Outline Tracing from Sketches

WONG, Ka Wing

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Philosophy

in

Computer Science and Engineering

The Chinese University of Hong Kong

September 2012

Abstract of thesis entitled:

Sketching is the earliest stage of production in art and design. Sketches are useful in conveying and developing ideas. However, raw sketches contain unnecessary strokes and must be converted to neat and tidy drawings before moving onto later stage of production. Traditionally, this conversion process is time-consuming and tedious since it is performed stroke by stroke manually. The situation is even worse when it comes to animation production which involves a huge number of sketches, so there is a strong motivation to automate the conversion process. Existing works formulate the conversion process as stroke grouping and curve fitting processes, in which close and continuous strokes are grouped together to form single strokes in the resulting image. Nevertheless, previous works overlooked an important law of visual perception: the law of closure in Gestalt principles. Gestalt principles concluded from early visual perception studies demonstrate how human perceive visual elements as different groups of lines and shapes. It states that we tend to group elements into closed shape even when a gap exists. In this thesis, we utilize the idea of law of closure and propose a region-based approach to refine sketches. Experiment result shows that this method outperforms the existing methods in terms of the capability of preserving salient regions in sketches.

草圖繪製是創意產業最早期的一個工序。研究發現，草圖不但表達了畫家的想法，在繪製的過程，草圖也能為畫家帶來新靈感，所以草圖繪製是不可缺少的工序。然而，原始的草圖包含許多不必要的筆觸，所以進行後期製作前，必須用線重新勾畫。傳統上，這個過程是費時和繁瑣的，畫家必須人手把每一條需要的線都重新勾畫出來。當應用到動畫創作的時候，由於涉及的草圖數量龐大，情況會變得更壞，所以有必要把整個過程自動化。現有的研究都視草圖勾勒為一個線條分組和曲線擬合的過程，他們會把相近而順暢的筆觸組合在一起，形成單一線條。然而，他們都忽略了視覺感知上的一個重要法則——格式塔理論中的閉合原理。格式塔理論是一個知名的心理學理論，解釋人類如何透過整合理解各種視覺元素。根據格式塔理論中的閉合原理，我們往往把各種分隔的視覺元素整合為一個封閉的形狀。在這篇論文中，我提出閉合原理比格式塔理論中的其他原理更能幫助我們理解草圖，從而提出了一種基於區域的方法來勾勒草圖。實驗結果發現，我的方法在保有草圖上封閉形狀的能力上比現有的方法更優勝。

Submitted by Wong Ka Wing

for the degree of Master of Philosophy in Computer Science and Engineering

at the Chinese University of Hong Kong in July 2012

Table of Contents

1. Introduction

In entertainment industry such as comics or animation production, and technical design work such as architectural design or product design, artists and designers draw sketches to present their ideas. These sketches are usually rough and contain unnecessary lines, and thus are unsuitable for direct usage at later stage of production. Therefore, an additional outline tracing process is needed to produce neat and tidy drawings. Currently, this process is usually done manually without using software tools. This is a tedious and time-consuming task which should be automated to save manpower and increase efficiency.

When we refine sketches manually, we analyze the sketch region by region where top-down information about the shape and form are utilized. In contrast, typical computational approach analyzes sketches in a bottom-up stroke-by-stroke approach. *Figure 1* demonstrates the idea. If we are given only a portion of strokes in a sketch, such as *Figure 1 (a)*, it is difficult to determine whether we should group any of the strokes. However, if these strokes form a closed region (or shape) such as *Figure 1 (b)* and *(d)*, it gives us extra information of how the strokes should be grouped. Therefore, closed regions are important information in sketches which help us analyze and refine the sketches. This observation is further supported by the law of closure in Gestalt principles of human visual perception, which is first suggested by Max Wertheimer in 1923 [16]. According to the law of closure, we tend to group elements into closed shape even when there is a gap in between.

With the above observations, we propose a region-based approach to refine sketches in this work. As the first sketch refinement algorithm utilizing the law of closure, we suggest a new approach to analyze sketches and the main contribution in this work is the retrieval of multi-scale regions from sketches.

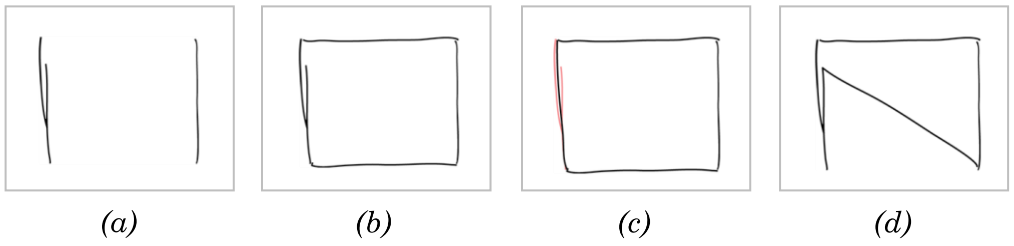


Figure 1: ***Sketch Analysis.*** *Without any closed region or shapes in the sketch, it is difficult to determine whether to group the strokes in the figure. Once extra strokes are added to (a) to form close regions as in (b), we can then decide we should group strokes in (b) to form (c) but not (d).*

This thesis is organized as follows. In Section 2, background knowledge and pros and cons of related works are described. In Section 3, gestalt principles are introduced and an in-depth study of how exactly these principles can aid the outline tracing process is explained. Section 4 gives the implementation details of the proposed method. Section 5 shows the experiment results followed by some discussions and limitations of our method. Summary and conclusion are written in Section 6.

2. Background

2.1 Role of Sketch in Art and Design

Sketches help artists to convey and present their ideas to their audiences. A number of studies [1][2] show that sketches can indeed help artists to develop new ideas by continuously reinterpreting the strokes when drawing sketches. This makes sketching a very important stage which cannot be replaced by digital or other forms of drafting.

In the old days when computers were not very common in the creative industry, sketches were usually drawn on paper. After drawing sketches, artists traced the outlines on another paper based on the sketches. The traced result was then used in the subsequent procedures such as colorization.

With the advance of technology, different drawing and animating software tools were introduced, such as Photoshop, Illustrator and Flash. More and more artists and designers prefer to work on computers instead. To make traditional on-paper sketches suitable for digital processing, artists has to scan the sketches into the computer using scanners and obtain a raster image. However, the scanning process may introduce noises to the raster image and lots of strokes information, such as the continuity or order, is lost in the process.

Later, drawing tablets were introduced. These drawing tablets can record the position, pressure and orientation of the stylus, providing similar drawing experience as pencil. The input from the tablets is in vector form which can be modified easily for the use of later stages. Therefore artists or designers usually use drawing tablets to draw sketches nowadays.

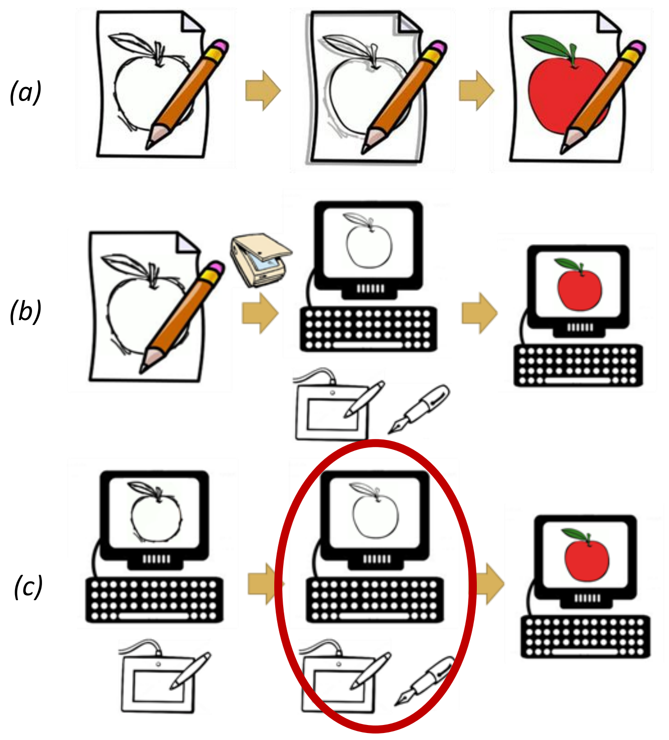


Figure 2: ***Three common workflows of early creative production.*** *(a) Traditional workflow in which all the work, including sketching, outline tracing and colorization, is performed on paper. (b) Workflow in which the outline tracing and colorization processes are performed in computer. (c) The modern workflow where all the processing work is performed in computer. In this thesis, we aim to automate the circled process.*

The three common workflows of early creative production are summarized in *Figure 2*. In this thesis, we follow the third workflow and automate the outline tracing process based on the strokes input by the user with drawing tablet.

2.2 Characteristics of Raw Sketches

Sketches have their own unique characteristics which make them different from natural images or other non-photorealistic images. These characteristics include small color range, formation of one single stroke combined by multiple strokes, presence of unnecessary strokes, imprecise stroke and shading.

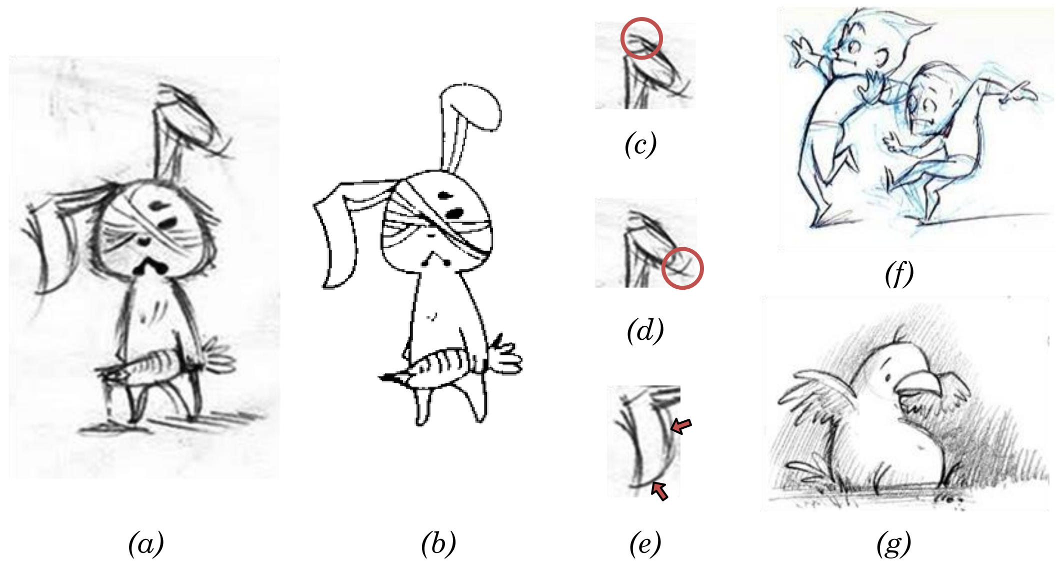


Figure 3: ***Characteristics of Sketches.*** *(a)(b) A raw grayscale sketch and its processed form. (c) An example from (a) showing strokes that are clipped to obtain the final image. (d) An example from (a) showing the presence of unwanted strokes. (e) An example showing the formation of one single strokes from multiple strokes. (f) A sketch with 2 colors. Core lines are represented by black strokes. (g) A sketch with shading*

Small Color Range Sketches have small color range. A typical sketch is drawn in gray scale as shown in *Figure 3 (a)*. Some other sketches may have multiple colors.

In this case, one of these colors represents the core lines of the sketches. One of these examples is given in *Figure 3 (f)* and the core lines are outlined in black.

Formation of One Single Stroke from Multiple Strokes By comparing the raw sketch and the processed image in *Figure 3*, we can observe that in some areas of the sketch, multiple strokes are grouped together to form a single stroke (*Figure 3 (e)*).

Presence of Unwanted Lines To facilitate the generation of ideas and shapes, sketches usually contains a number of strokes that are eventually excluded in the final image. *Figure 3 (d)* demonstrates an example of this.

Presence of Imprecise Stroke Sketching is a fast and imprecise process, thus the length of strokes is usually inaccurate. Sometimes, only part of a stroke is used to form the final image.

Presence of Shading Some artists or designers draw shading in their sketches to represent the background or dark areas, as shown in *Figure 3 (g)*.

In this thesis, we are going to evaluate and compare the performance of existing works and my proposed method regarding to these characteristics.

2.3 Related Works

As discussed in the previous section, sketches can be raster images or vector images depending on the input method we use. In this section, we will discuss different existing methods for the outline tracing process. These methods can be classified into two main categories, methods that are suitable for raster images and methods that are suitable for vector images.

2.3.1 Methods for Raster Image

When a sketch is drawn on paper and scanned into the computer, we obtain a raster image of the sketch. A number of methods which detect edges or centerline are developed from image processing research. Some of them are highlighted and discussed below.

Traditional Edge Detection Traditional edge detection techniques focus on locating the position of edges in raster images. These techniques include the first derivative methods and second derivative methods.

Some examples of the first derivative methods are Roberts Cross Edge Detection [3], Sobel Edge Detection [4] and Compass Edge Detection [5]. The core idea of these methods is that edges are located at places where there is a change in image intensity. However, using first derivative methods can result in double edges for thick strokes which are undesirable. Besides, these methods are sensitive to noise which is unavoidable during the scanning process of sketches.

Second derivative methods make use of the second derivative of the image intensity. Although these methods can avoid the double-edge problem, second derivative methods are still sensitive to noise. Therefore, further derived methods such as first derivative of Gaussian and the second derivative of Gaussian are used [6]. Both of them use the Gaussian function to smooth the image first before applying edge detection [7] and both are less sensitive to noise.

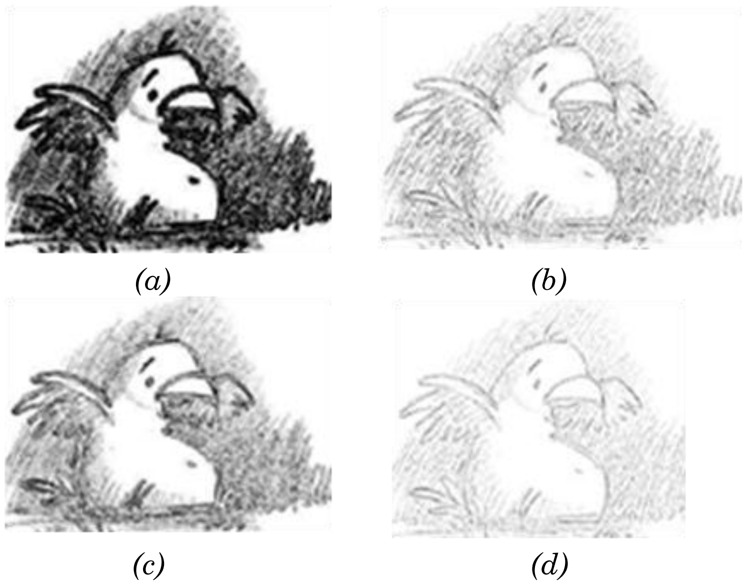


Figure 4: ***Results of different traditional edge detection techniques applied to the sketch in Figure 3 (g).*** *(a) First derivative method (b) Second derivative method (c) First derivative of Gaussian (d) Second derivative of Gaussian. (a) and (c) suffers from double-edge problem while (b) is sensitive to noise. (d) performs the best among the four methods but the lack of stroke information makes the outline tracing difficult.*

The results of these methods are shown in *Figure 4*. These methods are good at obtaining the location of strokes. However, they fail to determine the direction of strokes and distinguish one stroke from another. Besides, they do not have a stroke pruning scheme which prunes away unnecessary strokes in sketches. This makes them unsuitable for outline tracing.

Flow Based Edge Detection Later edge detection techniques give the direction of intensity change within a local area. This information gives us information on the direction of strokes in the sketches. Some examples include flow-based difference of Gaussian (FDoG) and structure tensor [9][10][11]. *Figure 5* shows the direction of strokes detected using structure tensor.

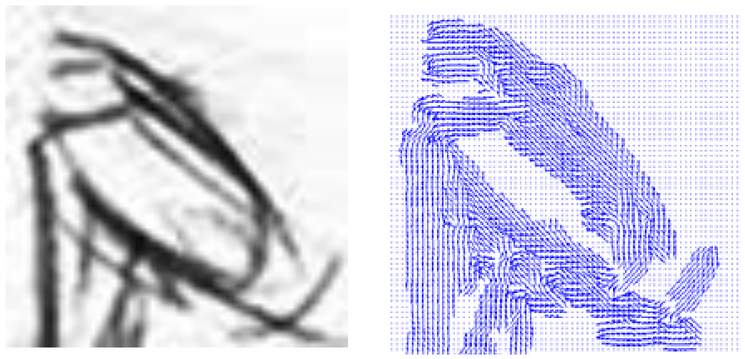


Figure 5: ***Direction of strokes obtained from structure tensor using Figure 3 (a) as input****. Only the rabbit's ear on the top right hand corner of the image is shown.*

Using this method, the direction of strokes at a particular position is the average directions of neighbouring lines. But it is difficult to distinguish between different lines.

<u>Centerline Detection</u> Centerline detection method first smears surrounding strokes into thickened lines, and the middle centerline of thickened lines is chosen as the final stroke. This may be useful in simple drawings but not complicated sketches, which otherwise may produce large number of unnecessary lines in the traced result.

2.3.2 Methods for Vector Image

The use of drawing tablets becomes popular as the input device for sketching. Thus, most of the sketch-related works take vector images as input. Among these works, some of them focus on building 2.5D [18] or 3D models [17] [24] based on sketches; some of them focus on domain specific sketch recognition [22]; some of them focus on beautification of sketches by fitting curves or straight lines and detecting corner points [20] [23]; Some of them refine imprecise sketches by adding details with the help of training set [19]; but very few of them have a stroke merging or pruning strategy which makes them suitable for refining rough sketches that contains unnecessary strokes. Among these limited number of works, they are either technique with or without user intervention. A brief discussion of these two categories of methods is given below.

Techniques with User Intervention There are a number of methods which handles vector input with information provided by the users [12] [13]. Baudel proposed the use of modifying strokes from user input to modify a curve. However, using this method does not preserve user input during the drawing process. This can inhibit the reinterpretation of sketches which is undesirable. Besides, during sketching, artists have to select among four different editing modes using keyboard which strongly interferes the sketching process.

Bae et al. proposed a method which obtains a single stroke by averaging a number of strokes. But whenever the user wants to merge multiple strokes into one single stroke, it has to be specified explicitly. This again is undesirable as it interrupt the sketching process.

To conclude from previous works, when designing tools for outline tracing task, we should keep user intervention as little as possible. Besides, we should keep all the strokes during sketching process to facilitate the reinterpretation of sketches.

<u>Techniques without User Intervention</u> Very few existing methods refine sketches without user intervention. Barla et al. [21] proposed a distance-based method to group close and continuous strokes. Their method assumes that in the same sketch, the grouping measures are the same across different areas, which is indeed not a valid assumption. When we determine which strokes are grouped together, we take into account of the size and shape of the region formed by the strokes. We have a higher tendency to group strokes which are further away from each other in a large shape than in a small shape. See *Figure 6* for an example.

The state-of-the-art work which does not require user intervention but can handle unnecessary strokes in sketches is proposed by Orbay and Kara [14]. They asked artists to group related strokes in their sketches manually. The stroke grouping together with the distance, misalignment and discontinuity between every stroke pairs were used as feature vectors to train a neural network. The trained neural network was used for grouping strokes in other sketches drawn by the same artist. After grouping and reordering point within the same group, cubic B-spline fitting was performed to the points in the same group to obtain the final outlines.

However, their method makes the same assumption as Barla. Besides, they assume that an artist draws sketches with similar degree of roughness, which is not always the case. Therefore, even if the neural network is trained by input from the same artist, the correct grouping of points is not guaranteed for the testing set.

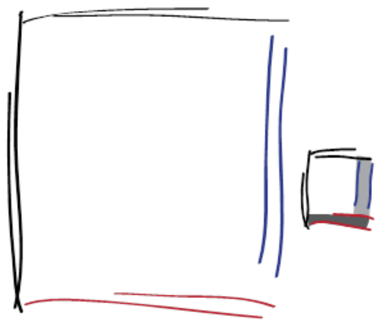


Figure 6: ***A comparison of the interpretation of strokes for shapes that are of different sizes.*** *The blue strokes are grouped in the larger shape while they are not grouped in the smaller shape. It is the same for the red strokes.*

2.3.3 Deficiency

As discussed earlier in this section, current methods for both raster images and vector images are not sophisticated enough to handle the outline tracing task and each of them has their own limitations. The common reason of their limitations is that they ignore the underlying semantics of strokes in sketches. When we perceive strokes in sketches, we do not only perceive them as separate strokes but groups of combined strokes and shapes. Without taking into account how we group visual elements together when we perceive a drawing, we cannot develop a sophisticate solution for this purpose. In the next section, we introduce several principles of how human perceives and analyzes sketches.

3. Gestalt Principles and Its Application

In Cognitive Psychology, gestalt principles are among the most influential principles in the area of visual perception proposed by Wertheimer [16]. These principles describe how human cognition group visual elements together in an automated manner. In this section, a brief introduction and an existing computational model of gestalt principles are described. At the end of the section, we propose how we can analyze and refine sketches by making use of gestalt principles.

3.1 Introduction to Gestalt Principles

There are a number of different versions of gestalt principles. Among them, the five most common gestalt principles which influence grouping of visual elements in a static scene are the laws of similarity, proximity, regularity, continuity and closure. *Figure 7* demonstrates how the five principles work.

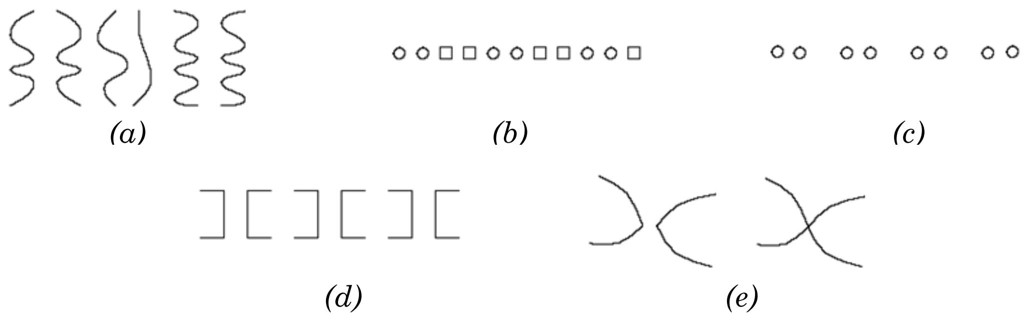


Figure 7: ***Example demonstrating Gestalt principles.*** *(a) Regularity (b) Similarity (c) Proximity (d) Closure (e) Continuity*

Similarity When we perceive a visual scene, we tend to group elements which are similar in a group. Similarity can be measured in terms of color, shape, size or orientation of member elements.

Proximity We tend to group elements that are close to each other as a group. Note that closeness is relative to the scale of the image. When an image is scaled differently, we will still obtain the same grouping as the relative distance between elements remains the same.

Regularity If the elements have a high regularity, we group them into a group. Regularity can be measured in terms of symmetry (*Figure 7 (a)*), parallelism or the pattern that neighbouring elements form. Regularity groups usually contain a number of similarity groups. See *Figure 8* for an example.

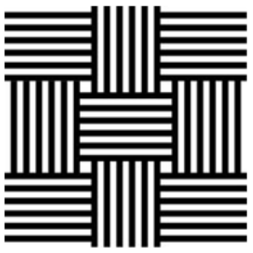


Figure 8: ***Another sample regularity group****. It is composed of similarity groups of vertical lines and horizontal lines.*

Continuity Strokes or shapes that are continuous are considered as one group.

Closure Even when individual elements are not continuous, if the elements form a recognizable closed shape, the elements are mentally connected and grouped together.

Different gestalt principles work concurrently when our brain analyzes an image. The principles interact and overlap with, conflict with or mask the effect of each other depending on the situation.

3.2 Existing Computational Model of Gestalt Principles

Though some research works made use of gestalt principles implicitly, such as the method proposed by Orbay and Kara [14], they did not handle the conjoining of different gestalt principles. Nan et al. proposed the first computational model combining several gestalt principles [15].

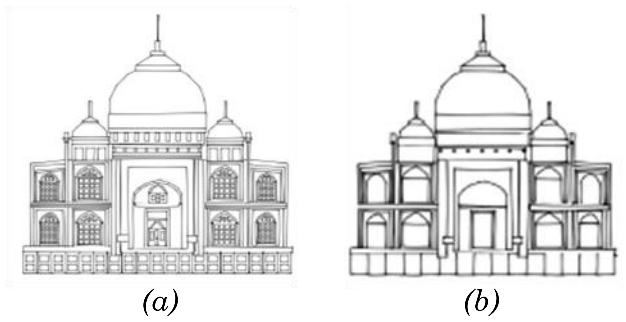


Figure 9: ***Abstraction of architectural drawings.*** *(b) is the abstraction of (a) using Nan's method.*

In Nan's work, they applied their model incorporating gestalt principles to obtain abstracted architectural drawings. A sample result is shown in *Figure 9*. The basic elements in the input are combination of a number of simple shapes. The relationship between each shape is represented as a multi-label graph. In the graph, each node represents a shape. Each of them belongs to a number of potential gestalt groups which are represented as labels. Neighbouring shapes are connected with edges and the weight of each edge is defined as the distance between two connected shapes.

There are four types of potential gestalt groups, proximity group, similarity group, regularity group and closure group while continuity is considered as a special case of closure. To determine the potential gestalt groups, they introduce some measures to calculate the affinity of a node to a group. A node is said to be potentially belongs to a group if its affinity to that particular group exceeds a threshold.

Each node may have multiples potential gestalt groups. These potential groups may overlap with, conflict with or mask each other. Graph cut minimization is used to handle conflict and masking while ignoring the overlapping of gestalt group. They minimize an energy function which is the sum of data cost, smoothness cost and label cost. Data cost reflects how well a node fits to a particular group, smoothness term increases the probability of neighbouring nodes to be classified to gestalt groups, and the label cost penalizes overly-complex grouping.

The result of their proposed computational model on their chosen application is quite satisfactory. However, their method is not general enough to be applied to other problems. First, their method can only be applied to simple problems. Their input elements are some simple shapes such as rectangle or circle. The affinity measure for different gestalt principles are defined based on this assumption. For example, the affinity measure for proximity group is defined by the distance between the centers of two shapes. However, this measure is obviously not suitable for our outline tracing task. Besides, they measure the closure by fitting predefined shape to the elements and this makes it unsuitable for measuring the closure of arbitrary shape.

Secondly, their method cannot handle overlapping of gestalt principles. However, sometimes, we cannot simply ignore the other overlapping gestalt principles and assign the element to either one of the potential gestalt principles. *Figure 10* demonstrates the problem of allowing only one grouping. This problem can be solved either by introducing a strategy to allow overlapping of groups or by cutting the strokes which are involved in more than one group at the junction points.

Although Nan's work is not suitable for our application, it gives us insight that gestalt principles can aid our outline tracing process.

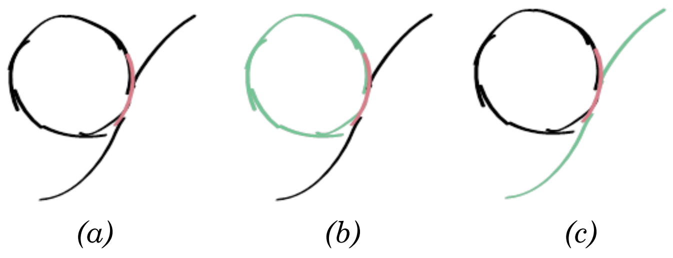


Figure 10: ***Overlapping of gestalt groups.*** *The red stroke in the sketch shown in (a) has multiple potential gestalt groups. It can be grouped to the circle as shown in (b) by closure and continuity principles while it can also be grouped to the curve shown in (c) by continuity principle. If we restrict the red stroke to be belongs to one gestalt group only, the other group will be lost or incomplete.*

3.3 Gestalt Principles for Outline Tracing

As discussed in Section 2, the existing related methods do not have a thorough understanding of the underlying semantics of strokes in sketches, making it impossible to trace the outline of sketches in a sophisticated way. Therefore, we propose using gestalt principles to group strokes together to form a number of regions first and follow with outline synthesis process based on the regions formed. In this section, we explain why different gestalt principles are useful in our application.

3.3.1 Similarity

As discussed in Section 2, some sketches are drawn in multiple colors while one of them represents the core lines. By taking into account of the similarity in terms of color, we can group strokes with similar color together. This can be achieved easily by computer. But since sketches drawn in multiple colors are not very common, especially for vector images, for simplicity, we do not handle them in this work. We assume the input images are single-tone images.

3.3.2 Proximity

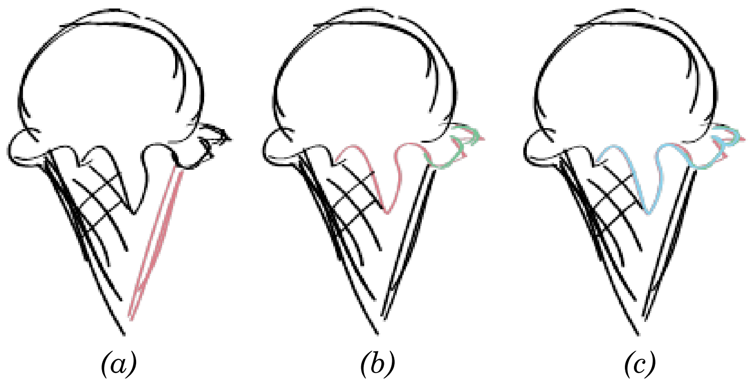


Figure 11: ***Application of law of proximity in sketches.*** *(a) Sketch with one of the proximity group marked in red. (b) Another proximity group. The strokes are grouped to form one single blue stroke in (c)*

Proximity is one of the most commonly used gestalt principles in outline tracing process. According to the law of proximity, close strokes should be grouped together to form a single group. This is demonstrated in *Figure 11*. But as discussed, degree of proximity is a relative measure to the neighbouring region. Therefore, unless we obtain regions from the sketch, the grouping based on proximity will not be accurate.

3.3.3 Continuity

Another common way to group multiple strokes into one single stroke is to measure the continuity between strokes. Two strokes are considered as continuous if there exists a smooth transition from one stroke to the other stroke. The idea is demonstrated in *Figure 12 (a)*. *Figure 12 (b)* and *(c)* show some sample continuity groups of a sketch. Note that for sketches, the law of continuity always comes with law of proximity. If two strokes are very far away, even if they are continuous, they will not be grouped together. Since law of proximity in turn depends on the neighbouring shape. We conclude that it is the regions that matter not proximity or continuity when grouping strokes together.

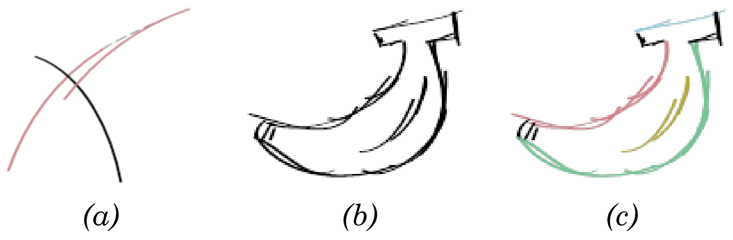


Figure 12: ***Application of law of continuity in sketches.*** *(a) Red strokes are grouped together due to smooth transition. (b) A sample sketch. Strokes grouped by continuity are shown in (c)*

3.3.4 Regularity

Shading is usually composed of regularly spaced strokes which form pattern. Therefore, shading of sketches can be grouped together by observing the regularity of strokes in a particular area. This is demonstrated in *Figure 13*. However, as shading is not very common in sketches, to avoid making the situation to be too complex, in this thesis, we assume that the sketch does not contain any shading and do not take into account of regularity during grouping. However, this kind of grouping is technically feasible.

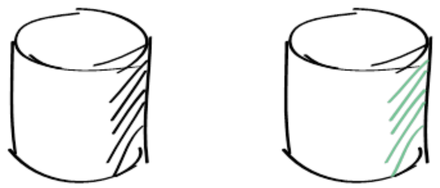

(a) (b)

Figure 13: ***Application of law of regularity in sketches.*** *(a) Sketch. (b) Strokes grouped by regularity*

3.3.5 Closure

Law of closure is the most important principles among all the gestalt principles discussed. By law of closure, we perceive sketches as a number of closed regions which are important elements in art. Salient closed region must be retained after the outline tracing process. More details are given in Section 4 when we discuss how we make use of regions in the proposed method. *Figure 14* demonstrates a rough idea of how this law works on sketches.

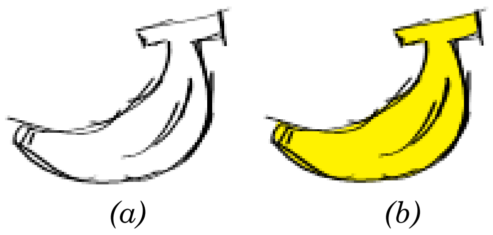


Figure 14: ***Application of law of closure in sketches.*** *(a) Sketch. (b) Region perceived from (a).*

4. Proposed Method

Based on our previous discussions, we propose a region-based approach to refine sketches. The proposed method consists of four modules. The framework is demonstrated in *Figure 15*. The first module is the multi-scale region retrieval module. In this module, given a vector image of the sketch, we retrieve regions of different scales from the sketch and construct a hierarchy of regions. Then, in the salient region retrieval module, we obtain a flattened region map based on the hierarchy. Only regions that are salient to human visual perception remain in the flattened region map. The region map is then passed to the outline synthesis module to reconstruct region-boundary strokes and feature strokes in sketch. Finally, the outlines are curve fitted using piecewise cubic Bezier curve in the curve fitting module and are exported as vector image. In this section, each of the four modules is described in detail.

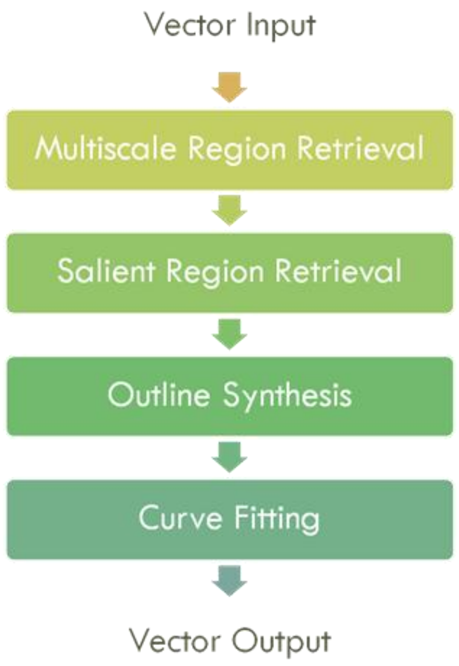


Figure 15: ***Framework of my proposed method.***

4.1 Multi-scale Region Retrieval

In the previous section, we pointed out that the law of closure governs the way we perceive sketch and suggested that we can refine sketches based on regions. Finding regions is an easy task for human but it is a difficult task for computer due to leakage problem. *Figure 16* demonstrates an example. Rolling ball method can effectively prevent leakage. The basic principle of this method is that we use a ball to roll along the boundary and fill the region which is accessible by the ball. Different size of ball will blocked at different size of gap and thus resulting in different shape of filled region. *Figure 17* demonstrates the idea.

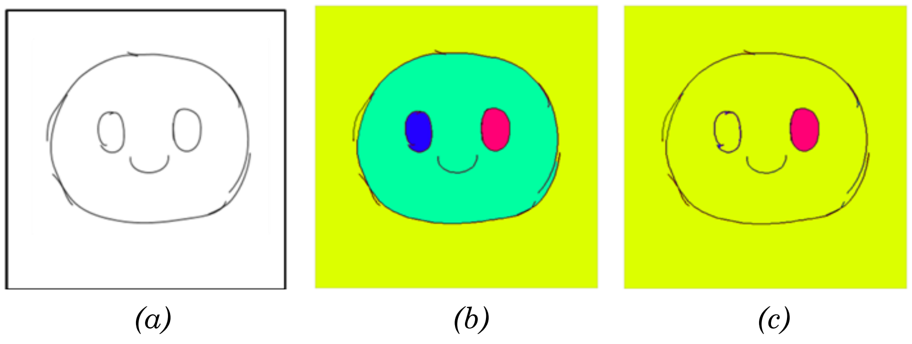


Figure 16: ***Comparison of region found by human and computer.*** *(a) is the sketch. (b) shows the region perceived by human while (c) shows the region found by computer using flood-filling method.*

Deciding a suitable ball size is a challenging task since small regions can only be detected by small rolling ball but a small rolling ball can lead to leakage in some other regions in the sketch. We cannot simply use a constant rolling ball size. Therefore, in multi-scale retrieval module, we prepare a hierarchy of regions in which each level corresponds to a particular rolling ball size and select the suitable regions from the hierarchy at the later stage.

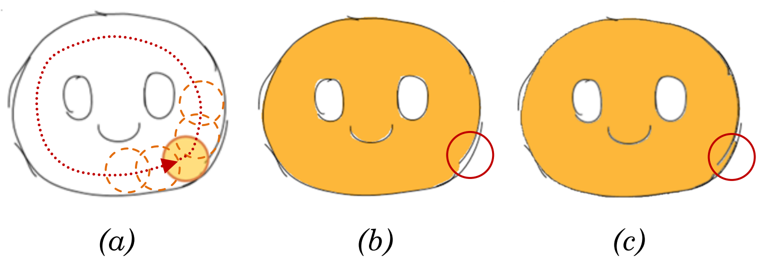


Figure 17: ***Rolling Ball Method.*** *(a) A ball is roll along the boundary to obtain region. (b) and (c) show the shape formed using a large rolling ball and a small rolling ball respectively. The red circles highlight the difference of the two shapes.*

4.1.1 Construction of Region Hierarchy

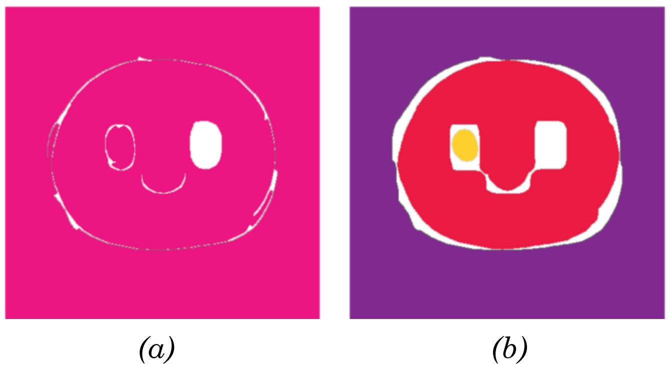


Figure 18: ***Parent-and-Children Relationship of Regions.*** *(a) shows a region (colored in pink) obtained with a smaller rolling ball (smaller scale) while (b) shows regions (colored in red, orange and purple) obtained with a larger rolling ball (larger scale).*

OBSERVATION As discussed earlier, different regions are obtained with different rolling balls (or scales). Despite this, we observe that regions of larger scale (larger rolling ball) must be contained by a region of smaller scale. Thus, a parent-and-children relationship can be formed which encourages the construction of region hierarchy. *Figure 18* demonstrates the observation.

In Practice To avoid missing thin regions, the input image is first scaled up at the very beginning of our refining process before construction of region hierarchy. This is demonstrated in *Figure 19*.

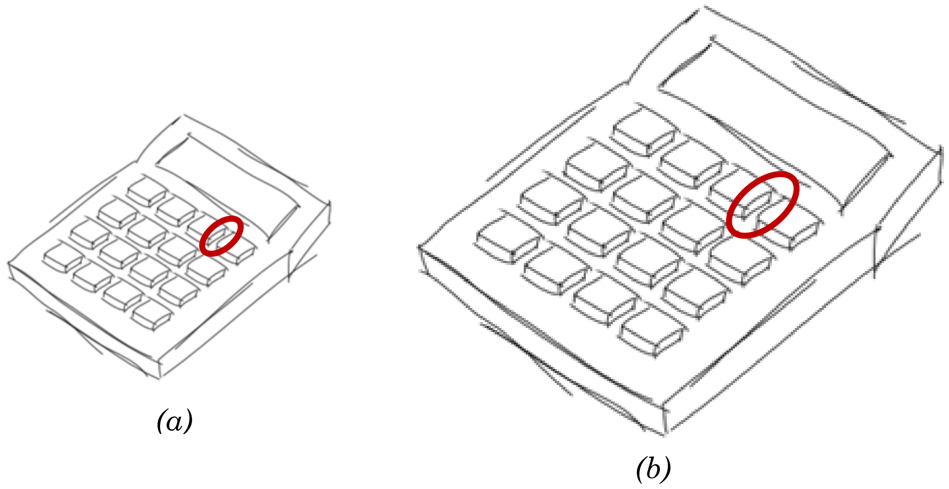


Figure 19: ***Scaling up of input image.*** *Thin gap between regions is detected in a larger scale (b) but not in a smaller scale (a). To obtain precise regions, scaling up of input image is needed.*

Next, to construct a region hierarchy from the sketch, we fill the sketch with rolling balls of different sizes, thus obtaining regions of different scales. Then, we arrange the regions from smaller scale to larger scale and put regions of smaller scale at the top of the region hierarchy. For the sake of completeness, an arbitrary level is added at the top of the hierarchy to represent the canvas as shown in *Figure 20 (a).* Then, we connect the regions which demonstrate parent-and-children relationship and form a hierarchical structure. A simplified version of the region hierarchy is shown in *Figure 20 (b)* in which regions are represented as nodes.

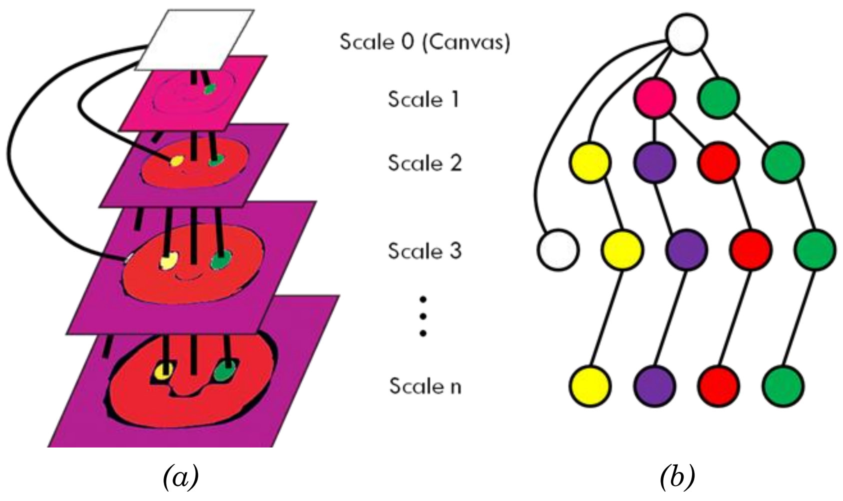


Figure 20: ***Construction of Region Hierarchy.*** *(a) shows the region hierarchy constructed. (b) is a simplified version of the region hierarchy.*

4.1.2 Region Refinement

OBSERVATION By looking at *Figure 18* again, we also observe that regions of larger scale are less precise than that of smaller scale. But thanks to the parent-and-children relationship of the shapes of different regions, we can refine the regions by propagating the shape of upper-level regions to lower-level regions.

IN PRACTICE As discussed, the region refinement process is performed by propagating shapes from the upper levels to the lower levels. This is achieved by taking each of the parent regions as a mask and then overlaid it to its child regions. A color diffusion process is then performed on the child regions and is stopped at the white region of the overlaid mask. This process is repeated until it reaches the leaf node of the region hierarchy. *Figure 21* demonstrates the whole process.

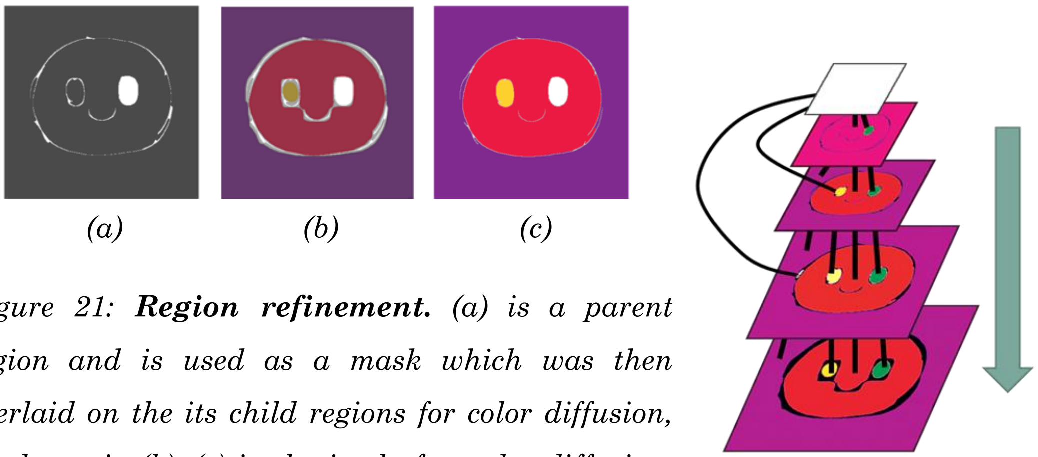


Figure 21: ***Region refinement.*** *(a) is a parent region and is used as a mask which was then overlaid on the its child regions for color diffusion, as shown in (b). (c) is obtained after color diffusion. The process is repeated from the top levels to the lower levels of the region hierarchy as shown in (d). The arrow shows the direction of shape propagation.*

4.2 Salient Region Retrieval

In the multi-scale region retrieval module, we obtain a region hierarchy. In the hierarchy, multiple regions of different scales may occupy the same space in canvas which violates human visual perception. Therefore, the region hierarchy must be flattened so that only one single region remains in each space. After the flattening process, regions are merged or pruned so that only regions salient to human visual perception remain.

4.2.1 Flattening of Region Hierarchy

Observation In the region hierarchy of a sketch, regions of upper level are more likely to be a result of leakage problem.

In Practice Based on the above observation, we conclude that when flattening the region hierarchy, for each region, we should choose its child regions whenever possible. Following this strategy, we can obtain a flattened region map by combining all the leaf nodes (or regions) in the region hierarchy as shown in *Figure 22*.

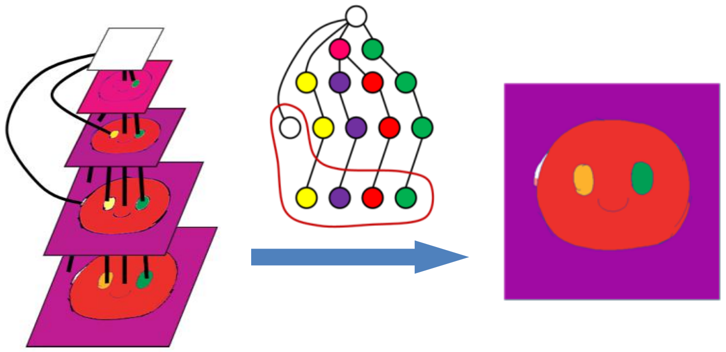


Figure 22: ***Flattening of Region Hierarchy.*** *By combining leaf nodes in region hierarchy, we obtain one single region map.*

4.2.2 Region Merging

OBSERVATION When we perceive regions, we tend to merge regions where there is a large gap in between while preserving those with a small gap. This phenomenon is demonstrated by comparing drawing (a) and (b) in *Table 1*. But note that whether regions should be merged is not purely determined by the absolute size of the gap. For example, by looking at drawing (a) and (c), we observe that even though the gaps between regions are of the same size, we tend to merge regions in (c) but not in (a). In fact, we only merge regions where the size of the in-between gap is comparable to the neighbouring regions. So the problem becomes what means by comparable to the neighbouring regions. The most intuitive method is to compare the size of neighbouring regions and the size of the in-between gap. However, this is proven incorrect by observing drawing (a) and (d). Even though regions of (a) and (d) are of similar size, regions in (d) are merged but not in (a). But we have an important observation based on drawing (a) and (d). We observe that the maximum rolling ball size we use to detect (d) is smaller than (a) as shown in *Figure 23*. Therefore, we conclude that whether two regions are merged is determined by two factors, the size of the gap and the maximum size of rolling ball used to detect the regions.

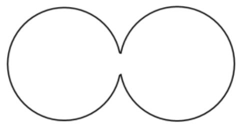
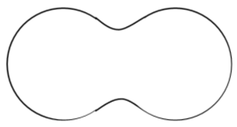
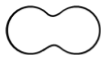
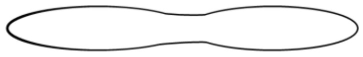
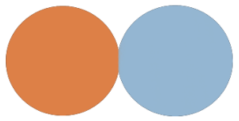
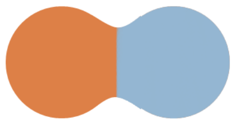
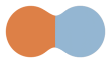
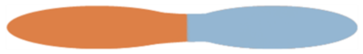
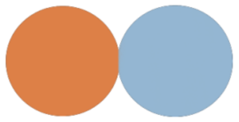
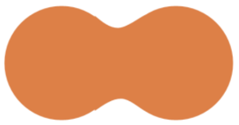
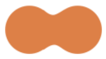
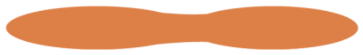
	(a)	*(b)*	*(c)*	*(d)*
Original drawing				
Region(s) obtained after flattening				
Region(s) perceived by human				

Table 1: ***Comparison of the regions obtained after flattening and the regions perceived by human.***

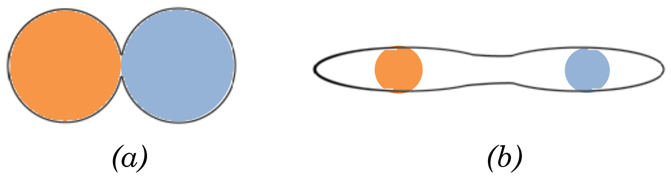


Figure 23: ***Maximum rolling ball used to detect regions.*** *Maximum size of rolling ball used to detect regions in (a) is larger than (b).*

IN PRACTICE After the region flattening process, we merge regions together based on the size of the gap and the maximum size of rolling ball used to detect the regions. This is achieved by first building a region connectivity graph and then merging regions based on the graph.

To construct a region connectivity graph, we make use of the flattened region map. Each of the regions in the region map is represented as a node in the graph. Then we connect regions which are neighbours and there is no stroke in between. The idea is demonstrated in *Figure 24.*

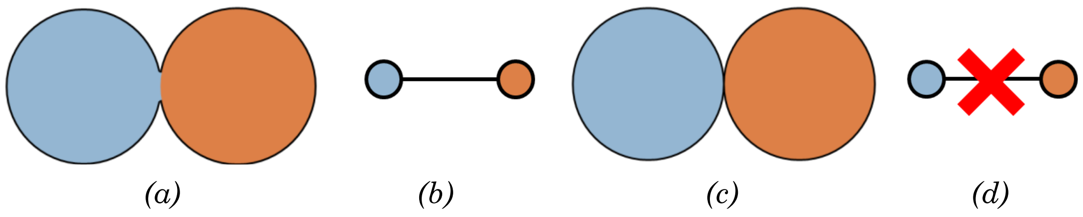


Figure 24: ***Construction of region connectivity graph.*** *Though (a) and (c) are similar but since the regions in (c) are separated by stroke, we claim that they are disconnected and thus do not have an edge between the nodes in the region connectivity graph as shown in (d). But for (a), we can construct a graph as (b).*

After constructing a region connectivity graph, we assign distance to each of the edges in graph. The distance between two connected regions defines the tendency of merging them together. If the distance is higher, the tendency to merge them will be lower and vice versa. The distance between edges is defined as:

$$Distance = \frac{Max(ballsize(i), ballsize(j))}{gapsize(i,j)}$$

Equation 1: ***Distance formula between nodes in region connectivity graph***

where i, j are the node indices. *ballsize(k)* is the maximum rolling ball size used to detect region k. *gapsize(k_1,k_2)* is the size of gap between regions k_1 and k_2. The distance formula is designed based on the observation that merging of regions is determined by gap size and maximum rolling ball size. Besides, this formula is designed so that no matter how you scale the sketch, the tendency of merging regions remains unchanged.

The next step is to merge regions based on the region connectivity graph. This is achieved by making use of a clustering technique called Hierarchical Clustering. We consider each region as a data point and each merged region as a cluster. There are several ways to measure the distances between clusters in Hierarchical Clustering. Among them, we select single linkage. That means the distance between two clusters (merged regions) is defined as the shortest distance between members of the two clusters. With this technique, we can build a dendrogram by progressively merge regions or clusters which are "close". But please be reminded that to define "close", we are not talking about the spacial distance between regions but the distance we obtained using *Equation 1*. *Figure 25 (a)* and *(b)* shows a simple example.

We can then cut the dendrogram at different threshold. If a higher threshold is used, more regions will be merged together to form one single region. This is demonstrated in *Figure 25 (c)* and *(d)*. In practice, this threshold is a constant and is obtained by experiments using different input sketches.

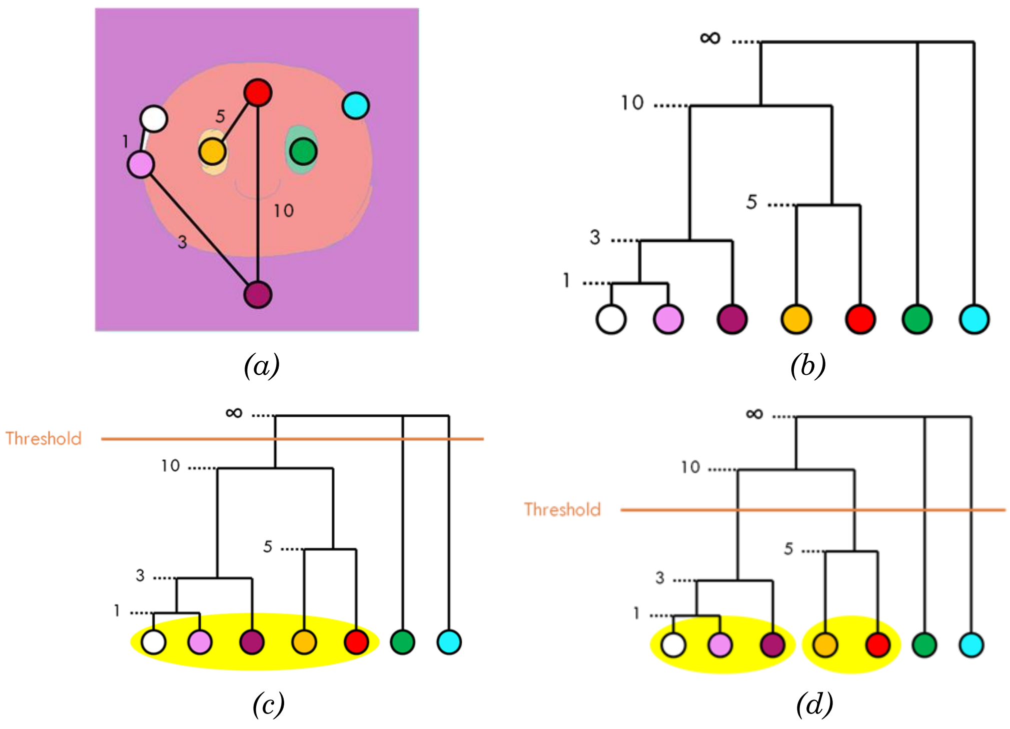


Figure 25: ***Mechanism of Region Merging.*** *Given a region connectivity graph (a), a dendrogram (b) is constructed by single linkage. Then, the dendrogram is cut at a specific threshold which is obtained by experiment. (c) and (d) shows the dendrogram cut at different threshold and the regions highlighted in yellow are the regions that are merged.*

4.2.3 Region Pruning

OBSERVATION After the region merging process, some non-salient regions to human visual perception remain which must be pruned away. Some of these regions are highlighted in *Figure 26*. We observe two important characteristics of these regions. First, they are thin. They can only be detected by a small rolling ball. But it does not necessarily mean they are small regions. Note that thin is a relative measure which is affected by the neighbouring regions. The idea is demonstrated in *Figure 27*. The highlighted red region has higher probability to be pruned away in

(b) than in (c). We observe that the only different between (b) and (c) is that the neighbouring region in (b) can be detected by larger rolling balls. Therefore, we conclude that we should prune away regions that are thin with respect to the neighbouring regions.

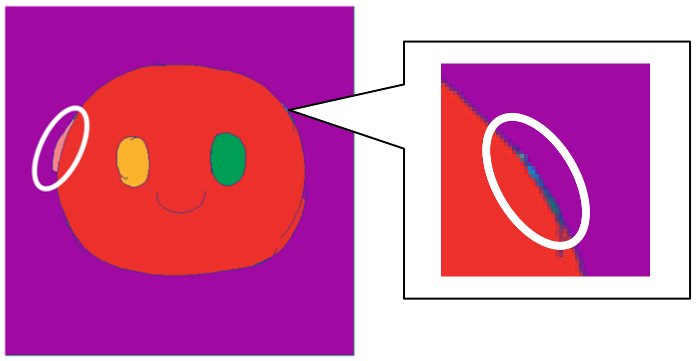


Figure 26: ***Non-salient regions.*** *The regions that are non-salient to human visual perception but are detected at the previous stages are circled.*

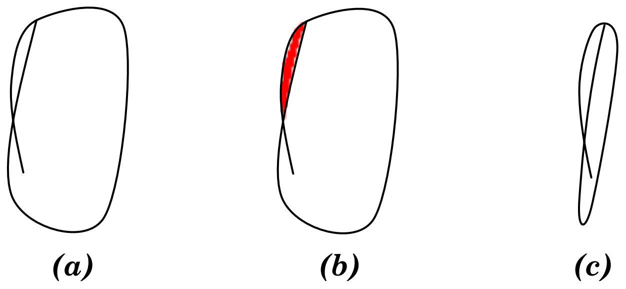


Figure 27: ***Neighbouring regions affect the pruning decision.*** *When we perceive the sketch in (a), we tend to prune away the thin red region shown in (b). On the other hand, we tend to retain the thin region in sketch (c) since it is significant when compared to the neighbouring regions.*

Besides, these regions are generally of similar shape as the region shown in *Figure 28*. By observing *Figure 28*, we can see that the direction vectors along the region boundary of this type of shapes are consistent, which also means that consecutive region boundary vectors point at similar direction.

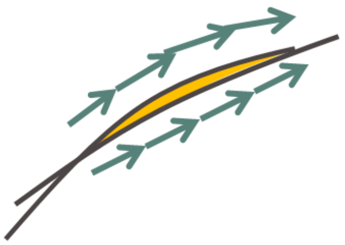


Figure 28: ***General shape of non-salient regions.*** *The shape of non-salient regions is usually thin and its direction vectors along region boundaries are consistent.*

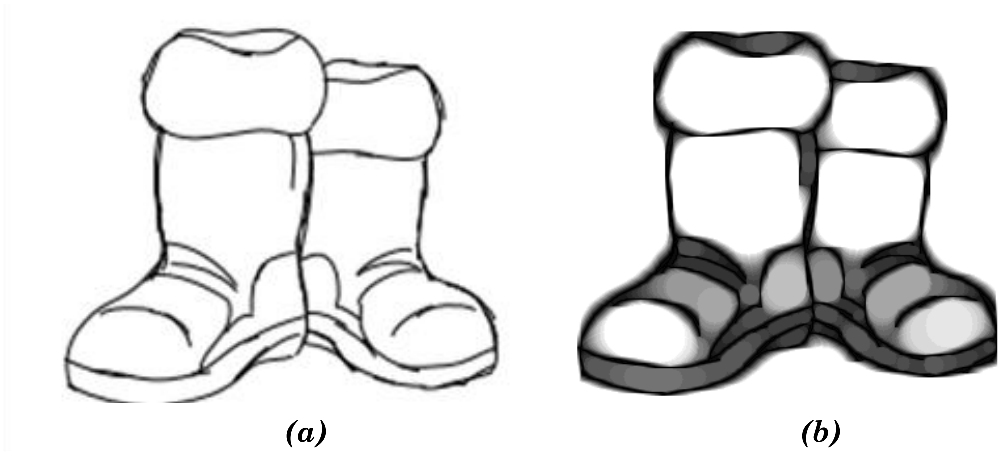


Figure 29: ***Accessibility map.*** *(b) shows the accessibility map obtained from (a). Darker pixel means it can only be detected by smaller rolling balls*

In Practice From our observations, we conclude that thin regions with high consistency of boundary vectors should be pruned away. However, some regions are simply too small or thin which make them non-salient to human visual perception no matter what. Therefore, to save computations, these regions are first pruned away before any calculations.

To identify regions that are thin with respect to their neighbouring regions, we define a relative thinness measure. We first construct an accessibility map based on the region hierarchy obtained in multi-scale region retrieval module. The value at each pixel in an accessibility map represents the maximum size of rolling ball used to detect that pixel. *Figure 29* shows a sample accessibility map. A dark pixel means it can only be detected by small rolling balls. It, in turn, means that it is in a narrow region. Then, we perform convolution on the map with a circular mask to obtain a blurred accessibility map which represents the average thinness of neighbouring regions at each pixel and identify target pruning region with the formula:

$$Relative\ thinness = \frac{n \times ballsize(i)}{\sum_{j=1}^{n} map(B_{ij})}$$

Equation 2: ***Relative thinness measure of regions***

where i is the region index; *ballsize(i)* is the size of the largest rolling ball used to detect region i; B_{ij} represents the jth boundary point of region i and there are n boundary points in total; *map*(B_{ij}) is the value of the blurred accessibility map at point B_{ij}. In short, the relative thinness measure is defined as the ratio between the absolute thinness of a region and the average absolute thinness of its neighbouring regions.

Then, we measure the consistency of the surrounding boundary vectors using the formula:

$$Consistency = \frac{1}{n-1}\sum_{i=1}^{n-1} abs(v_i \cdot v_{i+1})$$

Equation 3: ***Consistency of surrounding boundary vectors***

where v_i is the ith direction vector on region boundary and n is the total number of direction vectors. This formula, in turn, measures the average angle difference between two consecutive region boundary vectors.

By simple thresholding, thin regions with high consistency of surrounding boundary vectors are pruned away.

4.2.4 Region Merging by User

OBSERVATION Besides the Gestalt principles we discussed in Section 3, there is another Gestalt principle which affect the way we perceive sketches. It is the law of meaningfulness or familiarity. By the law of meaningfulness or familiarity in Gestalt principles, we tend to group things into something that are familiar to us. *Figure 30* demonstrates the idea. There are a banana and an apple in the sketch. Since the shapes of banana and apple are familiar to us, we know that the middle region should either be grouped to the yellow region or the red region. However, for computational methods, as we have no idea what the artist is drawing, we do not have enough information to group the regions together. Therefore, in salient region retrieval module, after region flattening, merging and pruning, the artists must be allowed to merge regions manually.

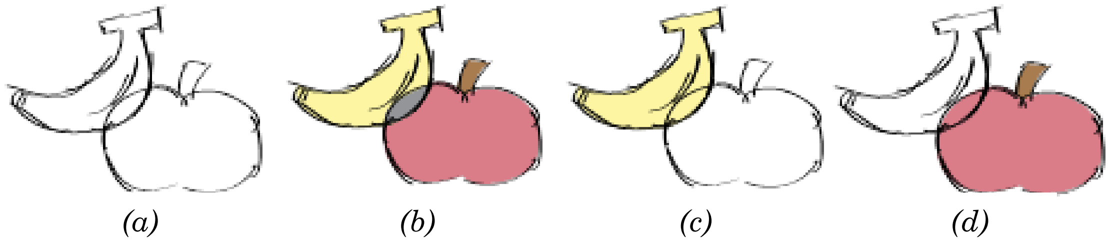


Figure 30: ***Law of familiarity.*** *With (a) as the original sketch, after region flattening, merging and pruning, we obtain (b). However, when we perceive the sketch, since we know we are drawing a banana or apple, we know we should group the middle region either to the yellow region or the red region as shown in (c) and (d).*

IN PRACTICE Currently, for simplicity, we provide the user with a region map labeled with region indices and require the user to type in the region indices of the regions that he wants to merge in the command line. Definitely, a better user interface can be used but since this is not important at this stage, the current solution is still acceptable. After manual region merging by user, we obtain a region map which contains only regions that are salient to human visual perception.

4.3 Outline Synthesis

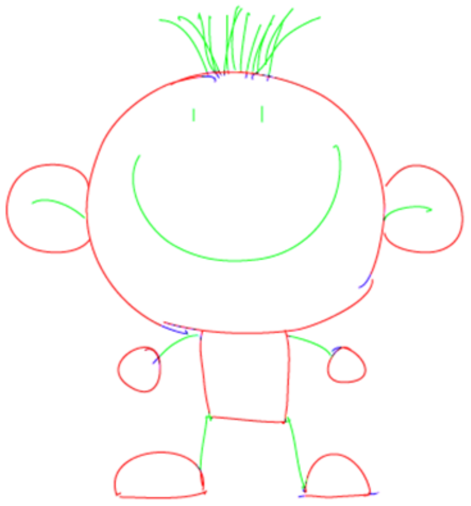


Figure 31: ***Different types of strokes in sketches.*** *Region-boundary strokes in red; feature strokes in green; unnecessary strokes in blue.*

In outline synthesis module, we reconstruct the outlines based on the region map obtained in the salient region retrieval module. We observe that there are two important types of strokes that should be retained in the outline tracing process. They are region-boundary strokes and feature strokes. Region-boundary strokes are strokes which form closed shapes. Feature strokes do not form closed shapes but the strokes themselves define the shape of a component such as a mouth or an eye. They can also be feather or hair. Some examples of region-boundary strokes and feature strokes are highlighted in *Figure 31* with red and green respectively. Due to the different natures of strokes, region-boundary strokes and feature strokes are synthesized separately in the outline synthesis module.

4.3.1 Outline Synthesis of Region-boundary Strokes

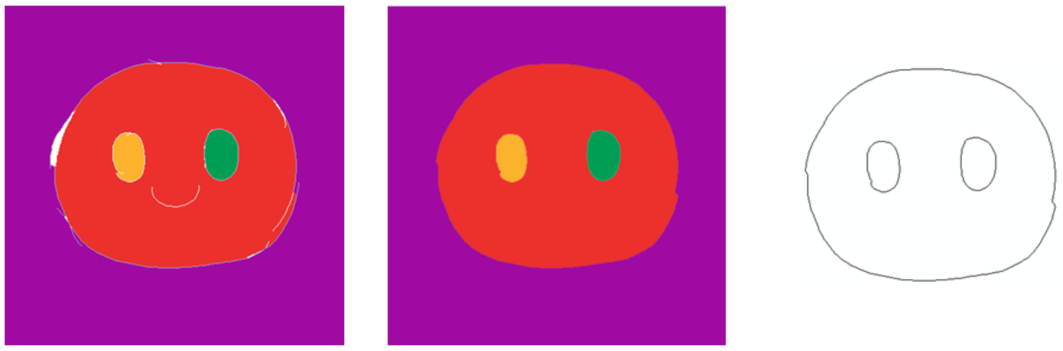


Figure 32: ***Outline Synthesis of Region-Boundary Strokes.*** *Given a region map in (a), diffusion process is performed to obtain (b). Then, we retrieve the region boundary from (b) and obtain (c).*

OBSERVATION By definition, region-boundary strokes are strokes which separate regions in sketches. Region-boundary strokes synthesis is straight forward but since there is some empty space in the region map obtained in the salient region retrieval module, double edge may appear. Therefore, we must remove any empty space in the region map before region-boundary detection.

IN PRACTICE To remove empty space in region map, a diffusion process is performed. Regions are diffused until they meet another region. A sample output after the diffusion process is shown in *Figure 32 (b)*. After diffusion, the locations at which neighbouring pixels are of different regions are detected. These points are then connected and cut at junctions to obtain different branches by connectivity for later processing. *Figure 32 (c)* shows a sample region-boundary strokes obtained in this module.

4.3.2 Smoothing of Region-boundary Strokes

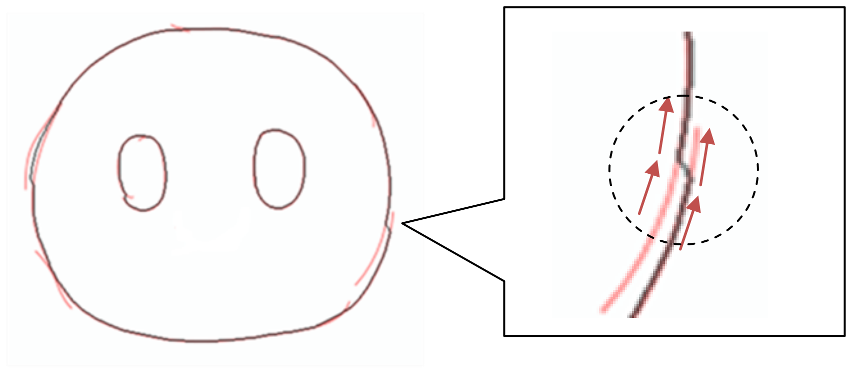


Figure 33: ***Characteristic of Discontinuous Point Introduced in Refining Process.*** *By overlaying the reconstructed outline (black strokes) on user input strokes (red strokes), we observe that at the discontinuous point we introduced, the directional vectors of the surrounding user input strokes are similar.*

Observation During the salient region retrieval process, we may introduce discontinuity in the region boundaries. One example is shown in *Figure 33*. These discontinuous points are unacceptable in the refined outline. Therefore, we must smooth the outline at these points. Though these discontinuous points look similar to other corner points in sketches, there is an important characteristic that makes them different. At the discontinuous points we introduced, the directional vectors of the surrounding user input strokes are of similar direction. Thus, we make use of this characteristic to identify target smoothing positions and perform smoothing.

In Practice To identify all the discontinuous positions in sketches, we measure the angle between consecutive stroke segments on our synthesized region-boundary strokes. If it exceeds a threshold, it is identified as a discontinuous position. Among these points, some of them are introduced during the sketch refining process while some of them are corner points intentionally drawn by artists.

To distinguish between them, we define a neighbourhood at each discontinuous position. If the direction of the user input strokes is consistent in the neighbourhood, we identify the discontinuous point as a target smoothing position. Instead of using a fixed neighbourhood, the size of neighbourhood is calculated using the blurred accessibility map obtained in the region pruning step. The formula is given in *Equation 4.*

$$Size\ of\ neighbourhood^2 = k \times map(D_i)^2 + h$$

Equation 4: ***Size of neighbourhood at each discontinuous point***

where k, h are parameters; *map(*D_i*)* is the value of the blurred accessibility map at ith discontinuous point. In short, it means that points in denser area will have a smaller neighbourhood and vice versa.

This design can avoid including unwanted directional vectors from other regions in dense area or missing directional vectors in sparse area. The idea is demonstrated in *Figure 34.*

To measure the consistency of the user input strokes in a neighbourhood, we perform clustering on their directions. If the discontinuous position is a corner point, there should be at least two clusters. If the discontinuous position is introduced during the refining process, there should be only one cluster and we should smooth the outline at this position.

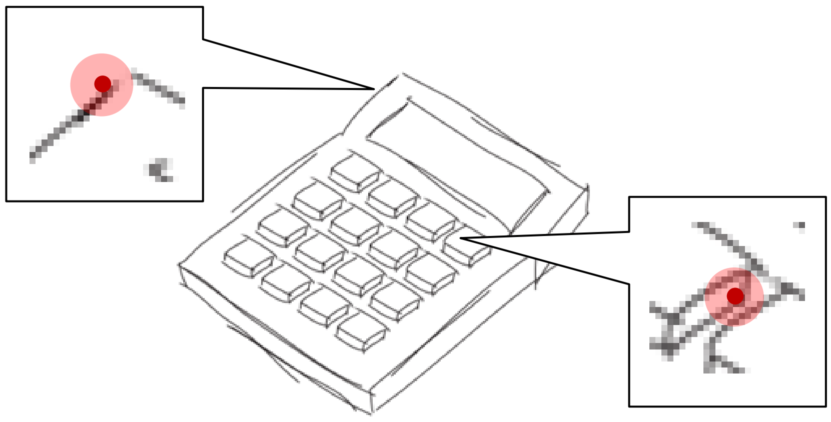


Figure 34: ***Problem of fixed neighbourhood.*** *Use of fixed neighbourhood may include unwanted directional vectors from neighbouring regions in dense area while missing directional vectors in sparse area.*

After identifying the target smoothing positions, we perform smoothing. In this step, we first perform clustering on the target smoothing points. For each cluster, we then evaluate its average position and adjust the position of its neighbouring points by calculating the weighted average of the position of the points and the cluster's center. The equation we use is:

$$P_i{}' = \frac{w \times P_i + (k - w) \times P_{center}}{w + (k - w)}$$

$$w = abs(i - j)$$

Equation 5: ***Adjustment of neighbouring points of target smoothing group***

where k is a parameter; P_i is the neighbouring point to be adjusted; i is the index of the point on region-boundary strokes and j is the index of its closest target smoothing point within the cluster; P_{center} is the center of the target smoothing cluster. In simple words, we are trying to pull neighbouring points towards the center of target smoothing group in order to smooth the synthesized strokes.

To determine how many neighbouring points we should adjust for each target smoothing group, every time, we adjust different number of neighbouring points and evaluate the smoothing performance of each adjustment. To evaluate the performance, for each point on the outline, we calculate the angle between its directional vector and the directional vector of the closest user input stroke. The adjustment with the smallest angle difference on average is selected as the final result.

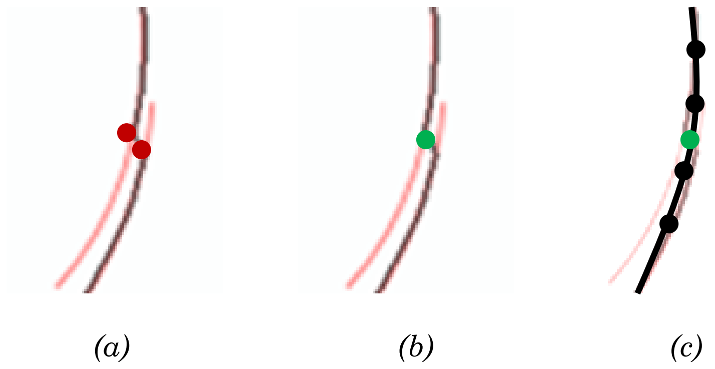


Figure 35: ***Summary of outline smoothing process.*** *(a) Identify target smoothing positions (b) Perform clustering on target smoothing positions and calculate the center of each cluster (c) Adjust neighbouring points on synthesized outline*

A summary of the steps in outline smoothing process is given in *Figure 35*.

4.3.3 Outline Synthesis of Feature Strokes

OBSERVATION We observe that feature strokes are important by themselves, any pruning or merging of these strokes can destroy the overall structure of the drawing. For example, in *Figure 31*, if we prune away or merge any hair that is close and continuous, the hair will be sparser than expected. Therefore, when synthesize feature strokes, we tend to be conservative. Feature strokes have two important characteristics. First of all, they must be found within regions (Note that the canvas is considered as one of the regions in sketch). This characteristic makes feature strokes distinguishable from region-boundary strokes but not unnecessary strokes (shown in blue in *Figure 31*) which should be pruned away. Fortunately, feature strokes have another important characteristic. The maximum distance between a point on the strokes and its closest region boundary point is generally longer than unnecessary strokes.

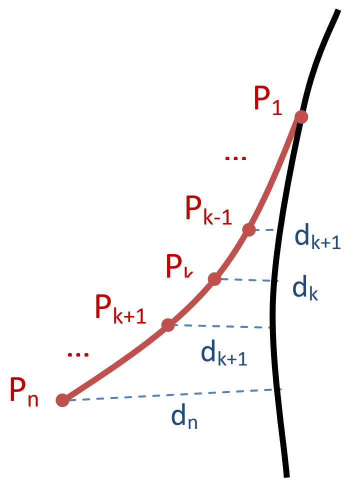


Figure 36: ***Identification of Feature Strokes.*** *Black stroke and red stroke represent the region-boundary strokes and segmented user input stroke respectively. If max(d_i) exceeds a threshold, the segmented user input stroke is identified as feature stroke.*

In Practice To preserve the structure of feature strokes, we make use of the user input strokes as the base of feature stroke synthesis. We cut the user input strokes when consecutive points on stroke lie on different regions in the sketch.

Then, for each segmented strokes, we measure the distances between points on strokes and their closest point on region-boundary strokes obtained in the previous step. If any of these distances is larger than a threshold, the stroke remains in the sketch and is identified as a feature stroke while the remaining strokes are pruned away. The measurement is demonstrated in *Figure 36* where P_i is a point on segmented user input stroke and d_i is its corresponding shortest distance from region-boundary strokes.

4.4 Curve Fitting

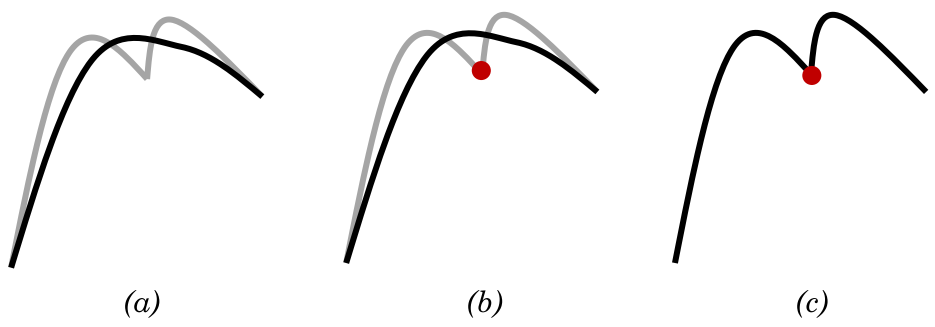


Figure 37: ***Iterative piecewise cubic Bezier curve fitting.*** *(a) Fit the synthesized outline using single cubic Bezier curve. (b) Find out the point on synthesized outline which deviates most from the fitted curve. (c) Cut the curve at that point and fit each sub-divided segment using cubic Bezier curve again.*

After outline synthesis module, we fit the outlines with piecewise cubic Bezier curve which is commonly used in design software. The fitting is performed by minimizing the squared error of the fitted Bezier curve. We start the fitting with one cubic Bezier curve first and then every time, we cut the outline at the position at which the error is the largest and fit the sub-divided segments again with cubic Bezier curves. The process is performed iteratively until the maximum squared error of the fitted curves is smaller than a threshold. The whole process is demonstrated in *Figure 37*.

Since we scale up the image at the very beginning of the refining process, the Bezier curves obtained are then scaled down before exporting as a vector image, which is a SVG file in our case. The output image can then be accessed and modified by other software tools such as Adobe Illustrator.

5. Result and Discussion

Table 2 shows some of the experiment results of our proposed method. All output images are overlaid on their corresponding input image for comparison. Our method generally performs quite well in refining the sketches. Among those examples given, example (d) is selected and discussed in detail in this section.

As discussed in the previous section, traditional outline tracing methods group strokes based on their proximity and continuity. However, these methods can be unreliable and some strokes may be grouped unexpectedly. This is demonstrated by example (d). From *Figure 38 (a)*, we observe that even though some strokes are close and continuous, for example the strokes circled in red, they should not be grouped. This grouping can be prevented by having a smaller tolerance on the closeness measure. However, this can lead to missed grouping in some other areas in the sketch such as the area circled in blue in *Figure 38 (a)*. *Figure 38 (b)* and *(c)* show two expected results of existing methods using different closeness measure. They contain either unwanted grouping or missed grouping of strokes. Comparing to the existing methods, our proposed method succeeds in pruning those unnecessary strokes while preserving the salient regions in sketches, as demonstrated in *Figure 38 (d)*.

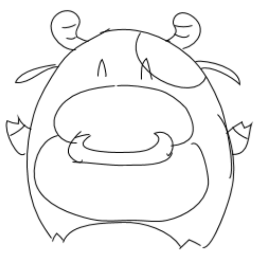
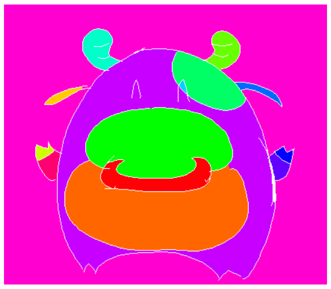
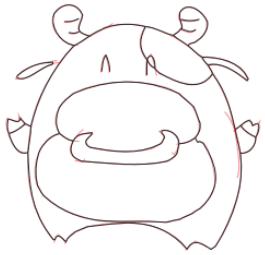
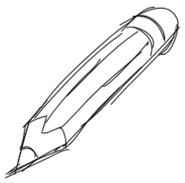
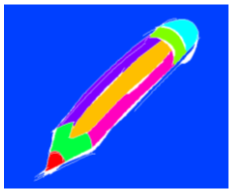
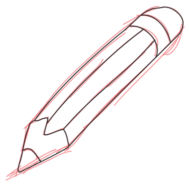
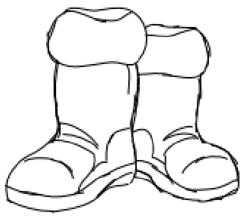
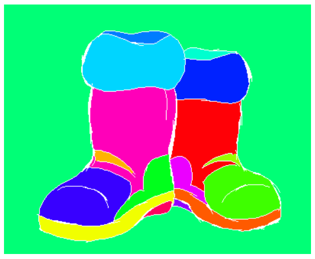
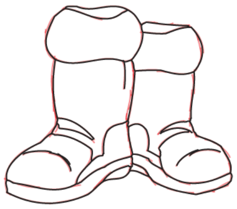
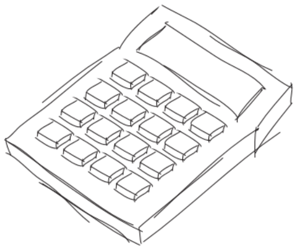
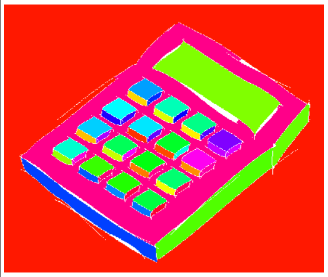
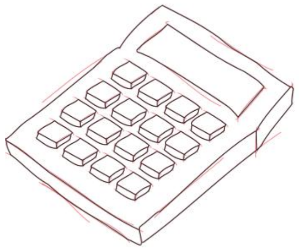
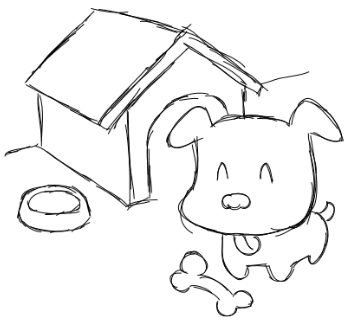
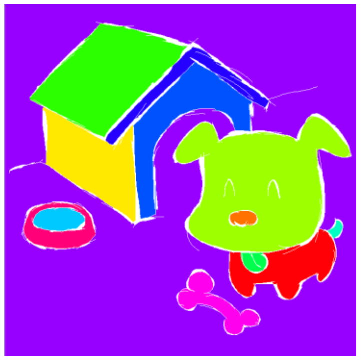
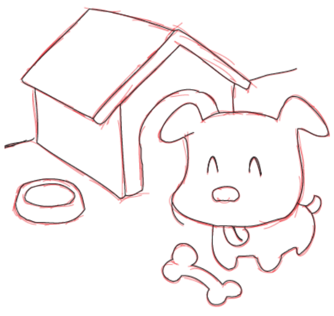
	Input	*Regions Identified*	*Output*
a			
b			
c			
d			
e			

Table 2: ***Experiment result.*** *Five test cases are shown. The first column shows the input sketches; The second column shows the regions obtained after salient region retrieval module; The third column shows the refined sketches which are vector images obtained using our method.*

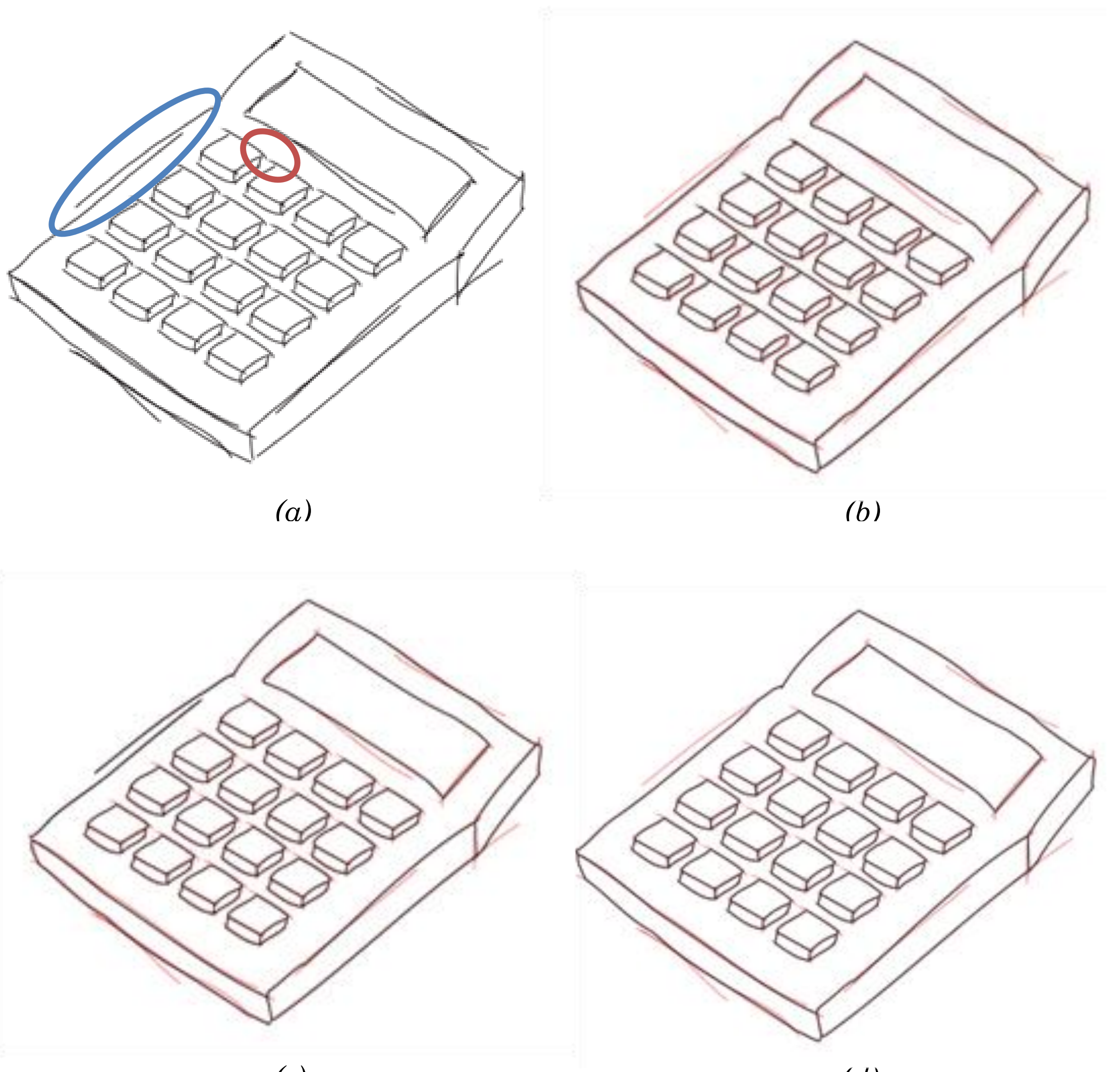


Figure 38: ***Comparison of the expected results of existing methods and our result.*** *(a) The red circle highlights the strokes which are close and continuous but should not be grouped while the blue circle highlights the strokes that should be grouped. (b) An expected output of existing methods with high tolerance of closeness measure. (c) An expected output of existing methods with low tolerance of closeness measure. (d) Our result.*

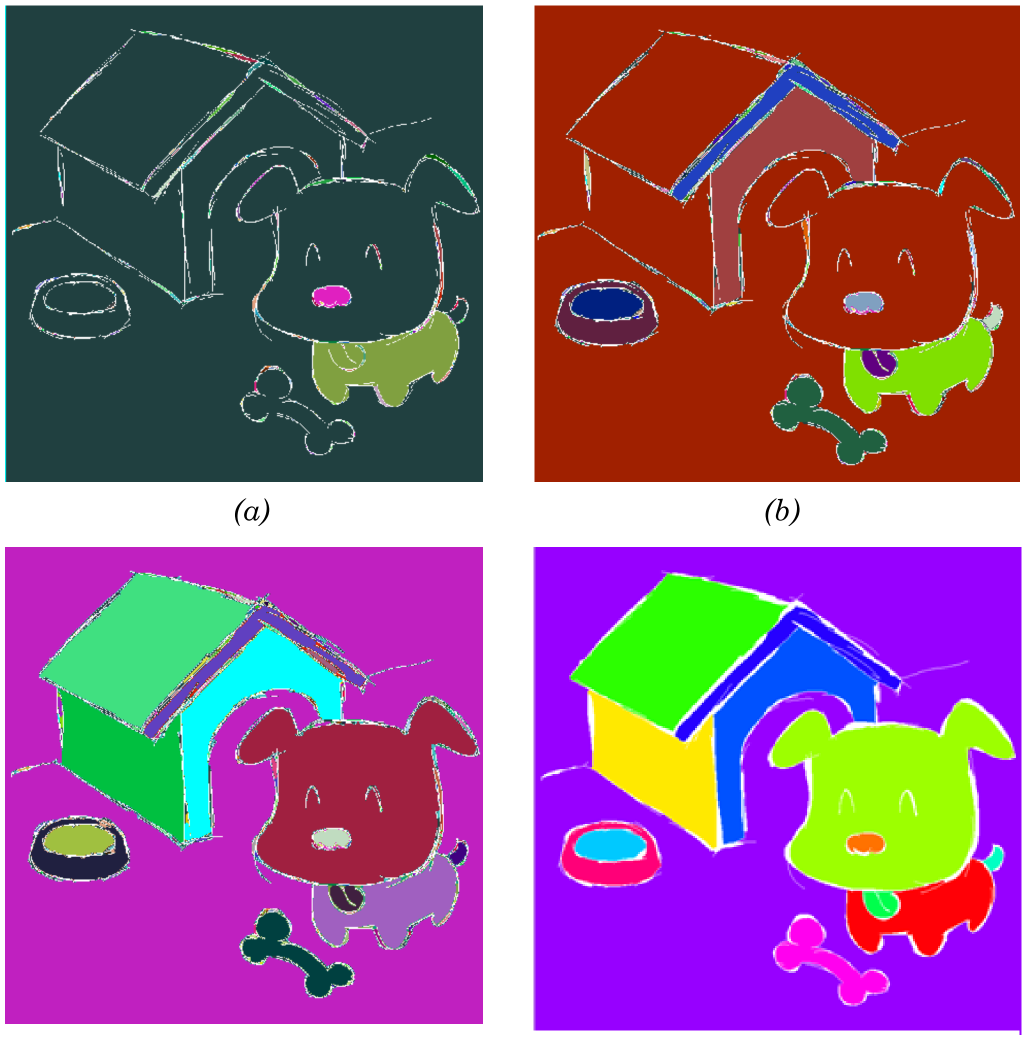

(a) (b) (c) (d)

Figure 39: ***Comparison of the regions obtained using flood-filling method, rolling ball method and our method.*** *(a) Flood-filling method produces severe leakage. (b) Rolling ball with radius=1 causes some minor leakage. (c) Rolling ball with radius=2 produces no leakage but some regions are separated into multiple regions. (d) Our method can prevent leakage without unwanted separation of regions. All non-salient regions are pruned.*

The main contribution of our work is the introduction of multi-scale regions retrieval method. *Table 2* shows the regions obtained using different input images.

Figure 39 shows a comparison of the regions obtained from example (e) using flood-filling method, rolling ball method and our method. Our multi-scale regions retrieval strategy outperforms the other methods in terms of the ability to prevent leakage and obtain classification of regions that is comparable to our visual perception. Although not the main aim of this work, the regions obtained not only can serve as a base for region refinement but also a preprocessing algorithm for other sketch processing applications such as colorization or texturing of sketches.

From the tests we performed on our algorithm, we conclude that our proposed method perform well in refining the region-boundary strokes. However, we may fail to retain some of the feature strokes as shown in *Figure 40 (a)*. This is unavoidable because it is sometimes hard to distinguish between unnecessary strokes and feature strokes if we do not know the expected shape of the object being drawn. For the same reason, we may also fail to prune away some of the unnecessary strokes as in *Figure 40 (b)*.

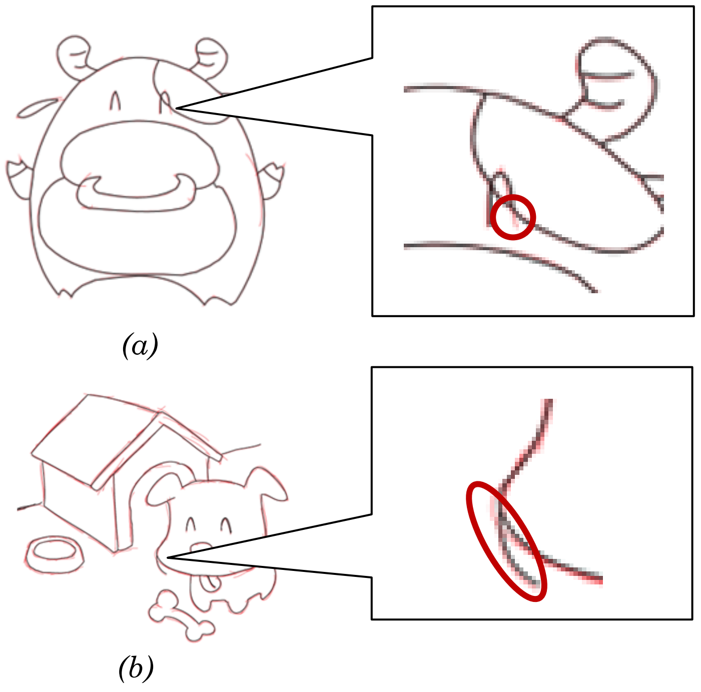


Figure 40: ***Failed cases of proposed method.*** *We fail to retain feature strokes or prune unnecessary strokes in a number of cases since it requires high level understanding of sketch. For example, when we perceive (a), we know that the circled stroke is part of the eye of the character. Therefore, we tend to retain it in the refined sketch. However, without high level understanding of the sketch, the stroke is indistinguishable from other unnecessary strokes. Thus, it is pruned away in our method. For the same reason, we fail to prune away long and deviated stroke in (b).*

6. Conclusion

Since outline tracing of sketches is an important but tedious process for artists and designers, there is a genuine motivation for its automation using computational methods. A good sketch-refining method should retain the basic structure of the drawing while avoid adding unnecessary structures. In this work, we discovered that regions provide important information in to sketch analysis since their shape and position give structure and meaning to strokes. However, it is hard to obtain closed regions from sketches by computational methods due to leakage of shapes. In this thesis, a multi-scale region retrieval technique is proposed to solve this problem. We make use of rolling ball method to build a hierarchy of regions followed by region merging and pruning. Experiment shows that this technique can successfully retrieve salient regions from sketches while removing regions that are non-salient to human visual perception. The closed regions obtained provide top-down information for analyzing sketches by computational algorithms. They help us to refine sketches by identifying and synthesizing region-boundary strokes and feature strokes in sketches and our method outperforms existing methods in terms of the ability to preserve salient regions and structures in sketches

7. Reference

[1] C. Stones, T. Cassidy. Seeing and Discovering: How Do Student Designers Reinterpret Sketches and Digital Marks during Graphic Design Ideation? Design Studies, vol. 31, issue 5, pp. 439-460, Sep. 2010.

[2] M. Suwa, B. Tversky. Seeing into Sketches: Regrouping parts Encourages New Interpretations. Visual and Spacial Reasoning in Design II, pp. 277-219, Sydney, Australia: Key Centre of Design Computing and Cognition, 2001

[3] L. G. Roberts. Machine Perception of Three-Dimensional Solids. Optical and Electro-optical Information Processing, J.T. Tippet (Ed.), MIT Press, pp. 159-197, 1965.

[4] R.O. Duda, P. E. Hart, Pattern Classification and Scene Analysis. New York: Wiley, 1971.

[5] G. Robinson. Edge Detection by Compass Gradient Mask. Comput. Graphics Image Processing, vol. 6, pp. 492-572, 1977

[6] C. Steger. An Unbiased Detector of Curvilinear Structures. IEEE Trans. Pattern Anal. Machine Intell., vol. 20, pp. 113-125, Feb. 1998.

[7] J. Canny. A Computational Approach to Edge Detection. IEEE Trans. PAMI 8(6), 1986.

[8] V.Torre, T.S. Poggio. On Edge Detection. IEEE Trans. Pattern Anal. Machine Intell., vol PAMI-8, no. 2, pp. 147-163, Mar. 1986

[9] W. Y. Ma, B. S. Manjunath. Edge Flow: A Framework of Boundary Detection and Image Segmentation. IEEE Trans. Image Processing, vol. 9, pp. 1375-1388, 2000.

[10] H. Kang, S. Lee, C. Chui. Coherent Line Drawing. Proc. ACM Symposium on Non-photorealistic Animation and Rendering, pp. 44-50, San Diego, CA, 2007.

[11] J. Bigün, G. H. Granlund. Optimal orientation detection of linear symmetry. Proc. First Int. Conf. on Computer Vision (ICCV '87), London, IEEE Computer Society Press, Washington, pp. 433-438, 1987.

[12] T. Baudel. A Mark-Based Interaction Paradigm for Free-Hand Drawing. UIST'94 Conference Proceedings, pp. 185-192, 1994.

[13] S.-H. Bae, W.-S. Kim and E.-S. Kwon, Digital Styling for Designers: Sketch Emulation in Computer Environment. ICCSA, pp. 690-700, 2003.

[14] G. Orbay, L.B. Kara. Beautification of Design Sketches Using Trainable Stroke Clustering and Curve Fitting. IEEE Trans. Visualization and Computer Graphics, vol. 17, no. 5, pp. 694-708, May 2011.

[15] L. Nan, A. Sharf, K. Xie, T.-T. Wong, O. Deussen, D. Cohen-Or and B. Chen. Conjoining Gestalt Rules for Abstraction of Architectural Drawings. ACM SIGGRAPH Asia 2011.

[16] Max Wertheimer. Laws of Organization in Perceptual Forms. Psycologische Forschung, vol. 4, pp. 301-350, 1923.

[17] B. Oh and C. Kim. Self-correctional 3D Shape Reconstruction from a Single Freehand Line Drawing. Lecture Notes in Computer Science, vol. 2669/2003, 983, 2003.

[18] Chen, X., Kang, S., Xu, Y., Dorsey, J., Shum, H.Y. Sketching Reality: Realistic Interpretation of Architectural Designs. ACM Transactions on Graphics, vol. 27, issue 2, no. 11, April 2008.

[19] S. Simhon and G. Dudek. Sketch Interpretation and Refinement Using Statistical Models. Eurographics Symposium on Rendering, pp. 23-32, 2004.

[20] B. Paulson and T. Hammond. PaleoSketch: Accurate Primitive Sketch Recognition and Beautification. Proceedings of the 13th International Conference on Intelligent User Interfaces, pp. 1-10, January 2008.

[21] P. Barla, J. Thollot, and F. Sillion. Geometric Clustering for Line Drawing Simplification. Proc. Eurographics Symp. Rendering '05, pp. 183-192, 2005.

[22] V. Vashisht, T. Choudhury, T. V. Prasad. Sketch Recognition using Domain Classification. IJACSA - International Journal of Advanced Computer Science and Applications, Nr. Special Issue (2011), pp. 1-9, 2011.

[23] S. Wang, M. Gao, L.Qi. Freehand Sketching Interfaces: Early Processing for Sketch Recognition. Human-Computer Interaction, Part II, HCII 2007, LNCS 4551, pp. 161-170, 2007.

[24] A. Severn, F. F. Samavati, J. J. Cherlin, M. C. Sousa and J. A. Jorge. Sketch-based Modeling and Assembling with Few Strokes. Sketch-based Interfaces and Modeling, ISBN 978-1-84882-811-7. Springer-Verlag London Limited, pp. 255, 2011.