공학석사 학위논문

Divide-and-conquer Method with Refined Trimap for High-resolution Image Matting

리파인드 트라이맵이 적용된 분할 정복 방법을 기반으로 한 고해상도 이미지 매팅

2018 년 2 월

서울대학교 대학원

전기·정보 공학부

박 현 우

Divide-and-conquer Method with Refined Trimap for High-resolution Image Matting

리파인드 트라이맵이 적용된 분할 정복 방법을 기반으로 한 고해상도 이미지 매팅

지도 교수 고 형 석

이 논문을 공학석사 학위논문으로 제출함
2017 년 11 월

서울대학교 대학원
전기·정보 공학부
박 현 우

박현우의 공학석사 학위논문을 인준함
2018 년 2 월

위 원 장 _____ (인)

부위원장 _____ (인)

위 원 _____ (인)

Abstract

High-resolution image matting requires a substantial memory capacity. Therefore, it is very difficult to perform a high-resolution image matting, e.g. KNN image matting, in a common computing environment. This dissertation presents a simple yet novel methodology to solve a memory shortage issue involved in a high-resolution image matting with application of the divide-and-conquer method and a refined trimap. We use a refined trimap to minimize the potential quality degradation in the implementation of the divide-and-conquer method. A confidence map is computed by comparing the original input image and down-sampled image with lower resolution, and the confidence map is used to judge the reliability of the multigrid method in refining the trimap. The original input image is divided into fragments and individual alpha matte is computed with the refined trimap (fragment-level matting); each fragment includes a margin overlapping with nearby fragment(s). Then, we merge the fragments in consideration of the overlapping margin to make sure the merged image does not suffer from discontinuity among fragments. This is the refined D-C method dealing with high-resolution image matting regardless of matting method. We propose this fast and versatile method for high-resolution image matting. This method protects users from potential memory shortage while obtaining a reasonable-

quality alpha matte from a high-resolution image.

Keywords: Image matting, High-resolution image, Divide-and-conquer, Refined trimap, Confidence map

Student Number: 2016-20911

Table of Contents

List of Tables

List of Figures

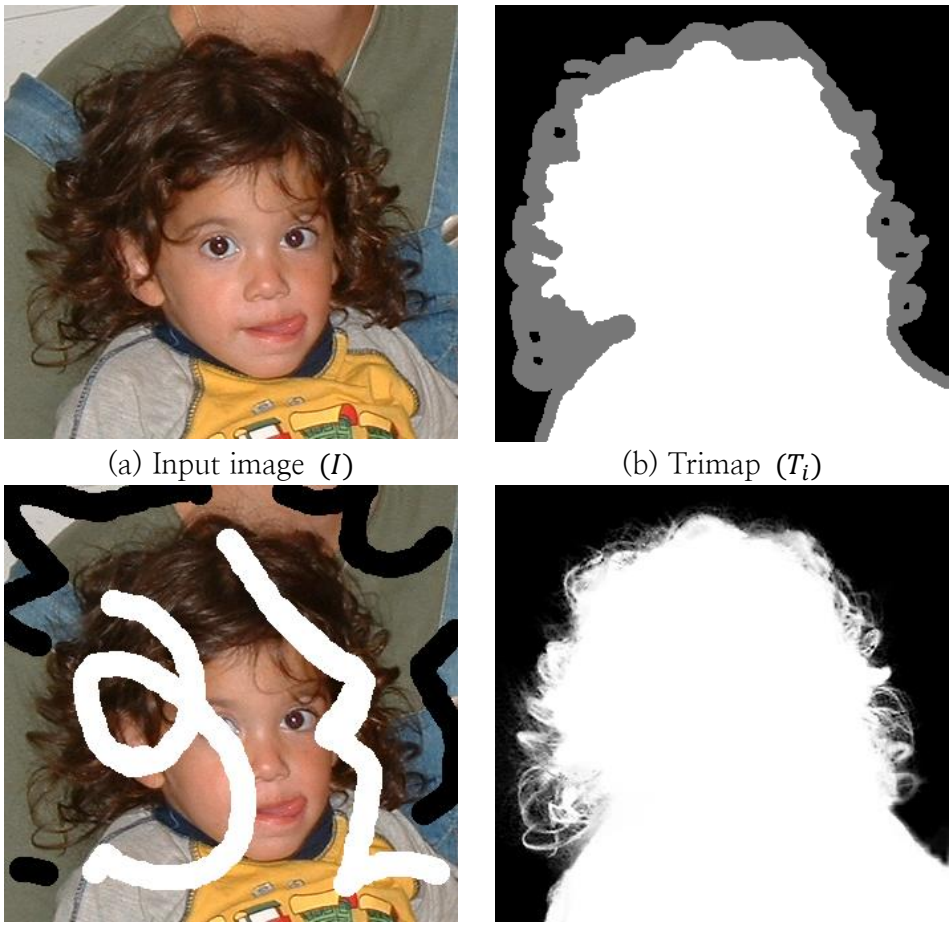
Chapter 1. Introduction

Image matting is an important technique in image processing. Image matting is a process of calculating a foreground opacity, known as the alpha matte(α), to extract the foreground of a given image. Image matting is a primary interest of both academia and industries. Since an image is a composite of foreground and background, it can be expressed in a functional form:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i \tag{1}$$

where I is the input image, F, B, and α representing foreground, background, and foreground opacity respectively. The alpha matte is what we want to know for image matting. Hence, the alpha matte (α) can be computed from the image function (1), where $\alpha =$ [0, 1]. If α_i= 1, i represents a foreground pixel, and if α_i= 0, i is a specific background pixel. Image matting is hardly an under-constrained problem because there are more unknowns than knowns in the problem. Therefore, we need some additional information to solve the matting problem posed in the equation. For example, we can obtain the additional information from manually created trimap or scribbles.

Figure 1.1 shows components and a result of image matting. Fig 1.1 (a) is an original image input for image matting, (b) is a manually created trimap, (c) is a manually created scribbles on the input image, and (d) is the desired alpha matte we are looking for.



(a) Input image (I) (b) Trimap (T_i)

(c) Scribbles on the input image (d) Alpha matte (α)

Figure 1.1: Input Image, Trimap, Scribbles on the Image, and Alpha Matte from [12]

We can use the trimap—we also obtain a rough trimap from scribbles—to compute the alpha matte, because the trimap provides additional information:

$$T_i = \begin{cases} 1.0; \text{ if } i\text{-th pixel is foreground} \\ 0.0; \text{ if } i\text{-th pixel is background} \\ 0.5; \text{ otherwise} \end{cases} \tag{2}$$

That is, in a trimap or scribbles, we can obtain some additional information because a pixel painted white in a trimap or scribbles indicates that it is a foreground pixel; a pixel painted in black indicates that it is a background pixel. The gray part of the trimap ($T_i = 0.5$)—this is equivalent to remainder of definite scribbles—is an unknown part that we need to figure out whether each pixel is a foreground pixel or a background pixel. Even though a trimap gives more information about whether a particular pixel is of foreground or background than scribbles, trimap is not necessarily superior to scribbles because more user-inputted manual efforts are required to create a trimap.

1.1. Motivation

Existing image matting methods have gained great achievements on alpha matting evaluation [15]. But those methods are mainly concerned with low-resolution images, because more memory has to be allocated is needed for high-resolution images. Greater the memory is allocated in solving a linear system, e.g. $Ax = b$, for higher-resolution image matting, because A is a large sparse matrix proportional to a squared number of image pixels.

With recent advances in technology, higher-resolution images have become more popular; now more people have an easy access to SLR cameras and the built-in camera in smart phones create high-resolution images. The general size of image resolution grows day by day; thus, it is getting more difficult to exploit existing matting methods on images. For example, KNN matting [13] requires 8.91 GB of memory allocation when performing matting on an 8.1 Megapixel image which size is 3133 x 2600. This is a great demand of memory allocation considering most home desktops are sold with 8GB of memory, even nowadays.

To account for potential memory shortage, we implement the divide-and-conquer method in image matting. Unfortunately, a simple implementation of the divide-and-conquer method in image matting does not yield a favorable result; the thereby-obtained alpha matte may well contain a problematic discontinuity along the border(s) of conquered fragments as in Figure 1.2. In this paper, we introduce a refined trimap to

yield a proper alpha matte in the divide-and-conquer image matting, so that the high-resolution image matting is more easily obtainable in ordinary work stations.

(a) Input image (GT21) (b) Scribbles (c) The alpha matte with the divide-and-conquer on GT21

Figure 1.2: Input Image, Scribbles and Divide-and-conquer Result

The refined trimap is based on the manually inputted trimap and scribbles, and it substitutes the role of trimap or scribbles in divide-and-conquer image matting process. This way, the discontinuity in divide-and-conquer generated alpha matte is substantially eliminated. The refined trimap is automatically generated with a two-step algorithm: the first step is to create a newly defined confidence map by comparing input image and up-sampled low-resolution image in the Hue, Saturation, and Value (HSV) space. Then we generate a refined trimap using the confidence map and the matting result of the down-sampled image.

In the divide-and-conquer process, the input image and the refined trimap are divided into fragments for matting. In this paper, we divide the original image into four fragments with a margin overlapping with nearby fragment(s). Fragment-level alpha mattes are computed individually, and the four fragment-level alpha mattes are merged. Since the process is automated, the only required user-level effort is creating a trimap or scribbles enabling the problem solving possible, just like other extant matting methods. The alpha matte from this refined divide-and-conquer method (hereby the refined D-C method) is free from discontinuity along the border of fragments, and this method can expand all existing matting methods using trimap or scribbles to solve for the alpha matte. The applicability of scribbles depends on whether the original matting method—or the base matting method—accepts scribbles or not.

In this paper, KNN matting [13] is used as a base matting method to elaborate the refined D-C method for high-resolution image matting, because KNN matting is simple and easy to implement. Therefore, it is a good base matting method to focus on strengths of the refined D-C method. Having KNN as the base matting method, we use KNN matting to find the alpha matte in both low-level matting and application of the divide-and-conquer method (or fragment-level matting).

Therefore, we argue that the major contribution of this paper is proposing an easily adaptable application of the divide-and-conquer that

enables high-resolution image matting for all existing image-matting methods using trimap or scribbles. This is a fast and versatile method dealing with high-resolution image for all existing matting methods. That is:

1. This paper introduces a method that circumvents the inevitable memory allocation problem associated with high-resolution image matting.

2. This method is applicable to all existing matting methods that solves the linear system of Ax=b, such as closed-form solution [12] or KNN matting [13].

3. This method maintains the quality of estimated alpha matte, compare to the alpha matte estimated with the base matting method.

4. This method requires similar or less time than the base matting method.

The rest of this paper is structured as follows: In Chapter 2, we review relevant literature on the extant image matting methods, the multigrid method, and the divide-and-conquer method. Chapter 3 will discuss how the newly proposed refined trimap is automatically generated. In Chapter 4, we explain how the refined trimap is used in the divide-and-conquer method in reducing the discontinuity created along the border of conquered fragments. In chapter 5, we summarize the experimental results. Lastly, we conclude the paper with discussion for further application in chapter 6.

Chapter 2. Literature Review

2.1. Extant Literature on Image Matting

Image matting is not a well-posed problem. That is, it is an ill-posed problem with 7 unknowns with 3 equations.

$$\begin{aligned} \mathrm{I}_{\mathrm{i}}^{\mathrm{R}} &= \alpha_i F_i^R + (1 - \alpha_i) B_{\mathrm{i}}^R \\ \mathrm{I}_{\mathrm{i}}^{\mathrm{G}} &= \alpha_i F_i^G + (1 - \alpha_i) B_{\mathrm{i}}^G \\ \mathrm{I}_{\mathrm{i}}^{B} &= \alpha_i F_i^B + (1 - \alpha_i) B_{\mathrm{i}}^B \end{aligned} \tag{3}$$

As it is in (3), the matting equation can be divided into three channels: R, G, and B. This is true for a color image with three channels. That leaves us with seven unknown variables of α_i, F_i^R, F_i^G, F_i^B, B_{i}^R, B_{i}^G, and B_{i}^B. There are many approaches to solve an under-constrained problem. Blue screen matting [18] is the most widely used method in movie and media. This matting requires that there is no blue color in foreground, given the background is blue. This allows an assumption to reduce seven to three unknowns, so the set of three-channel matting equations can yield a single unique solution. However, the blue screen matting does not work well when the background color also exists in the foreground. Therefore, it has an inherent limitation that requires uniquely controlled setting for a successful matting; this fails to yield an appropriate alpha matte from natural images.

In a natural-image matting, a trimap is used to circumvent the under-constrained problem. A trimap is also composed of three parts; a trimap has foreground, background and unknown regions. A trimap-based matting finds α_i in the unknown region. There are two types of matting methods using the trimap: sampling-based matting, and affinity-based matting.

The sampling-based matting method is based on sampling. This method basically collects some samples from both foreground and background regions, where the alphas of pixels are already verified by the provided trimap, to gather information to approximate the alpha of pixels in the unknown region. The alpha of each pixel in the unknown region is approximated with the best representable pairs of a foreground pixel and a background pixel, where the proximity between pixels in unknown regions is ignored. Many sampling algorithms to find the best foreground and background pair are introduced by [5,6,8,11,17,20]

The affinity-based matting method uses affinity matrix constructed by the color or Euclidean distance similarity among all the pixels in the image. To obtain an alpha matte, the affinity-based matting method propagates alpha values in the known regions to unknown regions. A closed-form solution [12] uses color-line model in the local window. The matting Laplacian matrix is thereby computed upon the information in the color-line model, and the Laplacian matrix is the affinity matrix.

KNN matting [13] is a kind of affinity-based matting, and this is used as the base matting method to elaborate the refined D-C method in this paper. KNN matting well presents why matting of high-resolution image is difficult in each matting process. In KNN matting, each pixel finds a K-nearest neighbor in the feature vector to make an affinity matrix A. KNN matting defines feature vector as

$$X(i) = (cos(h), sin(h), s, v, x, y)_i \tag{4}$$

where h, s, and v are the HSV color coordinates of an i-th pixel and x, y are the coordinates of i-th pixel on image. The affinity matrix A is a large sparse matrix of N x N, where N is the number of all pixels in the input image. The matting Laplacian L is created by $D - A$, where the matrix D is a diagonal N x N matrix. To find the alpha matte, KNN matting [13] propagates alpha by iteratively solving the equation,

$$(L + \lambda D_s)\alpha = \lambda b_s \tag{5}$$

where L is the matting Laplacian from affinity matrix of image, λ is a big constant, b_s is a binary vector containing the alpha values provided by the trimap or scribbles, and D_s is the diagonal matrix of which diagonal element taking the value 1 for constrained pixels. This step brings about potential memory shortage problem because the size of the matting Laplacian L is gigantic when a high-resolution image matting is processed. The size of matting Laplacian is a squared number of image pixels, so the Laplacian size dramatically increases with the image resolution inevitably. For example, a

high-resolution image of 3133 x 2600 pixels makes the matting Laplacian matrix L of 8145800 x 8145800. This is how a high-resolution image matting triggers the Laplacian matrix cause a memory shortage in solving the system equations. Also, the ease of implementation makes KNN a good based matting method to present the refined D-C method introduced in this paper; KNN matting has relatively fewer parameters yet it is strong matting method applicable with both trimap and scribbles.

2.2. Multigrid Method

A multigrid method is a widely adopted method in many fields such as [2,3,7,19,21] that it is accelerating the convergence of the original iterative method in a fine grid by solving a problem in a coarser grid. Figure 2.1 presents a common multigrid V-cycle.

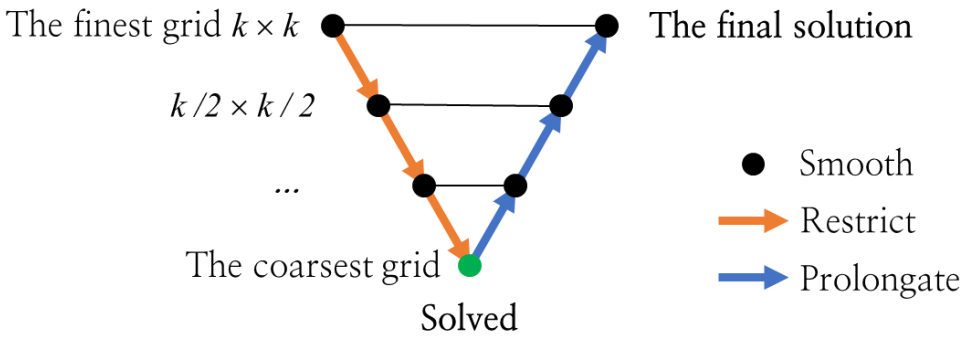


Figure 2.1: V-cycle of Multigrid Method

At each smooth stage, high frequency errors are reduced. Restricting a grid is down-sampling of the grid. This step is repeated until the defined coarsest grid is reached. At the coarsest grid, a problem is solved, and the solution is prolongated to the finest grid. The multigrid method has strong advantages: The multigrid method can solve a linear system with n unknowns in $O(n)$ time; the multigrid method requires smaller memory allocation compared to other methods to solve a linear system. Therefore, a high-res matting with the multigrid method has attempted [14], but the result is not strong or convincing. In this paper, the multigrid method is partly adopted in the automatic generation of a refined trimap.

2.3. Divide-and-conquer Method

The divide-and-conquer method solves a big problem by dividing parts. The method divides the original problem into two, four, or more fragment parts and independently solves partial problem in each part. After that, the method combines the results of each fragment parts to produce a solution to the original problem. This method surely has advantages. First, divide-and-conquer method makes a big and difficult problem easier to solve, or sometimes makes an otherwise unfeasible problem solvable. The second advantage is in its algorithm efficiency①. Also, it enables effective memory

① Quicksort—the most popular algorithm in the sorting problem—has an average running time of $O(n\log n)$ using the divide-and-conquer method [9].

usage by dividing a big problem into smaller fragments, so the concurrent memory allocation requirement is substantially reduced to solve each smaller problem. Accordingly, an image matting using the divide and conquer method [4] is recently introduced. This method yields a good evaluation score from [15] with very little memory allocation, but this too has a limitation. This method divides an input image into many smaller images of 100 x 100 pixels, hence the process is extremely time consuming.

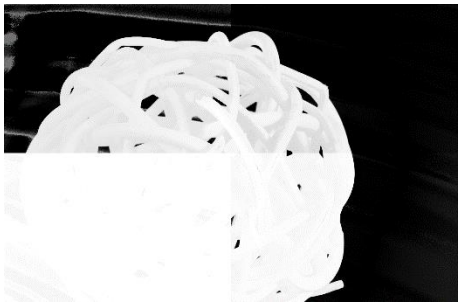
To sum up, existing image matting methods are broadly categorized into the sampling-based matting method and the affinity-based matting method. Both require solving an extremely memory-consuming problem in matting of high resolution image, which brings about concern for potential memory shortage. The proposed method in this paper suggests a fast and practical application of the divide-and-conquer method in image matting, circumventing potential memory shortage issue involved with high-resolution image matting in the existing image matting methods. We introduce the refined divided-and-conquer method (the refined D-C method). This is applicable to any existing matting method that uses trimap or scribbles.

Chapter 3. Automatic Generation of Refined Trimap

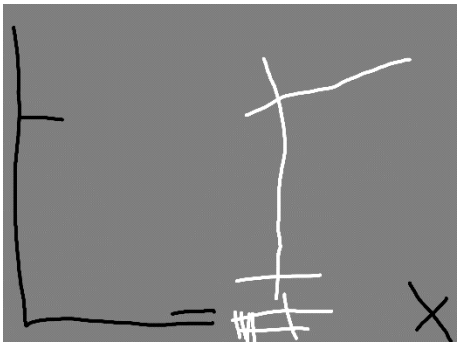
Directly applying a sparse user input to divide-and-conquer method results in discontinuity along the border of the divided fragments when fragments are merged back (see Figure 3.1 (b), (d) and (f)). This is because each trimap or scribbles of respective image fragment is not propagated to other separated image fragments. Since the trimap or scribbles is the key information in estimating the alpha of each pixel, the information from the trimap or scribbles must be universally spread throughout all image fragments without bias. Hence the trimap refinement is a crucial process of creating a refined trimap that propagates trimap information of an image fragment to all other remaining fragment(s) in the divide-and-conquer for image matting.

An automatic generation of refined trimap borrows the concept from the multigrid method, where a refined trimap is automatically generated through process in Figure 3.2. First, the input image and the user input trimap are reduced by the half of their width and length. We obtain a low-level alpha matte through KNN matting [13] using a down sampled low-level image and the trimap. Using the HSV difference method with the original input image and the restricted low-resolution image, we can define a new confidence map. Then, a refined trimap is generated in conjunction with the confidence map and the low-level matting result.

(a) Scribbles of GT02

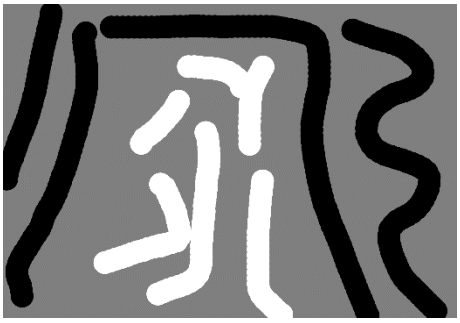


(b) The alpha matte with the divide-and-conquer on GT02



(c) Scribbles of GT13

(d) The alpha matte with the divide-and-conquer on GT13



(e) Scribbles of GT06

(f) The alpha matte with the divide-and-conquer on GT06

Figure 3.1: Scribbles and Result of Divide-and-conquer with Scribbles

3.1. Generating the Confidence map

The input image I_h and the Trimap T_h are down-sampled to I_l and T_l as half of the width and the height, like the restrict process in the multigrid method. In this paper, the bilinear down-sampling method is used to bring the average values of the four surrounding pixels to the pixel i. Then, the restricted image I_l is up-sampled back to its original-image resolution $I_{\uparrow_2 l}$. We can see the difference between I_h and $I_{\uparrow_2 l}$ in Figure 3.3. the up-sampled $I_{\uparrow_2 l}$ is blurrier than the original input image I_h, because the image has lost some information in the restrict process.

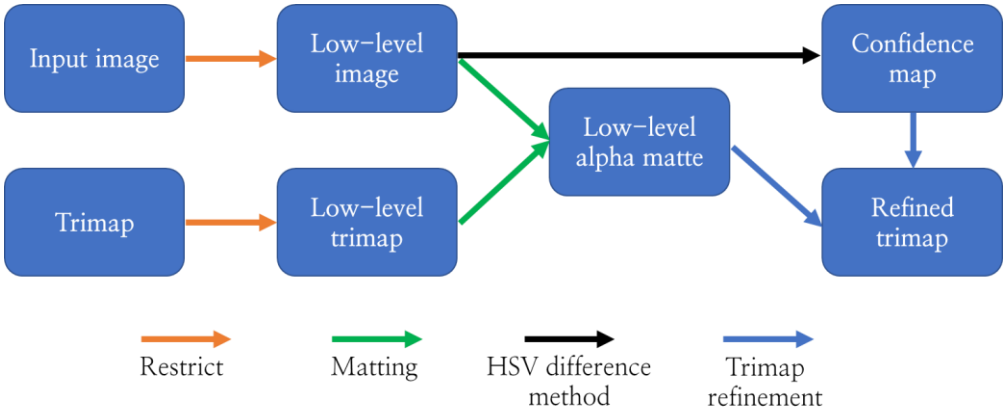


Figure 3.2: Flow Chart depicting the Process of Generating a Refined Trimap

We define a confidence map in this process. If the difference between $I_{(h,i)}$ and $I_{(\uparrow_2 l,i)}$ is bigger than a threshold, i.e. the pixel i loses more information in the restrict process, then we set a confidence value for the respective pixel zero; the confidence value is 1 for pixels otherwise. Equation

(4) from the KNN matting [13] is used to construct the confidence map. Feature vector $X(i)$ contains information about the color of pixel i and the distance among pixels, so it is suitable for comparing the difference between the high-resolution image I_h and the restricted image $I_{\uparrow_2 l}$.

(a) Original GT21 (b) Up-sampled GT21-restricted

Figure 3.3 Panel A: Comparing Original Input Image I_h with $I_{\uparrow_2 l}$.

(c) Original GT23 (d) Up-sampled GT23-restricted

Figure 3.3 Panel B: Comparing Original Input Image I_h with $I_{\uparrow_2 l}$.

Using the feature vector, we obtain a Euclidean distance. D_i is a Euclidean distance of feature vector between the input image and the up-sampled result of restricted input image:

$$D_i = \left\| X_h(i) - X_{\uparrow_2 l}(i) \right\| \tag{6}$$

The confidence C_i is defined based on D_i:

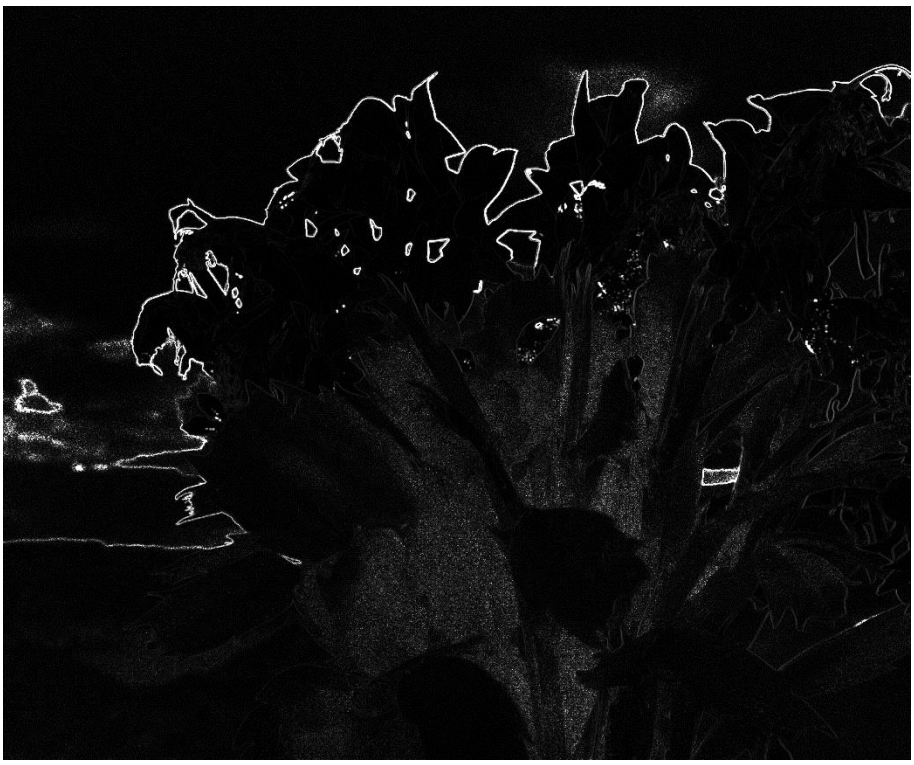
$$C_i = \begin{cases} 1; & \text{if } D_i < \text{threshold}, T_{(h,i)} > 0.99, \text{ or } T_{(h,i)} < 0.01 \\ 0; & \text{otherwise} \end{cases} \tag{7}$$

The trimap T_h gives us the information about the foreground and the background. If $T_{(h,i)} > 0.99$ or $T_{(h,i)} < 0.01$, this means i-th pixel in trimap T_h is either the foreground or background pixel. Hence, we do not have to recalculate the alpha for such pixel, so set $C_i = 1$. Pixels with $C_i = 0$ are the unknown part in the refined trimap, and this will be discussed later in the sub-chapter 3.2; the unknown part of the refined trimap needs to be recalculated in the divide-and-conquer method. We set the threshold to 0.1 to recalculate the top 10% of pixels where D_i values are high. Figure 3.4 shows the results of confidence map created on GT23 and GT21. In Figure 3.4 (a) and (c), white pixels are those with big D_i values. The boundary of objects or hairy parts presents big D_i values indicating that they lose information during the restrict process. Using D_i, the binary map C_i is generated. The white pixels in C_i are reliable pixels while black pixels are not. The refined trimap is computed upon the confidence map C_i, where the information of C_i is transmitted to the divided-and-conquer method.

(a) D_i of GT23

(b) C_i of GT23

Figure 3.4 Panel A: D_i and C_i of Input Image GT23



(c) D_i of GT21

(d) C_i of GT21

Figure 3.4 Panel B: D_i and C_i of Input Image GT21

3.2. Generating the Refined Trimap with Confidence map

To generate a refined trimap, the confidence map C_i and the low-level image matting result are considered. We obtain the low-level matting result α_l with the low-level input image I_l and the low-level trimap T_l (see Figure 3.2 matting arrow).

Since the total number of pixels in the low-level input image I_l is only a one fourth to that of the original input image I_h, the size of the matting Laplacian L_l is only 6.25% to that size of the original matting Laplacian L_h when computing α_l. Therefore, the low-level matting can be solved with much less memory allocation. Then, α_l is upsampled to $\alpha_{\uparrow_2 l}$, where the resolution is now identical to that of the original input image. Then the refined trimap $\hat{T}_i$ is defined as a function of C_i and the up-sampled low-level matting result $\alpha_{\uparrow_2 l}$:

$$\hat{T}_i = \begin{cases} \alpha_{(\uparrow_2 l,i)}; & \text{if } C_i = 1 \\ 0.5; & \text{otherwise} \end{cases} \tag{8}$$

If i-th pixel of low-level matting result $\alpha_{(\uparrow_2 l,i)}$ is a foreground pixel and if C_i = 1, i-th pixel of the refined trimap $\hat{T}_i$ is a foreground pixel. On the other hand, if $\alpha_{(\uparrow_2 l,i)}$ belongs to the background and $C_i = 1$, $\hat{T}_i$ is a background pixel. For any other scenarios, $\hat{T}_i$ set to 0.5; this represents a pixel in the unknown region of the trimap and i-th pixel's alpha value will be recalculated in the divide-and-conquer method in later process. The

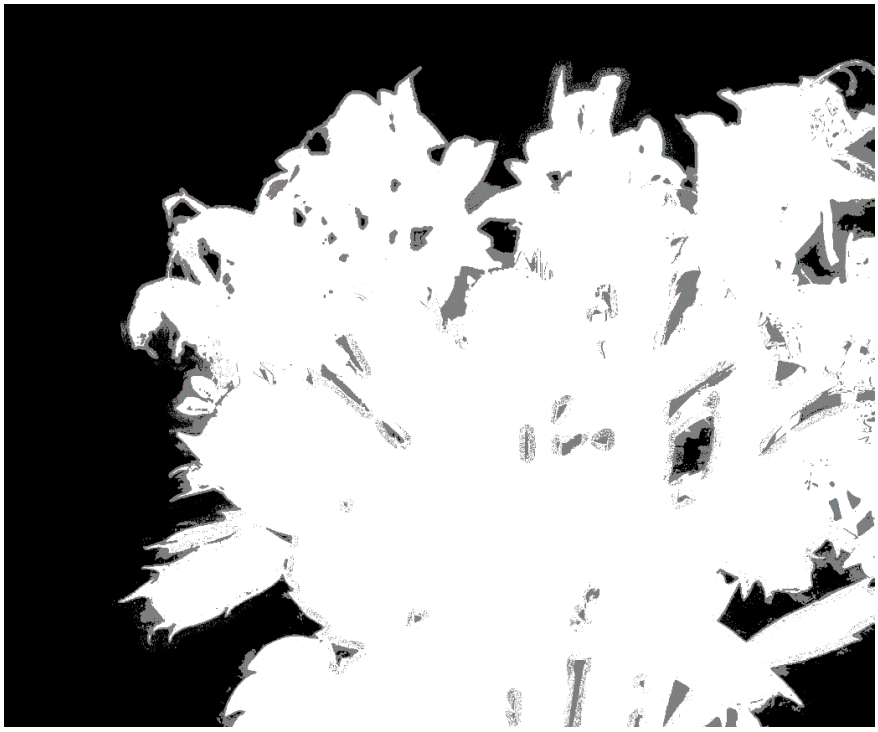
computed refined trimap of sample images are shown in Figure 3.5 and Figure 3.6. The refined D-C method can be applied to both trimap and scribbles.

(a) Trimap of GT23

(b) Refined trimap of GT23

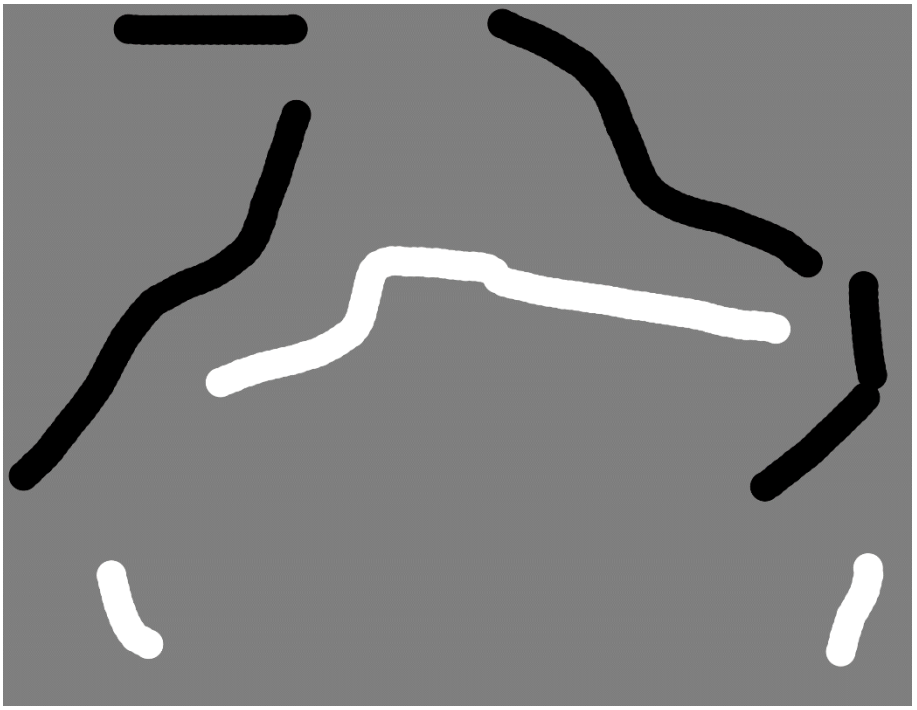
Figure 3.5 Panel A: Comparing the Trimap and Refined Trimap of GT23

(c) Trimap of GT21



(d) Refined trimap of GT21

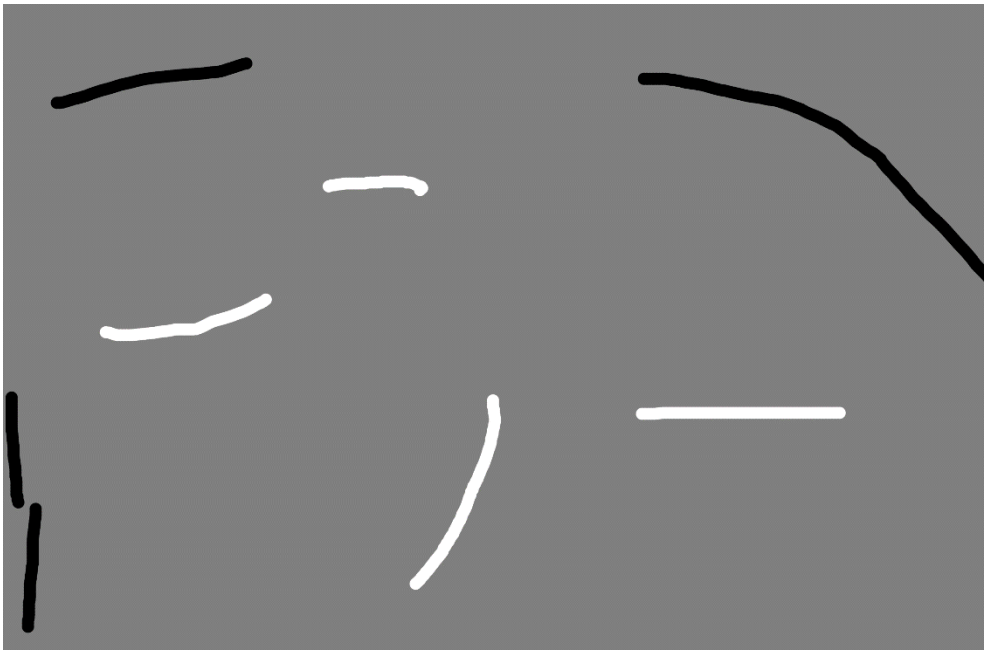
Figure 3.5 Panel B: Comparing the Trimap and Refined Trimap of GT21



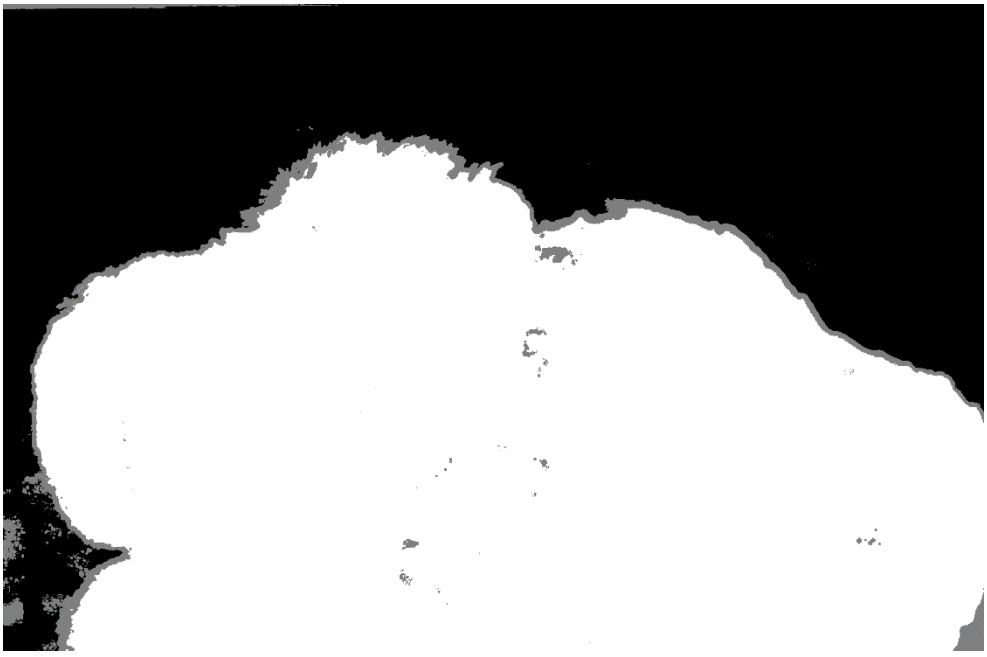
(a) Scribbles of GT07

(b) Refined trimap of GT07

Figure 3.6 Panel A: Comparing the Scribbles and Refined Trimap of GT07



(c) Scribbles of GT12



(d) Refined trimap of GT12

Figure 3.6 Panel B: Comparing the Scribbles and Refined Trimap of GT12

The refined trimaps in Figure 3.5 (b), (d), Figure 3.6 (b) and (d), look like alpha mattes, because a refined trimap is based on the computed low-level alpha matte. The information in scribbles propagates to the entire image in computation of low-level alpha matte. The pixels with low confidence in the confidence map become unknown parts in the refined trimap and they are recalculated in the later divide-and-conquer stage. Now, we obtain a refined trimap with superior information than a trivial trimap about the foreground and background of the input image. Therefore, the potential discontinuity problem of the divide-and-conquer image matting application can be resolved with the refined trimap.

Algorithm 1 Generating Refined Trimap

Input: Input image I_h, Trimap T_h
Output: Refined Trimap $\hat{T}$

T_l = DownSample(T_h) ▷ Downsample using bilinear donwsampling
I_l = DownSample(I_h)
$I_{\uparrow_2 l}$ = UpSample(I_l)
$X_h(i)$ is a feature vector of i-th pixel in I_h
$X_{\uparrow_2 l}(i)$ is a feature vector of i-th pixel in $I_{\uparrow_2 l}$

for all pixel $i \in I_h$, $I_{\uparrow_2 l}$ **do** ▷ Calculate D_i
  $D_i = \|\mathrm{X}_h(i)$ - $X_{\uparrow_2 l}(i)\ \|$

for all $i \in D_i$ **do** ▷ Make Confidence map C_i
  if $D_i < threshold$ or $T_{(h,i)} > 0.99$ or $T_{(h,i)} < 0.01$ **then**
    $C_i = 1$
  else
    $C_i = 0$

$\alpha_{\uparrow 2l}$ = Upsample(Matting(I_h, T_h)) ▷ Low-level Image Matting

for all $i \in C_i$ **do** ▷ Generate Refined Trimap
  if $C_i = 1$ **then**
    $\hat{T}_i = \alpha_{(\uparrow 2l,i)}$
  else
    $\hat{T}_i$ =0.5

return $\hat{T}$

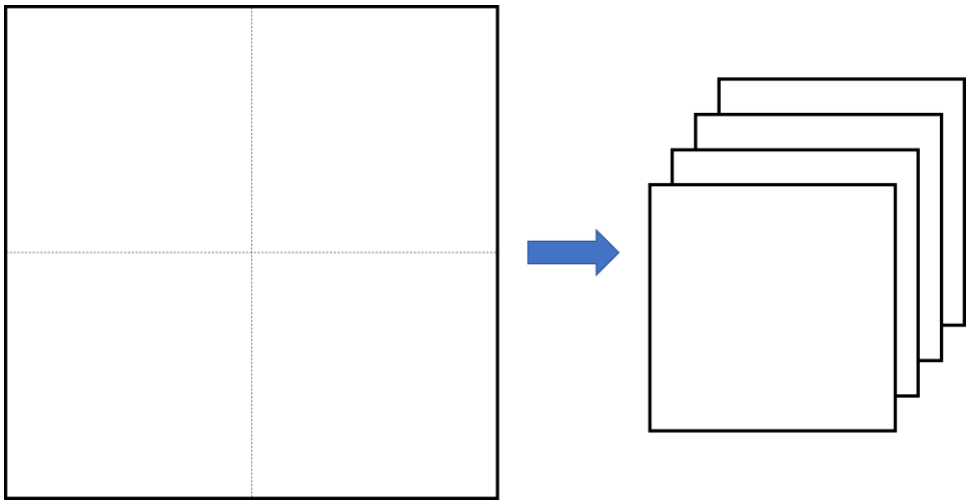
Chapter 4. Divide-and-conquer Method with Refined Trimap

As discussed earlier, the problem associated with a high-resolution image matting rises in calculation of the alpha matte with matting Laplacian because the size of Laplacian matrix is a squared number of image pixels. The divide-and-conquer method allows circumventing the problem, as the matting Laplacian of a smaller image fragment does not trigger a memory shortage problem.

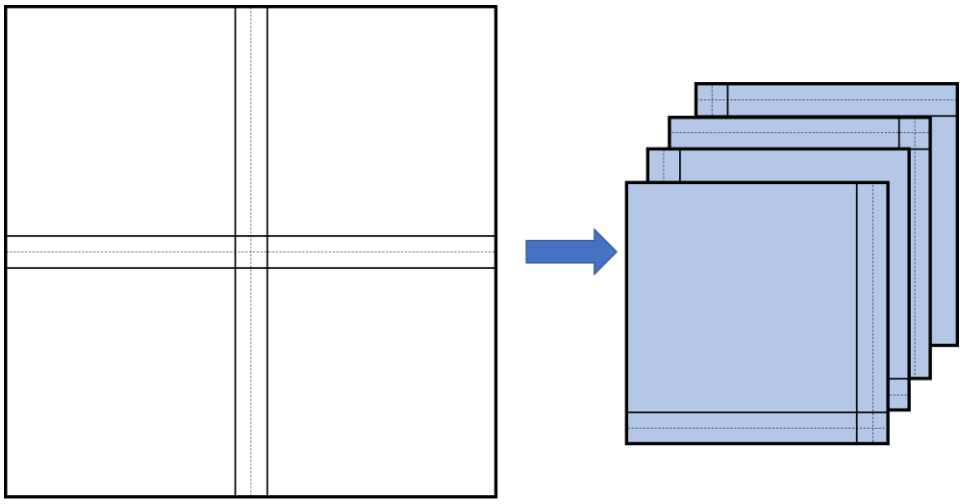
4.1. Applying the Refined Trimap

To apply the divide-and-conquer in high resolution image matting, we divide both the input image and the refined trimap into four fragments with overlapping margins. As noted earlier, the number of fragment is user discretionary, but we choose to divide the input image into four fragments in this paper. Pairs of each image fragment and the respective refined trimap fragment are used in the fragment matting process. The overlapping margin along the borders of fragments play a crucial role in eliminating a discontinuity created in the merging process after the fragment-level matting. Because the margin size determines the number of pixels repeatedly calculated in each fragment-level matting, the larger the overlapping margin yields a better outcome from the divide-and-conquer method. However, larger the margin size means bigger the fragment size, where the efficiency of the divide-

and-conquer method depends on how small each image fragment size is. Therefore, it is important to determine an adequate size of the overlapping margin to yield a balance between the efficiency and the quality of the divide-and-conquer process. In this paper, we set the overlapped margin to 10 pixels. As a result, pixels in the overlapping margin are computed at least twice during the fragment-level matting.



(a) Divide image into 4 parts without overlapped region



(b) Divide image into 4 parts with overlapped region

Figure 4.1: Divide Image with Overlapped Region

To deal with repeated computation of pixels in the margin—where computation result may vary depending on in which image fragment the overlapping pixel is being computed, we use the matting result of the neighboring image as a constraint to an overlapping pixel, when it is the second time the pixel is computed in the fragment-level matting process.

We know that KNN matting [13] propagates alpha by iteratively solving the equation, and we can recall that:

$$(L + \lambda D_s)\alpha = \lambda b_s \tag{5}$$

To update a constraint on a pixel in the overlapping margin, we set the element of D_s to 1 and assign an earlier matting result to b_s when the pixel is computed during the fragment-level matting process.

With the divide-and-conquer method, we can not only reduce memory allocation but also shorten the total running process time, as [1,10,16] has shown, because the time complexity of PCG used in [13] is $O(N^{1.25})$. As the input image is divided into smaller fragments, the expected computational gain is greater.

Chapter 5. Results

We use the high-resolution image and trimap provided in [15], and the scribbles used in KNN matting [13]. The results are summarized in table 5.1 and 5.2 respectively:

Image	GT12		GT13		GT21	
Resolution	3063 × 2018 6.18 Megapixel		3283 × 2443 8.02 Megapixel		3133 × 2600 8.14 Megapixel	
Method	KNN [13]	Refined D-C method	KNN [13]	Refined D-C method	KNN [13]	Refined D-C method
MSE	154.33	101.51	175.17	131.46	222.59	268.01
Memory (GB)	6.83	3.27	8.99	4.54	8.91	4.29
Running time (s)	150.52	125.52	216.86	157.97	202.51	168.60

Table 5.1: Results of using User Input with Trimap

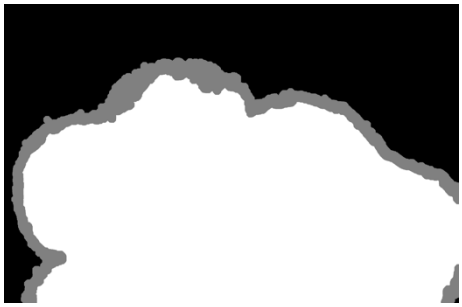
Image	GT12		GT13		GT21	
Resolution	3063 × 2018 6.18 Megapixel		3283 × 2443 8.02 Megapixel		3133 × 2600 8.14 Megapixel	
Method	KNN [13]	Refined D-C method	KNN [13]	Refined D-C method	KNN [13]	Refined D-C method
MSE	133.59	107.07	427.90	551.91	642.44	747.49
Memory (GB)	6.845	3.27	8.93	4.53	8.84	4.64
Running time (s)	205.66	123.17	300.77	165.85	272.19	165.254

Table 5.2: Results of using User Input with Scribbles

Table 5.1 shows the results of KNN matting [13] and the refined D-C method when using a high-resolution image and a trimap from [15]. Table 5.2 summarizes the image matting result with scribbles. The results are implemented in a 3.40GHz Intel® Core ™ i5-2600 processor, with DDR3 1333MHz 16GB memory, and Matlab® R2016b environment. When high-resolution image matting is performed using the KNN matting method [13], it consumes 8.99GB of memory for the GT13 that is an 8.02 Megapixels image. It is difficult to be handled in ordinary computing environment. The proposed method consumes 4.54GB of memory, and this is only half the usage of KNN matting method [13]. Also, the computation speed is 1.37 times faster than the original KNN matting. Similar results are recorded when scribbles are used. This is summarized in table 5.2. The memory cost is measured using the Matlab Profiler, and the peak memory usage is reported. The quality difference between the original KNN method and the refined D-C method is very trivial. Therefore, we are confident that the proposed method with confidence mapping is both memory- and speed-wise more advantageous.

(a) Input image (GT12)

(b) Ground truth



(c) Trimap

(d) Refined Trimap from (a)and (c)

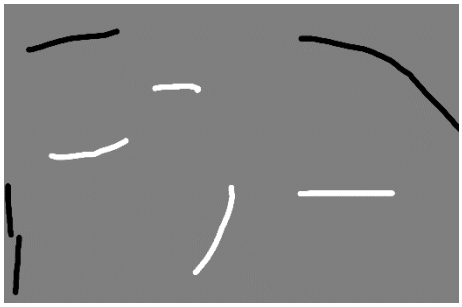
(e) KNN matting [13] with (c)

(f) Refined D-C method with (d)

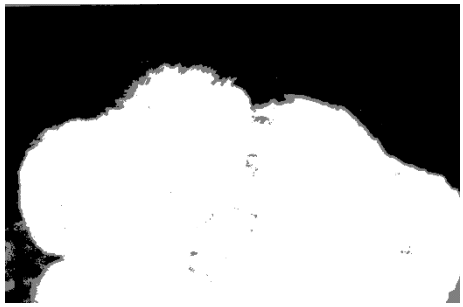
Figure 5.1 Panel A: Results of GT12 with Trimap

(a) Input image (GT12)

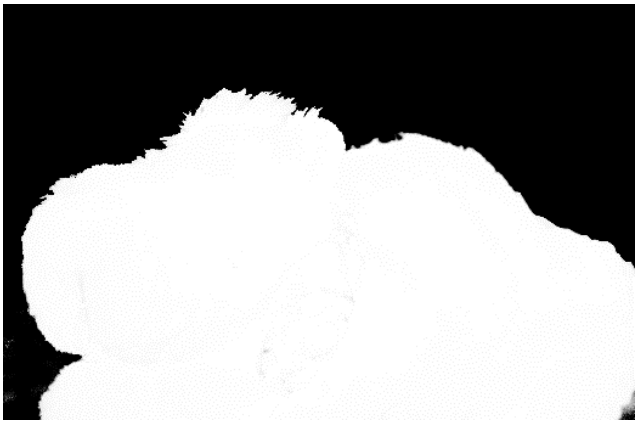
(b) Ground truth



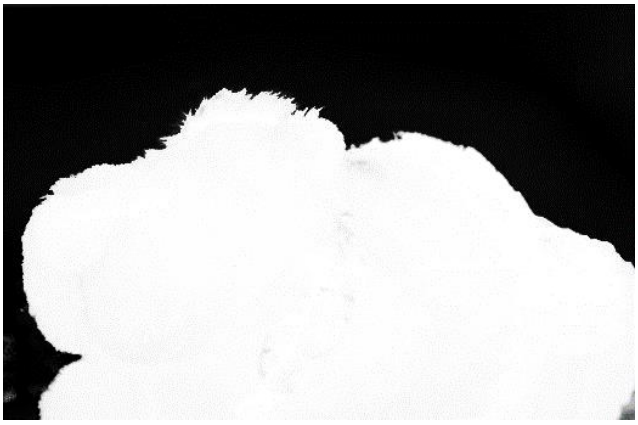
(g) Scribbles



(h) Refined Trimap from (a) and (g)



(i) KNN matting [13] with (g)



(j) Refined D-C method with (h)

Figure 5.1 Panel B: Result of GT12 with Manual Scribbles

(a) Input image (GT13)

(b) Ground truth

(c) Trimap

(d) Refined Trimap from (a) and (c)

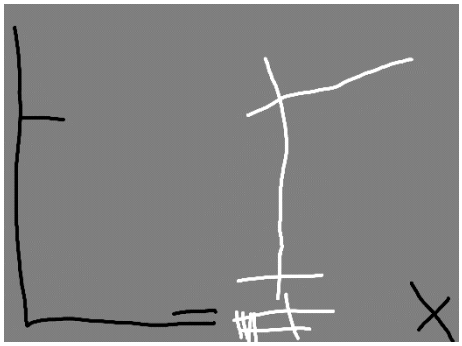
(e) KNN matting [13] with (c)

(f) Refined D-C method with (d)

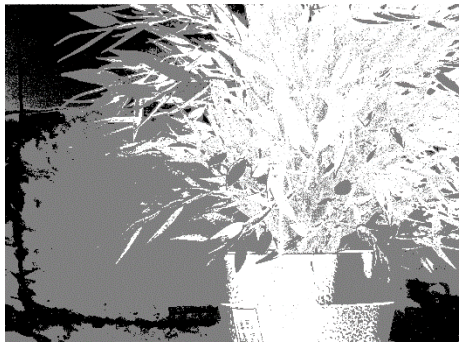
Figure 5.2 Panel A: Results of GT13 with Trimap

(a) Input image (GT13)

(b) Ground truth



(g) Scribbles



(h) Refined Trimap from (a) and (g)

(i) KNN matting [13] with (g)

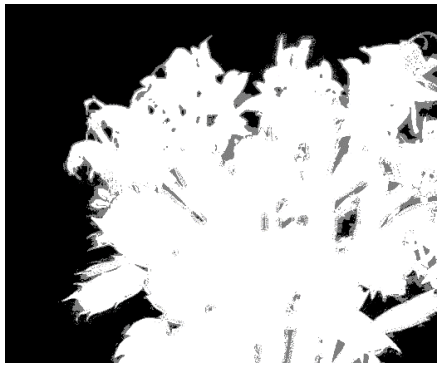
(j) Refined D-C method with (h)

Figure 5.2 Panel B: Result of GT13 with Manual Scribbles

(a) Input image (GT21)

(b) Ground truth

(c) Trimap



(d) Refined Trimap from (a) and (c)

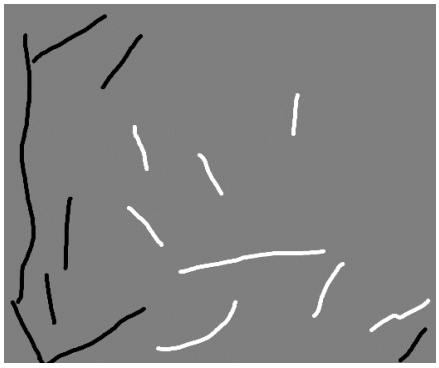
(e) KNN matting [13] with (c)

(f) Refined D-C method with (d)

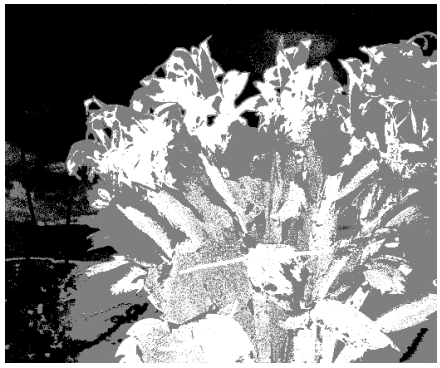
Figure 5.3 Panel A: Results of GT21 with Trimap

(a) Input image (GT21)

(b) Ground truth



(g) Scribbles



(h) Refined Trimap from (a) and (g)

(i) KNN matting [13] with (g)

(j) Refined D-C method with (h)

Figure 5.3 Panel B: Results of GT21 with Manual Scribbles

Chapter 6. Conclusion

As image size grows day by day, the need for high-resolution image matting increases. This paper proposes a solution to the potential memory shortage problem in high-resolution image matting. We propose an application of the divide-and-conquer method for high-resolution image matting with assistance of a refined trimap. We create a refined trimap based on a confidence map and the low-level alpha matting result. The refined trimap is created not only from existing trimap but also from sparse user input scribbles. The process of creating a refined trimap is algorithmically defined in this paper; we propose an automatic generation of the refined trimap, given an input image and a trimap. A memory shortage problem is caused by the fact that the size of matting Laplacian is proportional to a square of the image size. So, we propose that the divide-and-conquer method is a great solution to reduce the size of the matting Laplacian. To prevent discontinuity along the borders of divided image fragments in the merged alpha matte, we allow an overlapping margin in each image fragment and use the refined trimap. The refined D-C method can be used in high-resolution image matting using any matting method that is based on trimap or scribbles. As experimental results present, the memory shortage problem is mitigated with the refined D-C method. Moreover, the computation speed is vastly improved compared to existing methods, while maintaining a reasonable quality.

Bibliography

[1] Bern, M., Gilbert, J. R., Hendrickson, B., Nguyen, N., & Toledo, S. (2006). Support-graph preconditioners. *SIAM Journal on Matrix Analysis and Applications*, *27*(4), 930-951.

[2] Brandt, A. (1977). Multi-level adaptive solutions to boundary-value problems. *Mathematics of computation, 31*(138), 333-390.

[3] Briggs, W. L., Henson, V. E., & McCormick, S. F. (2000). *A multigrid tutorial*. Society for Industrial and Applied Mathematics.

[4] Cao, G., Li, J., He, Z., & Chen, X. (2016, September). Divide and Conquer: A Self-Adaptive Approach for High-Resolution Image Matting. In *Virtual Reality and Visualization (ICVRV), 2016 International Conference on* (pp. 24-30). IEEE.

[5] Feng, X., Liang, X., & Zhang, Z. (2016, October). A cluster sampling method for image matting via sparse coding. In *European Conference on Computer Vision* (pp. 204-219). Springer International Publishing.

[6] Gastal, E. S., & Oliveira, M. M. (2010, May). Shared Sampling for Real-Time Alpha Matting. In *Computer Graphics Forum* (Vol. 29, No. 2, pp. 575-584). Blackwell Publishing Ltd.

[7] Hackbusch, W. (2013). *Multi-grid methods and applications* (Vol. 4). Springer Science & Business Media.

[8] He, K., Rhemann, C., Rother, C., Tang, X., & Sun, J. (2011, June). A global sampling method for alpha matting. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on* (pp. 2049-2056). IEEE.

[9] Hoare, C. A. (1962). Quicksort. *The Computer Journal*, 5(1), 10-16.

[10] Jones, M. T., & Plassmann, P. E. (1995). An improved incomplete Cholesky factorization. *ACM Transactions on Mathematical Software (TOMS)*, 21(1), 5-17.

[11] Karacan, L., Erdem, A., & Erdem, E. (2015). Image matting with KL-divergence based sparse sampling. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 424-432).

[12] Levin, A., Lischinski, D., & Weiss, Y. (2008). A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(2), 228-242.

[13] Li, D. (2012, June). KNN matting. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 869-876). IEEE Computer Society.

[14] Rhemann, C., Rother, C., Rav-Acha, A., & Sharp, T. (2008, June). High resolution matting via interactive trimap segmentation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (pp. 1-8). IEEE.

[15] Rhemann, C., Rother, C., Wang, J., Gelautz, M., Kohli, P., & Rott, P. (2009, June). A perceptually motivated online benchmark for image matting. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (pp. 1826-1833). IEEE.

[16] Saad, Y. (2003). *Iterative methods for sparse linear systems*. Society for Industrial and Applied Mathematics.

[17] Shahrian, E., Rajan, D., Price, B., & Cohen, S. (2013). Improving image matting using comprehensive sampling sets. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition (pp. 636-643).

[18] Smith, A. R., & Blinn, J. F. (1996, August). Blue screen matting. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (pp. 259-268). ACM.

[19] Trottenberg, U., Oosterlee, C. W., & Schuller, A. (2000). *Multigrid*. Academic press.

[20] Wang, J., & Cohen, M. F. (2007, June). Optimized color sampling for robust matting. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on* (pp. 1-8). IEEE.

[21] WESSELING, P. *An Introduction to Multigrid Methods*. R.T. Edwards, Inc., 2004.

국 문 초 록

고해상도 이미지 매팅은 매우 큰 메모리 용량을 필요로 하기 때문에 일반적인 컴퓨팅 환경에서 수행이 어렵다. 본 논문은 기존 매팅 방법에 분할 정복(Divide-and-conquer) 방법을 적용하여 고해상도 이미지 매팅 시 발생하는 메모리 부족 문제를 해결하는 방법을 제시한다. 또한 원본 이미지와 트라이맵(또는 스크리블)을 바탕으로 개선된(Refined) 트라이맵을 만드는 알고리즘을 제시한다. 리파인드 트라이맵은 멀티그리드 방법을 차용하여 (1) 원본 이미지와 다운샘플링된 이미지를 비교한 신뢰도 맵과 (2) 다운샘플링된 이미지의 매팅 결과를 비교하여 자동으로 생성된다. 리파인드 트라이맵을 적용한 분할 정복 방법은 기존의 단순 분할정복과는 달리 가시적인 경계면의 불연속성을 최소화 한다. 본 논문이 제안하는 방법은 기존 고해상도 이미지 매팅 방식에서 문제가 되는 메모리 문제를 야기하지 않으며, 매팅 종류에 관계없이 고해상도 이미지에서 품질저하가 적은 알파 매트를 빠르게 도출한다.

주요어 : 이미지 매팅, 고해상도 이미지 매팅, 리파인드 트라이맵, 분할 정복, **신뢰도 맵**
학 번 : 2016-20911