國 立 交 通 大 學

資訊科學與工程研究所

碩 士 論 文

建構於高斯混合模型和區域成長的快速物件擷取工具

A Fast Image Cutout Tool Based on Gaussian Mixture Model and Region Growing

研 究 生：陳定樸

指導教授：林志青 教授

王任瓚 教授

中 華 民 國 一 百 零 二 年 六 月

建構於高斯混合模型和區域成長的快速物件擷取工具

A Fast Image Cutout Tool Based on Gaussian Mixture Model and Region Growing

研 究 生：陳定樸　　Student：Ting-Pu Chen

指導教授：林志青　　Advisor：Ja-Chen Lin

王任瓚　　Ran-Zan Wang

國立交通大學

資訊科學與工程研究所

碩士論文

A Thesis
Submitted to Department of Computer and Information Science
College of Electrical Engineering and Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Computer and Information Science

June 2013

Hsinchu, Taiwan, Republic of China

中 華 民 國 一 百 零 二 年 六 月

建構於高斯混合模型和區域成長的快速物件擷取工具

A Fast Image Cutout Tool Based on Gaussian Mixture Model and Region Growing

研究生：陳定樸　　指導教授：林志青 博士

王任瓚 博士

國立交通大學 資訊科學與工程研究所 碩士

摘要

本篇論文提出一個快速且精確的物件裁剪方法，讓使用者可以簡單經由拖拉一個包圍感興趣物件的矩形將物件精確切割出來。這項技術可讓使用者快速從圖片中擷取目標物件，作為圖片合成或編輯等應用中圖塊素材的來源。所提方法基於高斯混合模型來表示圖片的顏色分布，並設計一個迭代式區域生長演算法執行圖片切割運算。這個方法具下列優點：(1)操作方式十分簡便易學，使用者只需利用滑數拖曳一個包圍住目標物件的矩形即可。(3)物件剪裁具高度精確度。針對具明確邊界的物件或複雜輪廓的物件都能有效的裁剪出。本論文實作所提方法並與文獻中高效率的物件擷取方法 The GrapCut 做比較，實驗結果顯示所提方法無論在擷取物件的完整性或裁剪時間都優於 The GrapCut，顯示所提方法的可行性。

關鍵字： 影像切割，影像剪裁，區域成長，高斯混合模型。

A Fast Image Cutout Tool Based on Gaussian Mixture Model and Region Growing

Student : Ting-Pu Chen Advisor : Ja-Chen Lin

Ran-Zan Wang

Institute of Computer Science and Engineering

National Chiao Tung University

ABSTRACT

In this thesis a fast and accurate image cutoff method is developed. The method enables the users to clip object of interest out of an image, which is a useful tool for various applications such as image composition and/or editing. The proposed method represents the colors of an input image in Gaussian Mixture Model, and designs an iterative region growing based segmentation algorithm to draw out the target object. The proposed scheme has the following advantages: (1) the level of user interaction is low. The cut out operation is accomplished through simply drawing a rectangle encompassing the target object, and (2) the extracted objects are well-tailored. Both object with explicit contour and object with complicate contour can be extracted accurately. The proposed scheme is implemented and compared with the efficient object extraction method – the GrapCut. Experiment results show the proposed method exhibits higher performance than the GrapCut, both in the completeness of the extracted object and the computation time.

Keyword: Image Segmentation, Cut Out, Region Growing, Gaussian Mixture Model.

ACKNOWLEDGEMENT

The author is in hearty appreciation of the continuous guidance, discussions, and support from his advisors, Dr. Ja-Chen Lin, Dr. Ran-Zan Wang, not only the development of this thesis, but also in every aspect of his personal growth.

Appreciation is also given to the colleagues of the Computer Vision Laboratory in the Institute of Computer Science and Engineering at National Chiao Tung University for their suggestions and help during his thesis study.

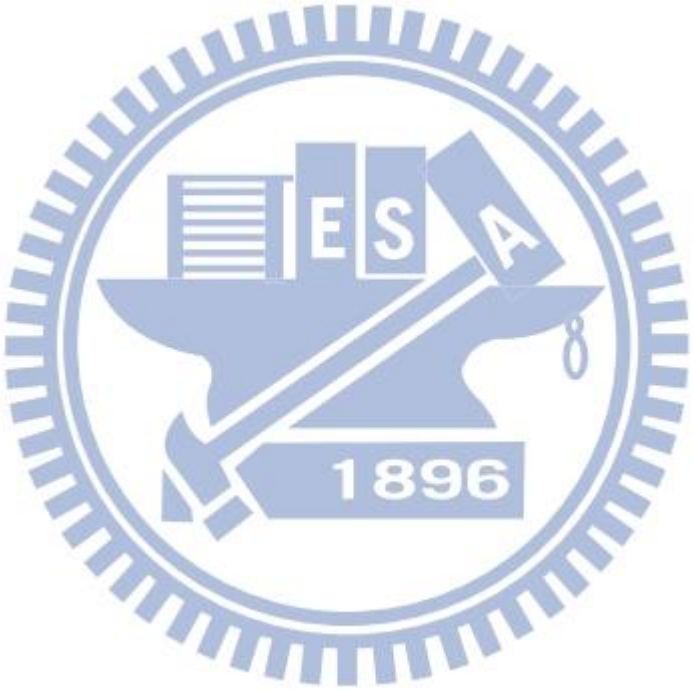
Finally, the author also extends his profound thanks to his dear family, mom, dad and sister, for their lasting love, care, and encouragement.

Ting-Pu Chen

National Chiao Tung University

June, 2013

CONTENTS



LIST OF FIGURE

LIST OF TABLE

Chapter 1 Introduction

1.1 Motivation and Background

Image segmentation [1,2] is the process about partitioning a digital image into multiple regions, in which the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture. The technique is typically used to locate objects and boundaries in images, which is one of the most important research topics in the fields of Computer Vision and Image Processing. In the past 50 years many efficient segmentation algorithms were proposed and applied for solving practical problems in many fields include pattern recognition, data clustering and medical imaging. Although many segmentation methods reported in the literature are efficient toward particular problems, an automatic algorithm to identify and extract well-defined objects from an image in the same way perceived by human eyes is a still big challenge in this research field, both from the requirements of execution time or the exquisite degree to the target objects.

The content of an image is usually composed of multiple scenes and objects; however, not all of the objects/regions on an image are interested to the users. An efficient tool for cutting out certain object/region from an image is interesting and useful in processing digital images. It provides plentiful materials for image applications such as composition and/or editing. In general, image cutout techniques can be classified into three categories: (1) Segmentation-based image grabbing method; (2) Boundary-based image cutoff algorithm; and (3) Image matting technique. The segmentation-based methods pick out the target object/region by selecting all pixels matching certain feature requirement such as similarity of color or texture. The popular object selection tool ‘Magic Wand’ in Photoshop [3] is one scheme based on this technique. It provides users a fast method to pick the neighborhood pixels having

similar color with a clicked pixel, and the process can be conducted iteratively to select the desired object. There are many semi-automatic cutoff tools proposed in the literature. In 2002 Reese and Barrett [4] proposed a method called Intelligent Paint. It first over-segments the image and then let the user to select the regions that form the foreground object manually. Lazy Snapping [5] and GrabCut [6] provide interactive min-cut-based cutout solutions. In both systems the action should be taken by the users is a coarsely indication step that drawing a few strokes to mark the foreground and background regions using the mouse, and the system will determine the boundary for segmenting the object using the proposed algorithm.

The second type of method for extracting a foreground object from an image is the boundary-based segmenting method. The method in this type usually tries to develop efficient algorithm to find the curves fitting to the boundary of the target object quickly and accurately. Because the process of drawing the boundary curve of an object manually using the mouse is a difficult and tedious job, the boundary tracing techniques such as the classic Active Contour Models [7], Intelligent Scissors [8], Image Snapping [9] and Jetstream [10] are proposed, these methods use curve optimization methods to significantly reduce the time and precision required in drawing the track of the boundary.

The last category of object extraction method is called Image Matting where pixels on the edge of a hard segmentation boundary often contain a mixture of the foreground and background. To seamlessly composite such mixed pixels requires the estimation to an opacity (alpha) value and foreground color for each pixel. Ruzon and Tomasi [11] showed how to estimate the alpha matte and foreground color using statistical methods. GrabCut [6] described border matting which assumes a strong prior model on alpha to quickly estimate a smooth matte.

In recent years the digital camera is replacing traditional film camera to take

photographs in the daily life of human beings, and many digital cameras are connected directly to the internet. The migration in hardware together with the popularity of the interactive multimedia networks has change the data communication style in our daily life. Sharing pictures on social media such as Twitters or Facebook has become a new trend and therefore, the needs of “airbrushing” which provides a handy tool for enriching the content or annotating marks on pictures are growing. To satisfy the user demands, a new image cutoff method which supplies an easy and quick way to segment object from an image for further processing is explored in this thesis.

1.2 Thesis Organization

The remainders of this thesis are organized as follows: the related works are reviewed in Chapter 2. The details of the proposed method are given in Chapter 3, and the experiment results are exhibited in Chapter 4. A brief conclusion and the future research directions are summarized in Chapter 5.

Chapter 2 Related Works

In this chapter, five related works about the proposed method are reviewed. We focus on the techniques in semi-automatic interactive image cutout tool where a coarse specification to foreground and/or background should be marked by the user manually. The methods include (1) The magic wand in Photoshop, (2) The Intelligent Scissors, (3) The Bayes Matting, (4) The Graph Cut, and (5) The Grab Cut.

2.1 The Magic Wand

It is a popular object selection tool supported in Adobe, which allows the user to select the area with similar color by drawing a point or a region. The user interface of the Magic Wand [3] is shown in Figure 2-1, which is the version from Adobe Photoshop 7. The process of selecting an area starts by marking a seed pixel on the image manually by the user, and the area grows from the seed pixel to find a region of connected pixels where all the selected pixels fall within certain tolerance of the color statistics of the specified pixel. The method is fast; however, it usually needs to specify many seed points for users to select the completely area of a target object.

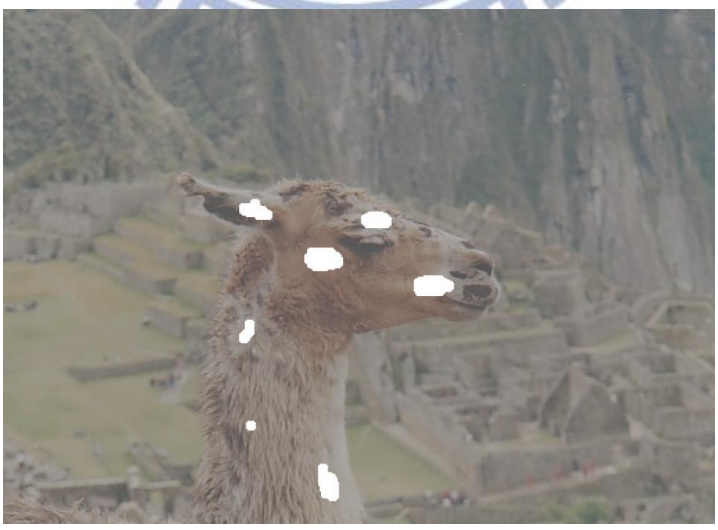


Figure 2-1 User interface of Magic Wand [3].

2.2 The Intelligent Scissors

Intelligent Scissors [8] is an object segmenting method proposed by Mortensen and Barrett in 1995. The working model of this method is shown in Figure 2-2. It allows the user to segment out the object from the background by roughly indicating the object's boundary using mouse points. The segmentation of the foreground object from background image is conducted by finding the "minimum cost contour" from the cursor position back to the last "seed" point. The process is analogy to the problem of solving a graph searching problem where the goal is to find the optimal path between a start node and a set of goal nodes. If the computed path does not reach the predefined requirement, additional user-specified seed points can be added to refine the result.

Figure 2-2 Working model of Intelligent Scissors [8].

2.3 The Bayes Matting

Image matting is the process of compositing two different images in a seamless blended image. In 2001 Chuang et al. [12] proposed a digital matting method based on the Bayesian theory which models color distributions probabilistically to achieve

full alpha mattes. The Bayes Matting [12] needs a "trimap" which classifies the pixels of an image in three types: (1) definitely foreground pixel T_F, (2) definitely background pixel T_B, and (3) uncertain pixel T_U, before extracting the image object. The trimap is generated by the user through marking out the definitely image foreground T_F and definitely background T_B with strokes of two different colors. The working interface is illustrated in Figure 2-3 where white color line segments represent the foreground and red color line segments are for the background. After the indication to the foreground and background, the color of the pixels in the remaining region T_U is determined by a compositing equation:

$$C = \alpha F + (1 - \alpha)B, \tag{2.1}$$

where F and B represent the color model of T_F and T_B, respectively, and α is the pixel's opacity component used to linearly blend between foreground and background, and C is the composition result.

Figure 2-3 Working model of Bayes Matting [12].

2.4 The Graph Cut

The Graph Cut [13] is a foreground object extracting technique that applies a similar setting to Bayes Matting, including the "trimaps" and probabilistic color

models. This method was first mentioned in 2001 by Boykov and Jolly [13], and its goal is to achieve robust segmentation even when the foreground and background color distributions of an image are not well separated. The working interface is similar to the configuration set in Bayes Matting, in which the user has to specify the foreground and background by using different colors of strokes as shown in Figure 2-4. In the illustration white color strokes represent the foreground and red color strokes are for the background.

After the user had pointed out the foreground and background, it defines a cost function called the "Gibbs" energy function. The "Gibbs" energy function as shown in Eq. 2.2 consists of two parts, one is to evaluate the degree of fitness *U* of the opacity distribution to the input data and the other is to calculate the smoothness term *V*:

$$E(\underline{\alpha}, \underline{\theta}, z) = U(\underline{\alpha}, \underline{\theta}, z) + V(\underline{\alpha}, z), \tag{2.2}$$

where $\underline{\alpha}$ is the opacity value associated with each pixel , $\underline{\theta}$ is the color model for the foreground and background defined by the user and *z* is the input data.

After the energy function is fully defined, it starts the segmenting process by finding a global minimum to the "Gibbs" energy function:

$$\underline{\hat{\alpha}} = arg \min_{\underline{\alpha}} E(\underline{\alpha}, \underline{\theta}) \ , \tag{2.3}$$

The minimization step is done by using a standard minimum cut algorithm proposed in the study of Boykov and Jolly [13].

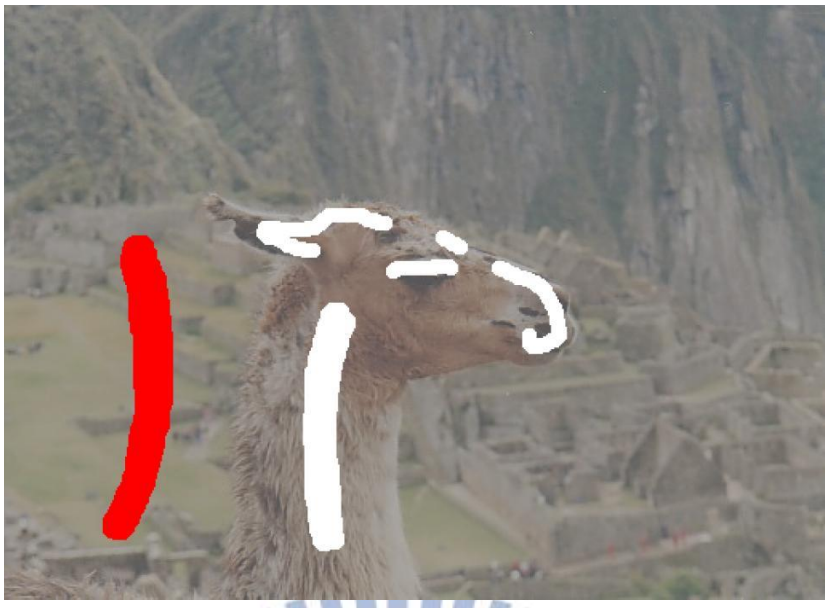


Figure 2-4 Working model of Graph Cut [13].

2.5 The Grab Cut

The object extraction method of The Grab Cut [6] is proposed by Rother et al. in 2004, which is an extension to the Graph Cut [13] described in Section 2.3. The method improves the Graph Cut method in three aspects. First, the monochrome image model represented in histogram is replaced by a Gaussian Mixture Model (GMM). Secondly, the one-shot minimum cut estimation algorithm is replaced by a more powerful, iterative procedure that alternates between estimation and parameter learning. Finally, the demands on region specification by the user are relaxed by allowing incomplete labeling. The user needs to specify only T_B for the "trimap", and this is done by simply placing a rectangle surrounding the object as illustrated in Figure 2-5.

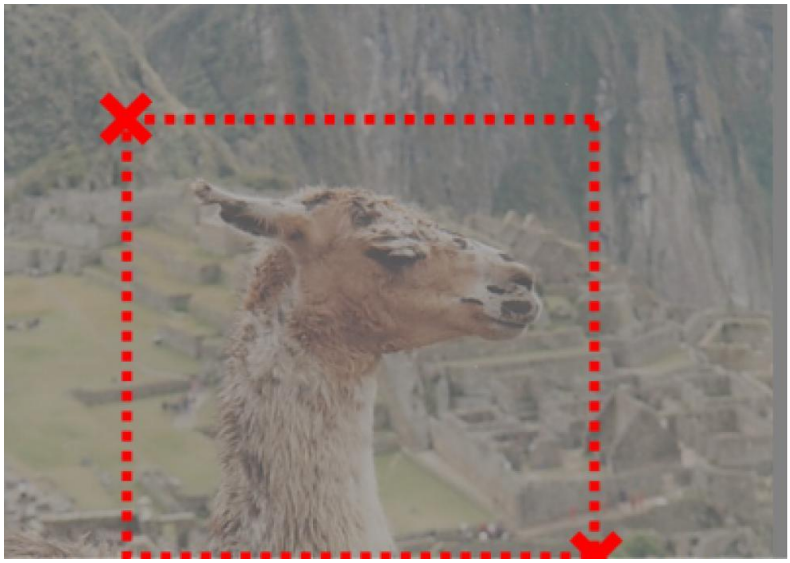


Figure 2-5 Working model of GrabCut [6].

Chapter 3 The Proposed Method

This chapter presents the details of the proposed object cutout method. Section 3.1 gives an overview to the proposed scheme. The working model and user interface are presented in Section 3.2. Section 3.3 gives the image data model in the proposed method, and the iterative segmenting algorithm is described in Section 3.4.

3.1 Method Overview

Before discussing the specific parts of the proposed cutout algorithm, an overview to the proposed method is presented. The proposed image cutoff process includes two main steps: (1) Image data modeling; (2) Iterative segmentation. The block diagram is shown in Figure 3-1.

The image data modeling step searches for a good representation for discriminating the foreground object from the background image. In the proposed scheme the user has to draw a rectangle encompassing the target object, which provides the information to build the Gaussian Mixture Model (GMM) in a hierarchical way as a representative of the color probability statistics for the object and background. The detail of this step is discussed in section 3.3.

The segmenting algorithm proposed in this study is an iterative region growing based method. It starts from the rectangle defined by the user and gradually extends the background region to fit the boundary of the target object. The process iteratively updates the GMM for representing the color distribution of the foreground object and background area, and the rectangle will shrink gradually to fit the boundary of the object. The detail is discussed in section 3.4.

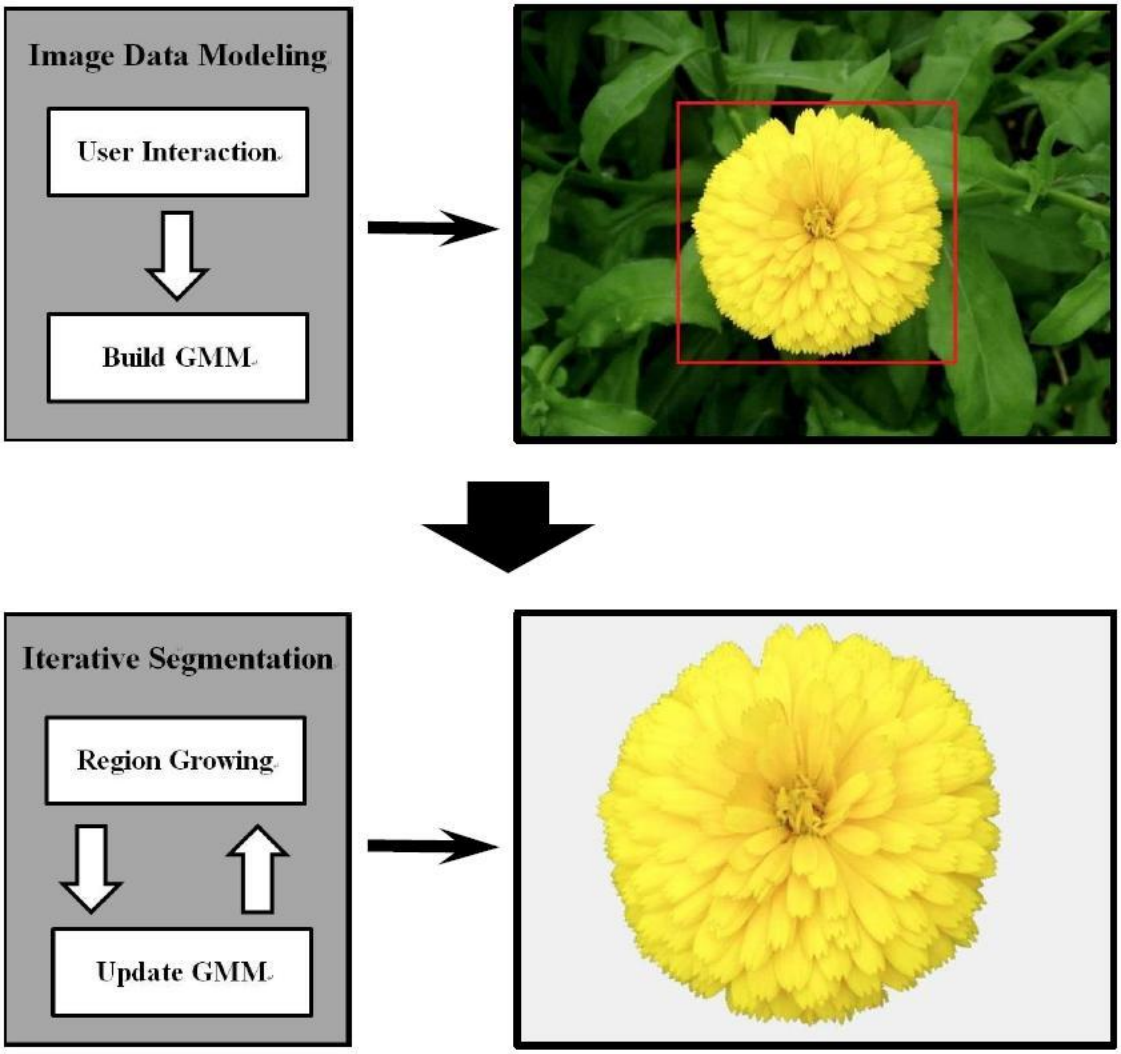


Figure 3-1 Block diagram of the proposed scheme.

3.2 User Interaction

Given an input image on which interested object is located, the goal of this study is to provide a simple and efficient tool for the user to select and cut out the object. The interaction between the user and the system should be as simple as possible, therefore makes the method feasible for novice user or in a low-resolution display device. The user interface for extracting an object from the input image in the proposed method is illustrated in Figure 3-2. The only one action should be taken by the user is drawing a rectangle box surrounding the target object using the mouse or other pointer device, and all of the remaining segmentation activities will be proceed

by the system automatically.

Figure 3-2 User interface of the proposed scheme.

The object specification method in the proposed scheme lessens the demand of the heavy input from the user to the system; however, it is a big challenge for the segmentation algorithm to produce a satisfactory result because of the inadequate information about the foreground object. In the proposed method, the area outside the rectangle box is the definitely background, while the area inside the box contains the target object as well as part of the background. The proposed method to build the foreground color model by using the user-defined background as a "guide" is discussed in section 3.4.

3.3 Image Data Modeling

A Gaussian Mixture Model (GMM) is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with different parameters. With this particular characteristic, the GMM can precisely describe the color distribution of an image because a picture is usually composed of several objects/regions with each object/region is dominated with certain

representative colors.

To generate a GMM, the first step is to decide the number of components K in the model. In other words, it is to choose the number of GMs associated in the GMM. After the number of components is decided, the parameters of each Gaussian Model (GM) should be determined. A GM can be characterized by θ_l:

$$\theta_l = \{\mu_l , \sum_l , l = 1,2, \dots, K\}, \quad (3.1)$$

where μ_l is the mean and the $\sum_l$ is the covariance matrix for the data. The Gaussian probability distribution of each pixel can be represented as:

$$P(z_n, \theta_l) = \frac{1}{(2\pi)^{d/2} |\sum_l|^{1/2}} exp\left(-\frac{1}{2}(z_n - \mu_l)^T (\sum_l)^{-1}(z_n - \mu_l)\right), \quad (3.2)$$

where d is the dimension of the data (which is 3 in RGB color space), and l = 1, 2, …, K. Also, the weight π_l of each Gaussian Model associated in the GMM should be decided as well.

An agglomerative hierarchical clustering method is designed to assign pixels with similar color attribute to the same GM in the proposed scheme. Agglomerative Hierarchical Clustering is a "bottom up" approach in which each data sample starts in its own cluster, and pairs of clusters are merged as one move up the hierarchy. Usually, it will take the two closest elements according to the chosen distance metric. After the merging operation, a distance updating process should be conducted to recalculate the distance between clusters. Each agglomeration occurs at a greater distance between clusters than the previous agglomeration, and one can decide to stop the process either when the clusters are too far apart to be merged or when the number of clusters reaches a predefined threshold. This clustering approach has the advantage of simple and high flexibility in the number of clusters compared with other methods.

The main goal of this clustering step is to find appropriate representative colors for the foreground object and background image, which is the basis in later segmenting activities. It is assumed that the input picture is a RGB image with n pixels $Z = \{z_1, z_2, \ldots, z_n\}$ in which both the foreground (the object to be cut out) and the background are composed of multiple principal colors, and the major colors of the foreground are discriminative from that of the background. The mean color is applied to represent the color of a group of pixels, and the Euclidean distance is employed to measure the distance between two groups of pixels.

The conventional Agglomerative Hierarchical Clustering Method is modified in two major ways to fit the requirements in the proposed scheme. (a) A protection set is added to postpone the merging time of two clusters with large quantity of pixels. (B) When a cluster is being merged in two candidate clusters with the same distance, it tends to be merged in the cluster with larger quantity.

The steps of the proposed clustering algorithm are summarized in Figure 3-5. In the proposed method, each color is initialized as a separate cluster. The count of each color is calculated and the K clusters with highest number of pixels are put in the protection set **S**. Because the algorithm is processed in RGB color space, the full set **C** will be:

$$\mathbf{C} = \{c_{0,0,0}, c_{1,0,0}, \ldots\ldots, c_{255,255,255}\}, \tag{3.3}$$

because the RGB value of each pixel range from 0 to 255. The use of the protection set **S** is to avoid merging clusters with large pixel count too quickly, and preserves the main components/colors of the image.

The Proposed Agglomerative Hierarchical Clustering Algorithm

Input

$Z = \{z_1, z_2, \ldots, z_n\}$: the input RGB Image

K: the maximum number of clusters

S: protection set

Output

t ($\leq K$) clusters of pixels $\{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_t\}$. Each pixel in Z belongs to a cluster.

Step 1: Count of pixel number for each color in Z, and initialize each color to a cluster. $\mathbf{C}=\{c_{r,g,b}\}$, $r,g,b = 0, 1, \ldots, 255$, where $c_{r,g,b}$ represents the number of pixels with color value (r, g, b).

Step 2: Sort the elements of **C** in non-increasing order.

Step 3: Set the top K elements of **C** to the protection set **S**.

Step 4: Sequentially fetch a not-process-yet cluster c_i from the top of **C**, and merge the cluster c_j to c_i, $i \neq j$ if (1) the distance between the two clusters is smaller than merge radius threshold R, i.e., $\text{dist}(c_j, c_i) \leq R$, and (2) c_j is not in **S**.

Step 5: Repeat step 4 until all the clusters in **C** are processed.

Step 6: Update the centroids of the new clusters, and set the threshold $R = \rho \times R$, where $\rho \geq 1.0$ is a user-defined radius increasing rate.

Step 7: Repeat steps 2 to 6 until the number of the clusters in **C** is smaller than $\beta \times K$, where β is the control parameter for the number of candidate clusters.

Step 7: Set the merge radius R to initial value.

Step 8: Repeat steps 2 and 6 without considering the Protection Set until the the clusters in **C** is smaller than K.

Step 9: End of algorithm.

Figure 3-3 The proposed agglomerative clustering algorithm.

After the protection set is determined, the adjacent clusters are merged together to get a new cluster. The proposed method merges all clusters c_j to cluster c_i, $i \neq j$ if the distance between the two clusters is smaller than merge radius threshold R, and c_i and c_j are not in **S** simultaneously. The order of the merging procedure is done from the biggest cluster to the smallest one, which can help the clusters to lessen the

influence from the outliers because bigger clusters usually have higher resistance against noise. When all of the clusters in $\mathbf{C}$ are merged, the centroids of the merged clusters are updated, and the radius R which set the threshold for merging two clusters is increased with rate ρ:

$$R_{new} = \rho \times R_{old} \ . \tag{3.4}$$

The cluster merging process repeats until the number of clusters are less than $\beta \times K$, where β is a user-defined parameter that controls the condition to stop the clustering process. Step 7 will reset the merge radius R to the initial value and start clustering again without the constraint of the protection set $\mathbf{S}$. The clustering procedure terminates when the number of clusters is equal to or smaller than the threshold K.

3.4 Iterative Segmentation

The proposed image cut out method is an iterative region growing based segmentation method. It can adaptively adjust the color model to better represent the color distributions of the image. Region growing [14] is a region-based image segmentation method and it is also classified as a pixel-based image cutout technique since it involves the selection of initial "seed points". The main goal of region growing is to partite an image into regions by examining neighboring pixels of the initial "seed points" and determine whether the pixel neighbors should be added to the region. To decide whether to join a pixel into the region, a logical predicate $\boldsymbol{L}$ is specified which in our case, is the color distribution model.

The basic formulation of region growing is shown in Figure 3-6 which is defined by R. C. Gonzalez and R.E. Woods. Part 1 & 2 shows that the segmentation of the image must be complete, which means that every pixel must be in a region and all the pixels inside the region must be connected. The next part indicates that all the regions must be disjoint. Part 4 deals with the properties that must be satisfied by the pixels in

a segmented region. For example, $\boldsymbol{L}(R_i)$ = TRUE if all the pixels in R_i belongs to the same color model. The last part shows that region R_i and R_j is different in the sense of the logical predicate ***L***.

Before perform the cutout algorithm, the clustering method proposed in Section 3.3 is applied to build two Gaussian Mixture Models to represent the color distributions of the foreground and background. The parameters that decide a model is an extension from (3.1):

$$\theta(\alpha, l) = \{\mu(\alpha, l),\ \sum(\alpha, l)\ , \pi(\alpha, l), \alpha = 0, 1, l = 1 \ldots \ldots K\}, \qquad (3.5)$$

where α is 0 for the background, and 1 for the foreground. $\pi(\alpha, l)$ represents the weight of the Gaussian Model inside the GMM:

$$\pi = \frac{The\ number\ of\ pixels\ in\ this\ Gaussian\ Model}{Total\ number\ of\ pixels\ in\ the\ Gaussian\ Mixture\ Model}. \qquad (3.6)$$

We assign a parameter α_n to each pixel to represent the color model that it is related to, 0 stands for the background and 1 is for the foreground. The classifying principle is decided by the user-defined area where the region outside the rectangle box is the background (α_n = 0) and the region inside is the foreground (α_n = 1).

Region Growing Segmentation Model

Parameters

- The regions **R**
- A logical predicate **L**

Basic Formulations

1. $\bigcup_{i=1}^{n} R_i = R$
2. R_i is a connected region, $i = 1,2,\ldots,n$
3. $R_i \cap R_j = \varnothing,\ \forall i = 1,2,\ldots n,$ and $i \neq j$
4. $L(R_i) = TRUE,\ i = 1,2,\ldots n$
5. $L(R_i \cup R_j) = FALSE$ for any adjacent region R_i and R_j

Figure 3-4 Basic formulation of Region Growing.

The proposed region growing based iterative cutout algorithm is given below:

Region growing based iterative mage cut out algorithm

Initialization

- Assign α_n to each pixel to represent the model that it is related to, $\alpha_n = 0$ for background (outside the box) and α_n = 1 for foreground (inside the box).
- Use the Agglomerative Hierarchical Clustering presented in Section 3.3 to build two GMMs, one for the background ($\alpha_n = 0$) and the other for the foreground ($\alpha_n = 1$).
- Initialize the set **S** with the pixels z_n that user defined as seed points (the pixels on the rectangle drawn by the user).

Iterative segmentation

Step 1: For each pixel z_n in **S**, assign it to the closest GMM:

$$\alpha_n := \arg\max_{\alpha} \pi(\alpha, l) \times P\big(z_n, \theta(\alpha, l)\big).$$

Step 2: If the newly assigned α_n belongs to the background, put the neighbors (inside the box) of the pixel z_n into the set **S**. Repeat step1 until **S** $= \emptyset$.

Step 3: Adjust the GMMs by the result of α_n and repeats the above steps until no pixels are re assigned to the background, i.e., all α_n are not changed.

Figure 3-5 The proposed image cutout algorithm.

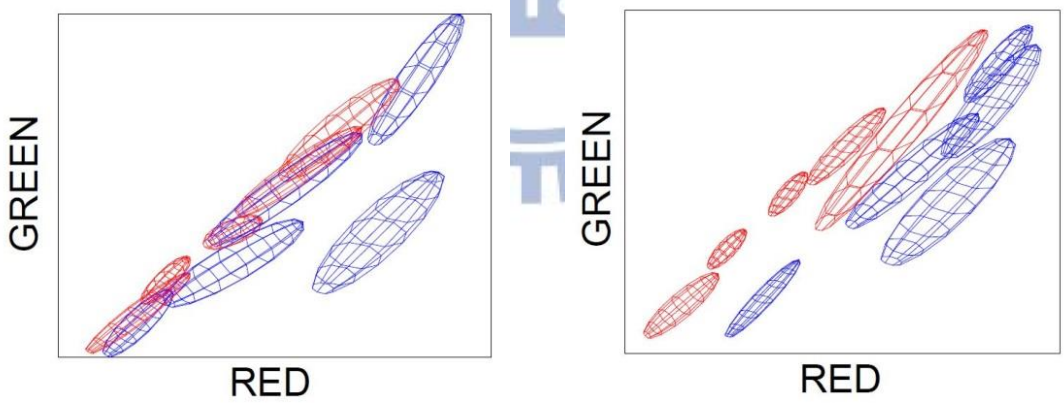
The proposed algorithm first initializes a set **S** with the pixels z_n that lies on the user marked rectangle as seed points for the iterative region growing process. For each pixel z_n in set **S**, the Gaussian probability distribution function of each Gaussian Model is evaluated to determine the likeness probability the pixel. The following rule are evaluate to assign pixel z_n to α_n:

$$\alpha_n := \arg\max_{\alpha} \pi(\alpha, l) \times P(z_n, \theta(\alpha, l)). \quad (3.7)$$

The above process can decide whether the pixel belongs to the background ($\alpha_n = 0$) or the foreground ($\alpha_n = 1$).

After assigning each pixel to the Gaussian Mixture Model, if the newly assigned α_n of each pixel belongs to the background (α_n = 0), put the neighbors (pixels that are inside the box) of the pixel z_n into the set **S** and repeat the assigning process until the set **S** is empty (**S** $= \emptyset$).

The next step is to adjust the Gaussian Mixture Models by using the result of the previous assigning approach. The effect of this step is to make the color model more similar to the color distribution of the image. As we used the region inside the user-defined box to represent the foreground, there is a proportion of the background included. To exclude the background, we need to iteratively execute the segmentation algorithm to make the color model more approximate to reality. The algorithm will terminate when the result of α_n does not change anymore. Figure 3-8(a) shows an example of two 5-component GMM in the R-G color space. As you can see, the two GMM overlap considerably. However, after the iterative segmentation, the results of the GMM are much better separated (Figure 3-8(b)).



(a) (b)

Figure 3-6 Evolution of the Gaussian Mixture Model. (a) The original Gaussian Mixture Model. (b) The result of the Gaussian Mixture Model after segmentation.

Chapter 4 Experiment Results

In this chapter, the experiment results of the proposed method are tested and discussed. The hardware used to implement the test program is a notebook with Intel Core i5-3210 2.50 GHz CPU and 4 GB DDR3 RAM. The program was written using C# in Microsoft .Net Framework 4.0 and ran in the Windows 8 system. All of the test images are in JPG file format.

Two types of target objects are cut off from an image using the proposed scheme. The first kind of object exhibits explicit contour and the boundary of the object can be clearly recognized by human eye. The second kind of object consists of complicated contour such as human hair or animal fur, which is difficult to segment form the background even when the operation is done by the user manually in contemporary image processing software. The cutout experiments of the two kinds of objects are shown in Section 4.1 and Section 4.2, respectively. To demonstrate the performance of the proposed scheme, the cutout results of the proposed scheme is compare with the result obtained in the popular image cutout tool GrabCut [6].

4.1 Cutout Object with Explicit Contour

Four test images with explicit object contours include (a) yellow flower, (b) orange flower, (c) metal object, and (d) building, are tested in this experiment. The following experiments exhibit the results for cutting out these target objects from the input images.

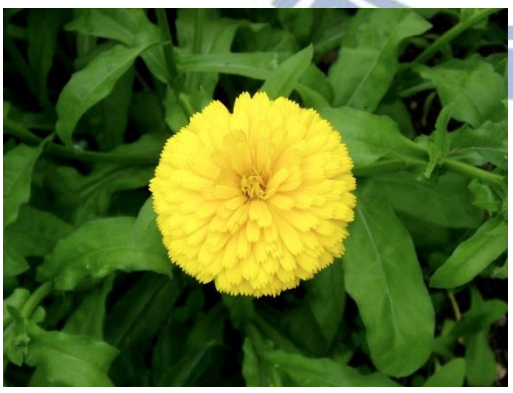
A. Yellow flower

The first experiment aims to cut out the yellow flower from the input image shown in Figure 4-1(a), which scale is 1024×768 pixels. Figure 4-1(b) shows the rectangle drawn by the user, which is done by moving the mouse to the left-top corner

to press down the mouse button, and drag the mouse to the right-bottom corner and press the mouse button again. It encloses the target object “yellow flower”. The cut out result using the proposed scheme is shown in Figure 4-1(c) and the result obtained in the GrabCut technique is shown in Figure 4-1 (d). It can be seen from the two images that both the two schemes successfully segment the flower from the background. The extracted flower object is complete and has clear-cut boundary, while the object extracted using GrabCut over-segment a little green area to the flower as shown in Figure 4-1(d) with a blue box. To test the execution speed of the two methods, four versions of the test image with different resolutions/sizes were designed and the processing times for cutting out the object were measured. Table 4-1 lists the execution times for the two methods. It can be seen that the processing time of the proposed method is faster than that of the GrapCut in all cases.

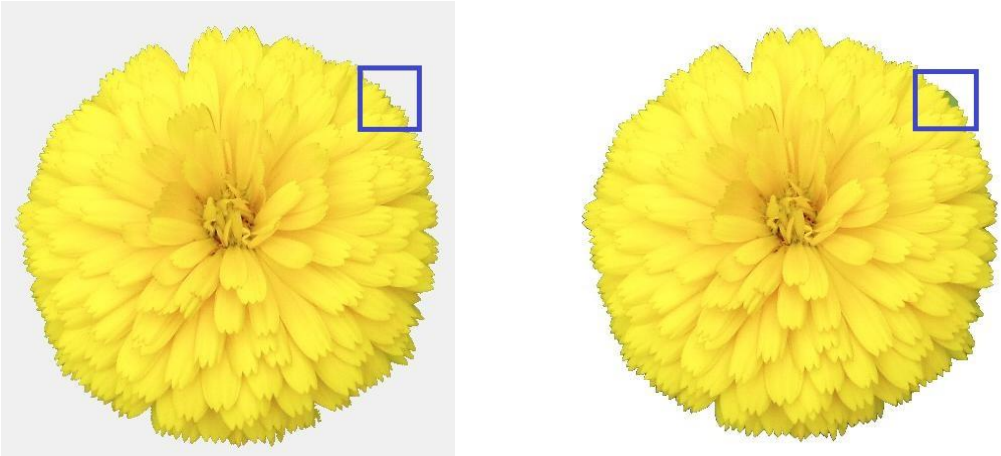
Table 4-1 Execution time of various image resolutions using yellow flower

Sequence	256×192	512×384	768×576	1024×768
GrabCut	1.76 sec	3.11 sec	6.82 sec	13.55 sec
Our Method	1.10 sec	2.82 sec	4.54 sec	6.31 sec



(a)

(b)

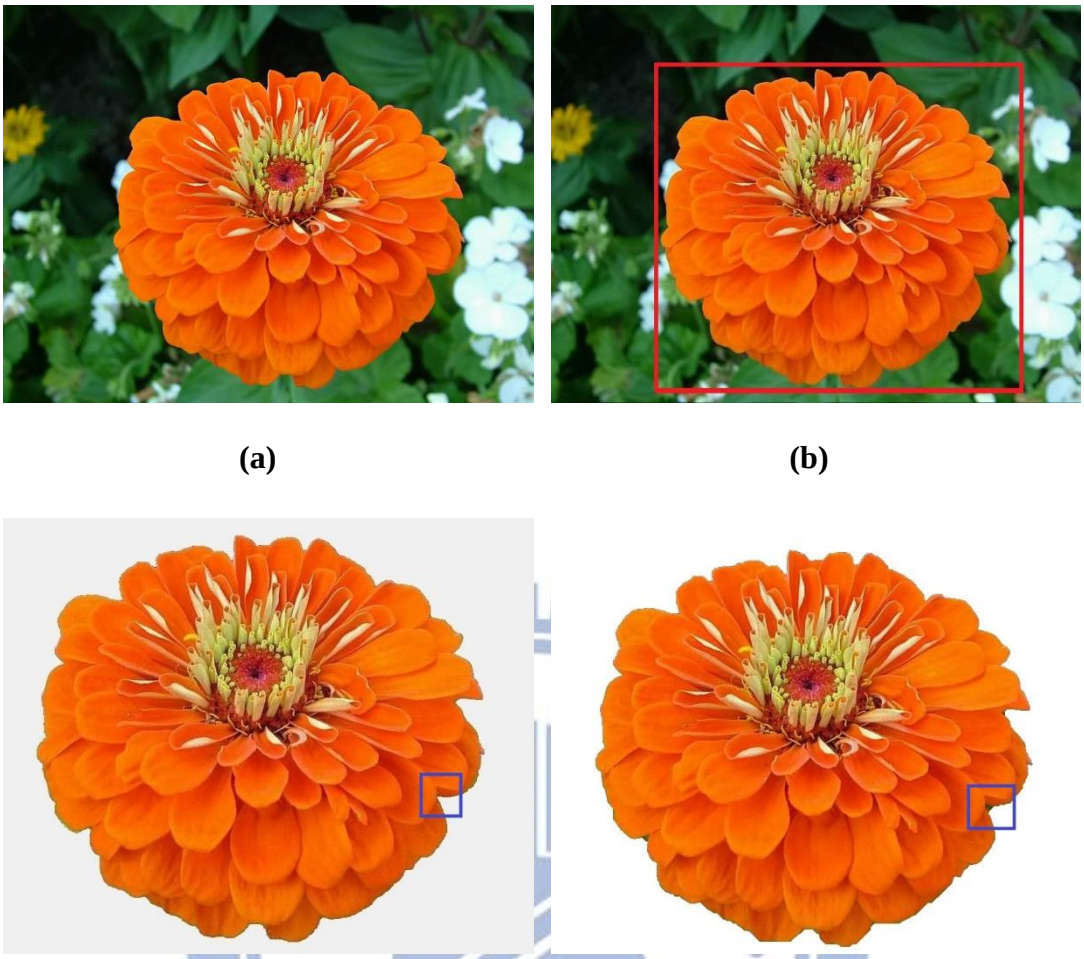


(c) **(d)**

Figure 4-1 Result comparisons using yellow flower. (a) The original image. (b) User labeling (red rectangle box). (c) Our result. (d) Result of GrabCut.

B. Orange Flower

The test image in this experiment is shown in Figure 4-2(a), its size is 1024×678 pixels and the background is more complicated compared with the previous tested image. The rectangle specified by the user is shown in Figure 4-2(b), and the segmented out object is exhibited in Figure 4-2(c). Figure 4-2(d) shows the cut out object in GrabCut. It can also be seen that the segmentation result of the proposed scheme is better than that in GrabCut, in that the contour of the segmented object in the proposed method is closely matched to the flower while the object segmented using the GrabCut contains a little green leaf (shown in Figure 4-2(c) and 4-2(d) with a blue box). In this test, the execution times for cutting out the flower from different resolutions of the input image were measured and listed in Table 4-2. Therefore, the execution time of the results had increased in both methods.



(a) **(b)**

(c) **(d)**

Figure 4-2 Result comparisons using orange flower. (a) The original image. (b) User labeling (red rectangle box). (c) Our result. (d) Result of GrabCut.

Table 4-2 Execution time of various image revolutions using orange flower

Sequence	256×192	512×384	768×576	1024×768
GrabCut	1.12 sec	5.13 sec	11.42 sec	19.04 sec
Our Method	1.54 sec	2.74 sec	7.72 sec	13.86 sec

C. F22 Jet Fighter.

The original image of the F22 Jet Fighter is shown in Figure 4-3(a) and the resolution size is 1020×678. The main purpose of using this picture is to test whether the cutout results will be affected by the sun. As being irradiated by the sunlight, the border of the wings became very similar with the cloud of the background which

may cause a bad cutout result. The results of the two methods out of this picture are very similar (see Figure 4-3(c) and Figure 4-3(d)). However, the execution time of our method performed better than GrabCut (see Table 4-3).

Table 4-3 Execution time of various image revolutions using F22 jet fighter

Sequences	255×170	510×339	765×509	1020×678
GrabCut	1.80 sec	6.42 sec	11.51 sec	23.78 sec
Our Method	1.27 sec	4.48 sec	9.20 sec	14.68 sec

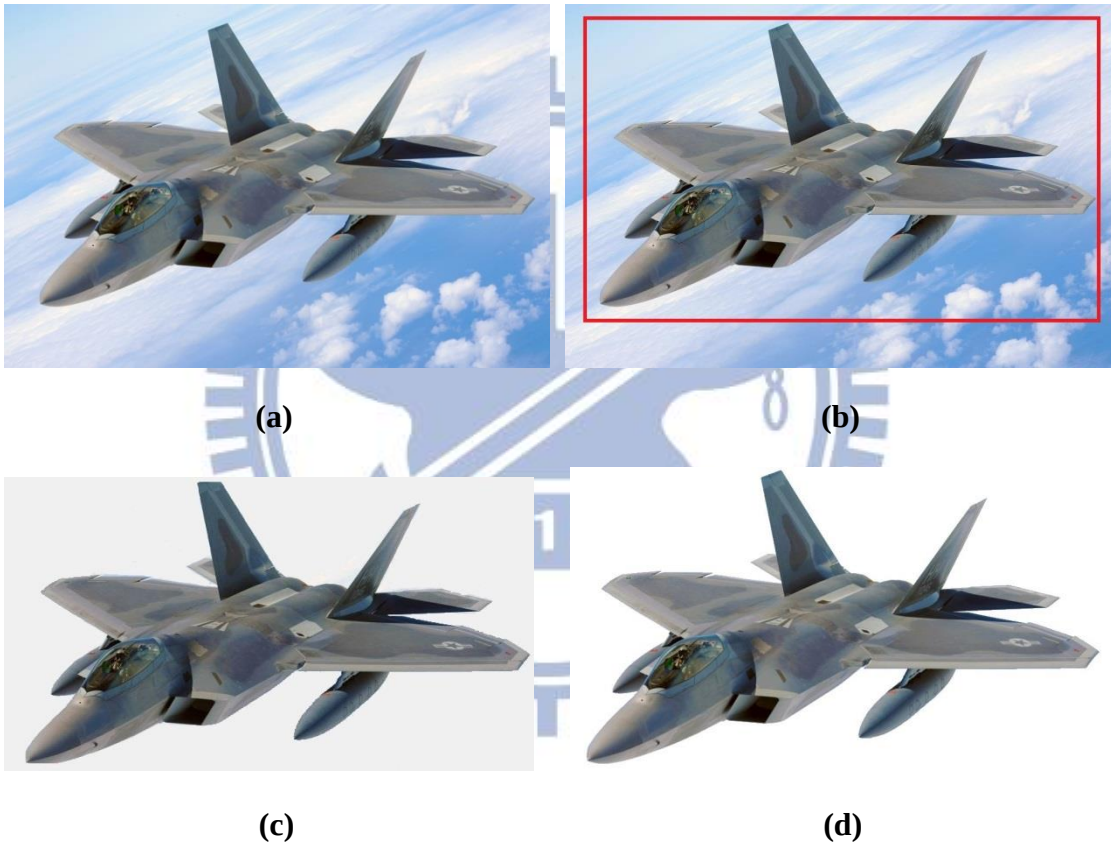


Figure 4-3 Result comparisons using F22 jet fighter. (a) The original image. (b) User labeling (red rectangle box). (c) Our result. (d) Result of GrabCut.

C. Big Ben Clock

Compared with the test images (1000×562) we used in the previous experiments, the proportion of the object inside the picture is much smaller (see Figure 4-4(a)). The smaller sized object may cause the segmentation results to lose its exquisite details.

The main difference of the results is shown in Figure 4-4(c) and Figure 4-4(d) with the blue box. GrabCut had failed to cut out the pointed tower at the top of the Big Ben Clock. In contrast, our method had a much better result both in execution time (see Table 4-4) and the cutout result.

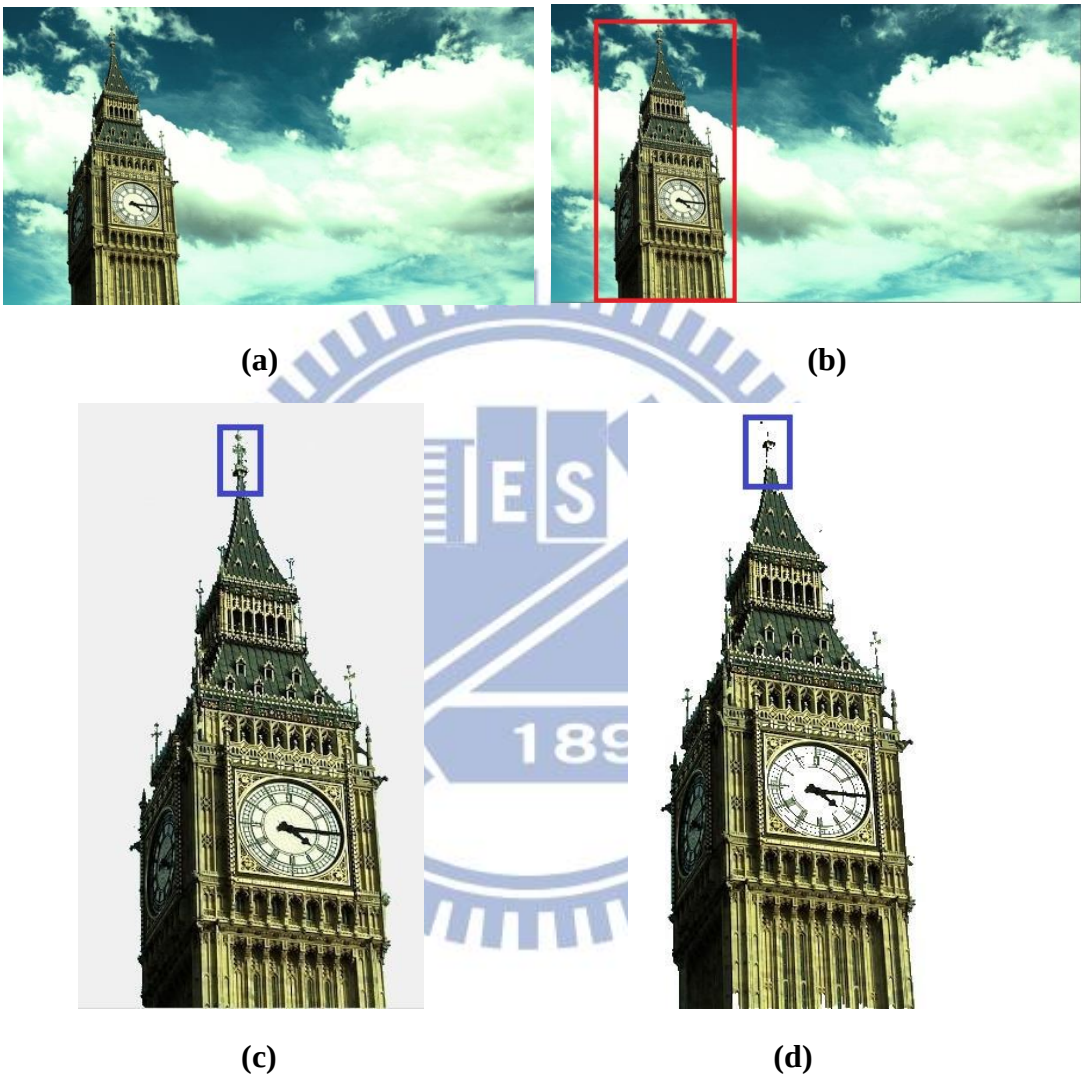


Figure 4-4 Result comparisons using Big Ben. (a) The original image. (b) User labeling (red rectangle box). (c) Our result. (d) Result of GrabCut.

Table 4-4 Execution time of various image revolutions using Big Ben

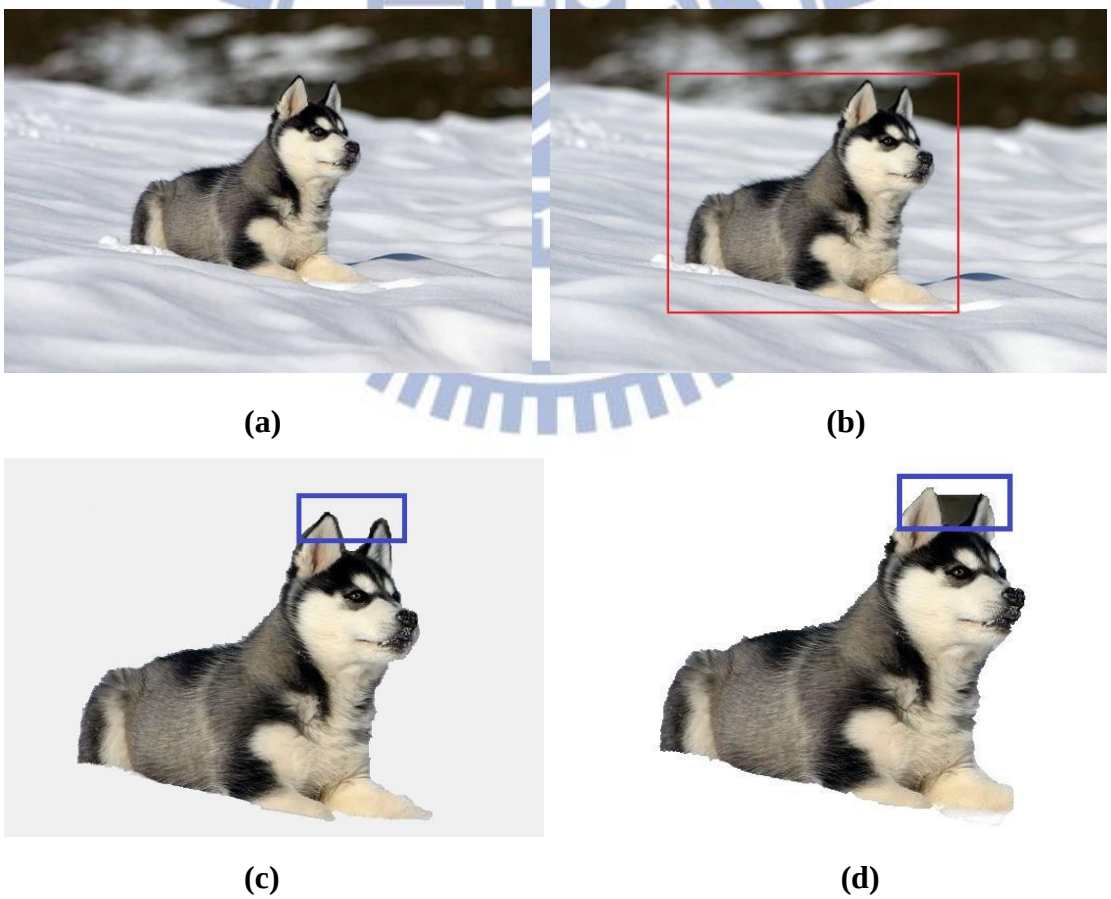
Sequence	250×141	500×281	750×422	1000×562
GrabCut	1.33 sec	4.75 sec	11.37 sec	20.47 sec
Our Method	1.32 sec	3.91 sec	6.92 sec	10.33 sec

4.2 Cutout Object with Complicated Contour

In this part, we will test some images with complicated contours which in other words, objects that have ambiguous edges such as animal fur, human hair, etc.

A. Husky

The Husky image we use in this experiment is a 1024×705 resolution size picture as it's shown in Figure 4-5(a). The fur of the Husky's leg is really similar to the snow, which makes it very difficult for the segmentation algorithms to perfectly cutout the Husky's body. The comparison of the results is shown in Figure 4-5(c) and Figure 4-5(d) with the blue box. GrabCut failed to remove the background near the ears and some snow is left around the front leg. However, the execution time of the two methods is quite close compared with the previous experiments (see Table 4-5).



(a) **(b)**

(c) **(d)**

Figure 4-5 Result comparisons using Husky. (a) The original image. (b) User labeling (red rectangle box). (c) Our result. (d) Result of GrabCut.

Table 4-5 Execution time of various image revolutions using Husky

Sequence	256×177	512×353	768×529	1024×705
GrabCut	1.34 sec	2.22 sec	4.44 sec	12.49 sec
Our Method	1.16 sec	2.69 sec	5.30 sec	9.14 sec

B. Tiger

The comparison of the methods is shown in Figure 4-6(d) with blue circles where GrabCut failed to keep the cutout object in one piece. Nevertheless, because the background of the image is so "flat", it lessened the time of the min-cut procedure for GrabCut. Therefore, the execution time (see Table 4-6) of GrabCut is really close to our method.

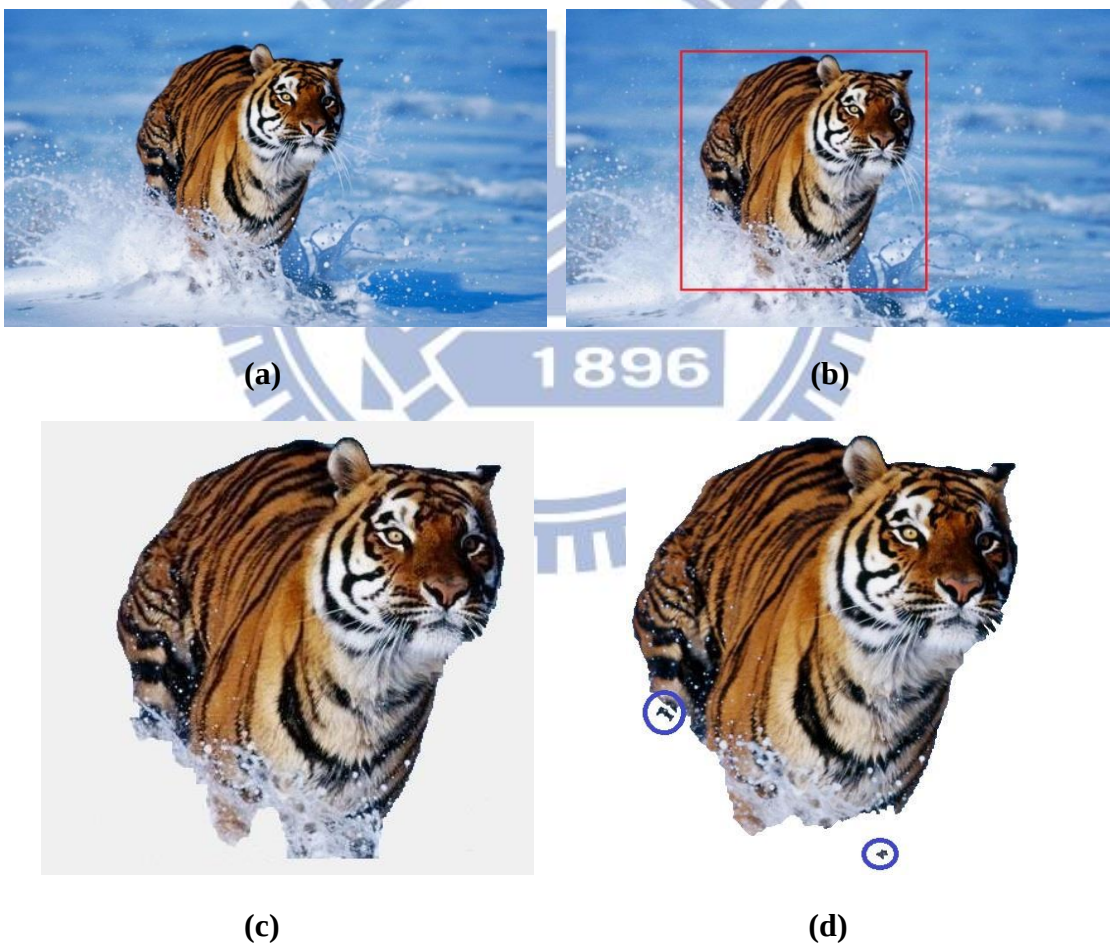


Figure 4-6 Result comparisons using Tiger. (a) The original image. (b) User labeling (red rectangle box). (c) Our result. (d) Result of GrabCut.

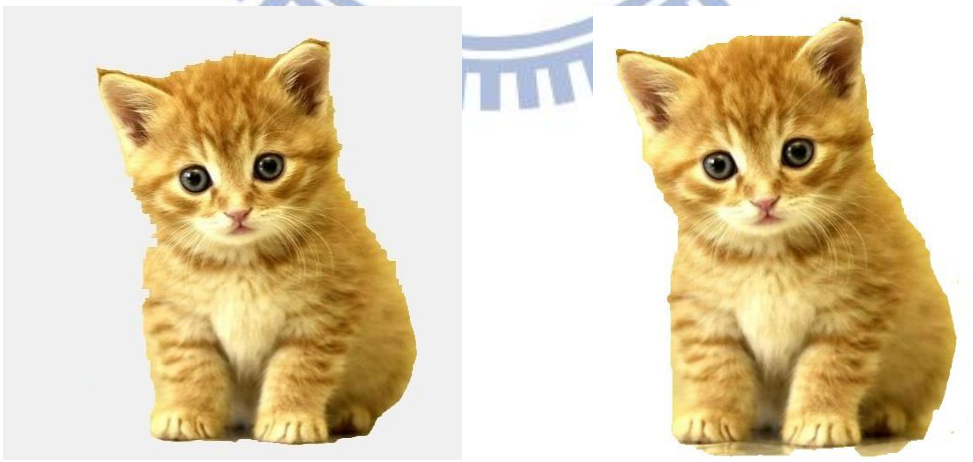
Table 4-6 Execution time of various image revolutions using Tiger

Sequence	256×150	512×300	768×450	1024×600
GrabCut	0.90 sec	1.52 sec	5.97 sec	10.93 sec
Our Method	1.29 sec	3.34 sec	6.77 sec	11.34 sec

C. Kitten

The foreground and background of this picture is very similar (see Figure 4-7(a)) so it's really hard to cut out the object brilliantly. The results from the two methods are shown in Figure 4-7(c) and Figure 4-7(b). Our cutout result is slightly better than GrabCut where the result of GrabCut left an area of shadow around the kitten. Also, our execution time is much faster than GrabCut (see Table 4-7).

(a) **(b)**

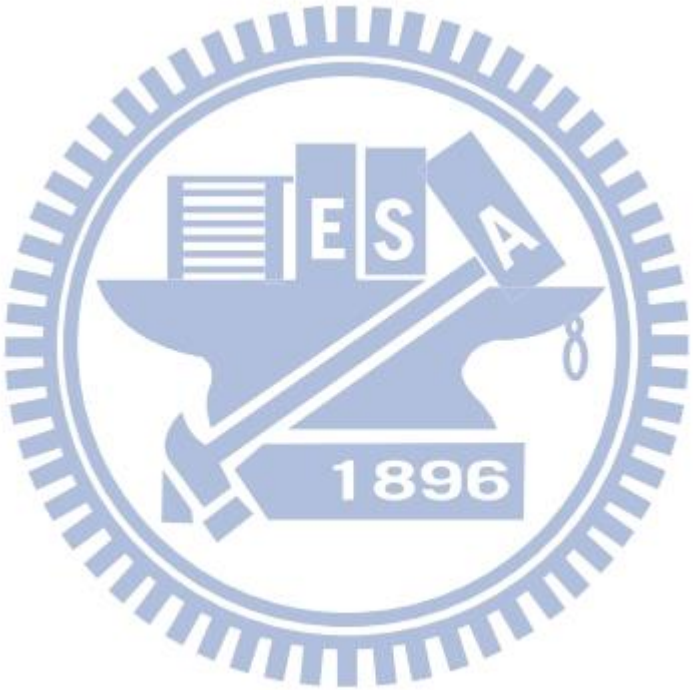


(c) **(d)**

Figure 4-7 Result comparisons using Kitten. (a) The original image. (b) User labeling (red rectangle box). (c) Our result. (d) Result of GrabCut.

Table 4-7 Execution time of various image revolutions using Kitten

Sequence	258×180	515×360	773×540	1030×720
GrabCut	2.05 sec	5.44 sec	11.53 sec	17.70 sec
Our Method	1.12 sec	2.97 sec	5.23 sec	9.42 sec



Chapter 5 Conclusion

In this thesis a fast interactive image cutout tool is developed. It is easy to learn and can produce better quality of result in less time than several reported image cutout tools. We successfully reduced the user-interaction level by only placing a rectangle box around the foreground object yet without increasing any execution time or lose the quality of the segmentation result. The combining usage of the Region Growing algorithm and Gaussian Mixture Model improved the speed of the segmentation procedure and the accuracy of the image color distribution probability.

Although we had outperformed GrabCut [4] in many aspects (see Chapter 4), there are still many problems that were not tackled in the proposed method. The method we proposed in this thesis only used the color distribution of the image as the main information to cutout the foreground object. As a result, it is less sensitive with objects that have severe luminance change and apparent boundaries compared with GrabCut [4]. However, adding a new feature to consider may cause the growth of the execution time. Therefore, finding the equilibrium point between the segmentation result and the execution time may be the top priority work in the future.

References

[1] N. R Pal and S. K Pal, "A Review on Image Segmentation Techniques," *Pattern Recognition*, Vol. 26, No. 9, pp. 1277–1294, 1993.

[2] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Transactions On Pattern Analysis And Machine Intelligence*, Vol. 22, No. 8, pp. 888–905, 2000.

[3] Adobe Photoshop User Guide, *Adobe Systems Incorporation*, 2002.

[4] L. J. Reese and W. A. Barrett, "Image Editing with Intelligent Paint," *Proceedings of Eurographics*, Vol. 21, No. 3, pp. 714–724, 2002.

[5] Y. Li, J. Sun, C. K. Tang and H. Y. Shum, "Lazy Snapping," *In Proceedings of ACM SIGGRAPH*, pp. 303–308, 2004.

[6] C. Rother, V. Kolmogorov and A. Blake, "GrabCut - Interactive Foreground Extraction Using Iterated Graph Cut," *In Proceedings of ACMSIGGRAPH*, pp. 309–314, 2004.

[7] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active Contour Models," *International Journal of Computer Vision*, Vol. 1, No. 4, pp. 321–331, 1987.

[8] E. Mortensen and W. Barrett, "Intelligent Scissors for Image Composition," *In Proceedings of ACM SIGGRAPH*, pp. 191–198, 1995.

[9] M. Gleicher, "Image Snapping," *In Proceedings of SIGGRAPH*, 95, pp. 183 190, 1995.

[10] P. Pres, A. Blake and M. Gangnet, "Jetstream: Probabilistic Contour Extraction with Particles," *In Proceedings of International Conference on Computer Vision*, Vol. 2, pp. 524–531, 2001.

[11] M. Ruzon and C. Tomasi, "Alpha Estimation in Natural Images," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp.

18–25, 2000.

[12] Y. Y. Chuang, B. Curless, D. H. Salesin and R. Szeliski, “A Bayesian approach to Digital Matting,” *In Proceedings of IEEE CVPR 2001*, Vol. 2, pp. 264–271, 2001.

[13] Y. Boykov and M. P. Jolly, “Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images,” *In Proceedings of IEEE International Conference on Computer Vision*, 2001.

[14] R. C. Gonzalez and R.E. Woods, *Digital Image Processing Second Edition*, Prentice Hall, New Jersey, 2002.