011.

[110] Q. Yu, Y. Yang, Y. Song, T. Xiang, and T. Hospedales. Sketch-a-net that beats humans. In *British Machine Vision Conference*, pages 7.1–7.12, 2015.

[111] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Conference on Neural Information Processing Systems*, pages 1601–1608, 2004.

[112] Q. Zhu, G. Song, and J. Shi. Sketch tokens: A learned mid-level representation for contour and object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[113] J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Computing Surveys*, 38(2):6.1–6.56, 2006.