

Portrait Extraction Algorithm Based on Face Detection and Image Segmentation

Chongyi Yang¹, Wanyu Huang¹, Ruoqi Zhang² & Rui Kong³

¹ Department of Electronic Information Science and Technology, Electrical and Information College, Jinan University, Zhuhai 519070, China

² Department of Software Engineering, Electrical and Information College, Jinan University, Zhuhai 519070, China

³ School of Electronic Information Science and Technology, Jinan University, Zhuhai 519070, China

Correspondence: Rui Kong, School of Electronic Information Science & Technology, Jinan University, Zhuhai 519070, China. E-mail: tkongrui@jnu.edu.cn

Received: February 8, 2019

Accepted: February 27, 2019

Online Published: March 25, 2019

doi:10.5539/cis.v12n2p1

URL: <https://doi.org/10.5539/cis.v12n2p1>

Abstract

Aiming to solve a series of problems in photo collection over citizen's license, this paper proposes Portrait Extraction Algorithm over our face based on facial detection technology and state-of-the-art image segmentation algorithm. Considering an input image where the foreground stands a man with unfixed size and its background is all sorts of complicated background, firstly we use Haar&Adaboost facial detection algorithm as a preprocessing method so as to divide the image into different sub-systems, and we get a fix-sized image of human face. Then we use GrabCut and closed-form algorithm to segment the preprocessed image and output an image which satisfies our requirements (i.e. the fixed size and fixed background). Up to now the GrabCut and closed-form algorithm has been realized, both of which have its own advantages and shortages.

Keywords: portrait extraction, image segmentation, GrabCut algorithm, Closed-Form algorithm, license photos' collecting system

1. Introduction

For individuals and social organizations, it is a tedious task to register and collect all kinds of residents' license photos. If registered in the social organizations, it tends to be time-consuming and labor-intensive. If residents upload photos on their own, there will be varieties of potential problems if image format does not meet the requirements. Every time residents are reluctant to spend time adjusting background color and portrait size for license photos. Exploring a convenient approach to solve such problems has become a hot topic at present. Suppose such approach can be widely applied in license photos' generation process, large amount of labor force will be reduced, and tedious photography procedures will be shortened. Furthermore, residents themselves may derive various kinds of license photos without leaving home. Using digital image processing technology, front-view portraits surrounded by arbitrary background can be recognized, extracted and processed, and foreground area (i.e. the portrait extracted and processed from the original photograph) is inserted into a specified background color. Via this original measure we can easily derive the qualified license photo eventually. Needless to say, it provides huge convenience for individual residents and social organizations. Most importantly, the idea conforms with the principle of "user-centered" in current well-known Internet of Things system.

Lots of solutions over image matting have been proposed in image processing domain. In 1996, Alvy Ray Smith and James F. Blinn made a definition of image matting problem via a brief formula $I = \alpha F + (1 - \alpha) B$. Given an image specified by I , it can be thought of as a composition of the foreground and background which can be differentiated by grade of transparency specified by α . The problem is how to deduce α , F and B at the right side according to I at the left side of equation (Smith & Blinn, 1996). Later, a new Bayesian framework was proposed to solve the matting problem. They use spatially varying Gauss mixture colors to model the distribution of foreground and background colors, and assume that a partial mixture of foreground and background colors produces the final output. Then the maximum likelihood criterion is used to estimate the optimal opacity, foreground and background simultaneously (Chuang, 2001). Jian Sun et al. proposed using tri-map as an auxiliary

tool. Tri-map is divided into three colors: black represents the complete background (where alpha is 0), white represents the complete foreground (where alpha is 1), and gray represents the uncertain region (where alpha is undetermined) (Sun, 2004). Yuanjie Zheng and Chandra KambhmettuX propose that for each point, a linear combination of the surrounding pixels alpha is expected to be used for prediction. The parameters of this linear combination are obtained by learning. This learning is a process of establishing the correlation between alpha and color feature vectors. When establishing the relationship between alpha and color eigenvector, the form of inner product appears in the analytic solution of the problem. Kernel Trick can then be introduced to elevate dimensions and learn more complex relationships (Zheng & Kambhmettu, 2009). Xiaoyong Shen et al. constructed a portrait image database using the traditional matting method. On this basis, a matting method based on CNN is proposed. There are two functions. The first one uses CNN to classify the pixels into three categories: foreground, background and uncertain label. The second is a new matting layer, which obtains matting information by propagating forward and backward (Shen, 2016).

This paper proposes to design an original Portrait Extraction Algorithm based on facial detection technology as well as state-of-the-art foreground segmentation algorithm. The original intention of our algorithm system is to provide a solution to license photos in daily life. Furthermore, the algorithm system can be applied in various domains such as facial recognition technology, computer vision and machine learning under recent tremendous development of artificial intelligence. In the process of designing the human image extraction algorithm system, this paper uses the face detection algorithm based on Haar&Adaboost to design the natural image segmentation subsystem. At the same time, the advanced Closed-Form algorithm is used to design a fixed size image segmentation subsystem.

2. Theoretical Preparation for Portrait Extraction Algorithm System

In this section, the article introduces some image preprocessing method including grayscale transform, Gaussian filtering and histogram equalization in detail. Then we mainly focus on a facial detection algorithm called "Haar&Adaboost".

2.1 Image Preprocessing Technology

2.1.1 Gray

Gray refers to the conversion of color space (such as RGB , $YCbCr$, etc.) into gray space. Gray image is represented by black with different saturation. Taking the conversion of RGB into grayscale image as an example, the commonly used conversion formulas are listed as follows:

- Psychological approach (based on the human eye's perception of color and brightness):

$$Gray = 0.299 * R + 0.587 * G + 0.114 * B \quad (1)$$

- Average method:

$$Gray = (R + G + B) / 3 \quad (2)$$

- Floating point method:

$$Gray = R * 0.3 + G * 0.59 + B * 0.11 \quad (3)$$

- Integer method:

$$Gray = (R * 30 + G * 59 + B * 11) / 100 \quad (4)$$

2.1.2 Histogram Equalization

The histogram equalization algorithm is shown below. The gray level of the image to be processed is M (gray level: $0, 1, \dots, M-1$), for example (Gonzalez, 2007):

Step1: List M -level grayscale images

$$D_A = (0, 1, \dots, M-1)$$

Step2: Statistical analysis of gray level pixels in the original image

$$H(k)$$

Step3: Computation of Normalized Histogram Array

$$C(k) = \frac{H(k)}{A^0} \quad (5)$$

Step4: Computation of an array of cumulative histograms

$$T(k) = \sum_{i=1}^k C(k) \tag{6}$$

Step5: Calculating the Gray Accumulated Distribution Array

$$D_B(k) = \text{int} (7 * t + 0.5) \tag{7}$$

Step6: Determining Gray Mapping Relation

$$D_A \rightarrow D_B$$

Step7: Statistical New Histogram Gray Level Pixel Array

$$H(k)$$

Step8: Computation of a New Normalized Histogram Array

$$C(k) = \frac{H(k)}{A^0} \tag{8}$$

2.1.3 Gaussian Filtering

The principle of Gauss smoothing is that in a given neighborhood, the average value of all the pixels is taken as the result of the final calculation. The weights of each pixel are different and the weights are determined by the Gauss function.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{9}$$

The square of x and y represent the distance between other pixels in the neighborhood and the central pixel in the neighborhood respectively, while the square of σ^2 represents the standard deviation.

The two-dimensional distribution of the function is roughly as follows (drawn by MATLAB):

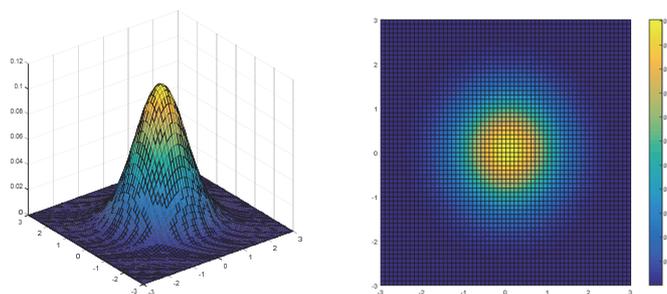


Fig1.2D Distributional Graph of Gaussian Function

It is easy to see that the smaller the standard deviation is, the smaller the two-dimensional Gauss image will be, and the less obvious the smoothness effect will be. The larger the standard deviation is, the shorter and wider the Gauss image is, the more obvious the filtering effect will be.

2.2 Facial Detection Solution: Haar&Adaboost Algorithm

Haar&Adaboost face detection algorithm, first proposed by Paul Viola in the paper Robust Real-time Object Detection, has been improved and widely used in today's face detection and face recognition because of its high accuracy and efficiency.

(1) Application of Haar-like Feature

Haar feature template was originally proposed by Viola and was then improved and derived from Haar-like feature afterwards. Haar-like has generally four types: edge feature, linear feature, circle center surround feature and specific direction feature. Corresponding eigenvalues are obtained by using white area pixels and subtracting black area. These four kinds of eigenvalues describe the characteristics of discriminant objects in different directions. Additionally, the positive and negative samples have different eigenvalues, so they can be used as descriptors for pattern recognition. A large number of Haar-like features will be generated according to the selected coordinates, rectangle size and category. The Haar-like eigenvalues of image samples can be quickly calculated by integral graphs. (Viola & Jones,2001) (Lienhar & Maydt, 2002)

(2) Procedures of Adaboost Algorithm

Adaboost changes the distribution weight of training samples to obtain different test samples. The algorithm aims to obtain an optimal weak classifier after each training and obtains the weight of each weak classifier when constituting a strong classifier according to its classification error. According to the results of each training, increase the weight of the wrong classification sample and decrease the weight of the correct classification sample simultaneously, and a new training sample set is obtained. The specific algorithm flow of Adaboost is:

Giving sample space set $S = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$, X is sample space, Y is tag set and $Y = \{1, -1\}$, $x_i \in X, y_i \in Y$.

Step 1: Initialize training sample weight

Initialize weight to $1/N$: $D_1 = (w_{11}, w_{12}, \dots, w_{1i}, \dots, w_{1N})$ and $w_{1i} = 1/N, i = 1, 2, \dots, N$

Step 2: Training weak classifier

Use $m = 1, 2, \dots, M$ to represent iteration times, as well the numbers of weak classifiers.

- (a) The samples with weight distribution D_m are trained and learned, select the threshold which minimizes the detection error rate as the current weak classifier:

$$G_m(x): x \rightarrow \{-1, 1\}$$

- (b) Calculating the Classification Error of $G_m(x)$ on Training Samples:

$$e_m = P(G_m(x_i) \neq y_i) = \sum_{i=1}^N w_{im} I(G_m(x_i) \neq y_i) \quad (10)$$

- (c) Calculating $G_m(x)$ coefficient α_m . α_m represents the importance of $G_m(x)$ in the final strong classifier:

$$\alpha_m = \frac{1}{2} \log \frac{1-e_m}{e_m} \quad (11)$$

- (d) Updating the weight of training samples to get the next training samples:

$$D_{m+1} = (w_{m+1,1}, w_{m+1,2}, \dots, w_{m+1,i}, \dots, w_{m+1,N}) \quad (12)$$

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} e^{(-\alpha_m y_i G_m(x_i))}, i = 1, 2, \dots, N \quad (13)$$

$$Z_m = \sum_{i=1}^N w_{mi} e^{(-\alpha_m y_i G_m(x_i))} \quad (14)$$

$$Z_m \text{ is normalization factor, makes } \sum_{i=1}^N D_{m+1}(x_i) = 1 \quad (15)$$

Step 3: Training strong classifier

$$G(x) = \text{sign}(\sum_{m=1}^M \alpha_m G_m(x)) \quad (16)$$

2.3 Portrait Extraction Solution I: GrabCut Algorithm

Step1 Initialization

- Users select the region of interest through the mouse to set the background T_B to initialize the ternary graph T , foreground is empty, $T_F = \Phi$; T_U is complement of background complement, $T_U = \overline{T_B}$.
- For $n \in T_B$, there is $\alpha_n = 0$; for $n \in T_U$, there is $\alpha_n = 1$.
- Use $\alpha_n = 0$ and $\alpha_n = 1$ these two aggregates to initialize the GMM model of foreground and background.

Step2 Iterative minimization

- Obtaining GMM model parameters k_n corresponding to each pixel n in T_U requires solving the equation of k_n as follow:

$$k_n = \text{argmin} D_n(a_n, k_n, \theta, z_n) \quad (17)$$

- Get GMM model parameters θ from data Z . The corresponding equation of θ is:

$$\theta = \text{argmin} U(\alpha, k, \theta, z) \quad (18)$$

- Using minimum energy to get initial segmentation:

$$\min_{\alpha_n=T_B} \min_k E(\alpha, k, \theta, z) \quad (19)$$

- d) Repeat from step 4 until convergence.
- e) Perform boundary optimization and process it with continuous values. There is a transparent narrow band near the hard segmentation boundary. The main task of boundary optimization is to count the image of this narrow band, which can solve the problem of blurring and pixel overlap at the boundary.

Step3 User Interactive Editing

- a) Specify some pixels $\alpha_n = 0$ (background) or $\alpha_n = 0$ (foreground), correspondingly update ternary graph T.
- b) Optimize: Perform the entire iterative minimization algorithm. Iterative minimization can be seen as a monotonic decrease of total energy E in three aspects k, θ, α . In this way, the algorithm can guarantee the final convergence to the minimum value of E. Automatic termination of iteration when no significant attenuation is judged, this ensures the convergence of the algorithm. (Rother & Kolmogorov, 2004)

2.4 Portrait Extraction Solution II: Closed-Form Algorithm

Step1 Initialize

- a) Users select the region of interest through the mouse box to provide an initial trimap, foreground F (shown in white), background B (shown in black) and unknown α (shown in gray).
- b) α_i is transparency, $\alpha_i \in [0,1]$; F_i is foreground pixels, B_i is background pixels.
- c) Color Composition Model:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i \quad (20)$$

- d) When $\alpha_i = 0, I_i \in B_i$; When $\alpha_i = 1, I_i \in F_i$.

Step2 Global optimum α

- a) Assuming that there is a 3*3 small window w nearby, and the foreground color F and background color B in the small window remain roughly unchanged, so α change linearly in the small window w :

$$\alpha_i \approx \alpha I_i + b, \forall i \in w \quad (21)$$

- b) Definition of energy function

$$J(\alpha, a, b) = \sum_{j \in I} (\sum_{i \in w_j} (\alpha_i - a_j I_i - b_j))^2 + \varepsilon \alpha^2 \quad (22)$$

Suppose a, b is constant, there is only one energy function about α :

$$J(\alpha) = \alpha^T L \alpha \quad (23)$$

In the formula: α is an $N*1$ vector; N is the number of vectors; L is a matting Laplacian matrix, which is a $N*N$ symmetric matrix.

- c) Find the element at (i, j) in the symmetric matrix:

In the formula: δ_{ij} is Kronecker sign; μ_k and σ_k^2 are local window color and respectively; $|w_k|$ is the number of pixels included in local window w_k .

- d) The minimum of energy function $J(\alpha) = \alpha^T L \alpha$ is global optimum α .

Step3 Expanding into Color Images

- a) Use 4D linear model to replace the Step2-a) linear model:

In the formula: c is the sum of all color channel.

- b) In a local window F_i is on a line of RGB color space, the same to B_i , there is:

$$F_i = \beta_i F_1 + (1 - \beta_i) F_2 \quad (24)$$

$$B_i = \beta_i^B B_1 + (1 - \beta_i^B) B_2 \quad (25)$$

- c) The colour image matting model is:

$$I_i^c = \alpha_i (\beta_i^F F_1^c + (1 - \beta_i^F) F_2^c) + (1 - \alpha_i) (\beta_i^B B_1^c + (1 - \beta_i^B) B_2^c) \quad (26)$$

d) So get the function:

$$J(\alpha, a, b) = \sum_{j \in I} (\sum_{i \in w_j} (\alpha_i - \sum_e a_j^e I_i^e - b_j)^2 + \varepsilon \sum_c a_j^{c^2} \tag{27}$$

e) Form Step2 we know $J(\alpha) = \alpha^T L \alpha$

In the formula: L is a N*N matting Laplacian matrix, the (i, j) of the matrix is:

$$\sum_{k|(i,j) \in w_k} (\delta_{ij} - \frac{1}{|w_k|} (1 + (I_i - \mu_k)(\sum k + \frac{\varepsilon}{|w_k|} I_3)^{-1} (I_j - \mu_k))) \tag{28}$$

Step3 User Interactive Editing

The user re-labels the selected foreground *F* and background *B*, re-performs the operation.(Levin,2006)

3. Portrait Extraction Process & Result Analysis

3.1 Previous Work I for Closed-Form Algorithm: Image Cutting

Before embarking on scribbling images and foreground extraction process, we are required to resize and cut the arbitrary portrait into one that only contains face and shoulders in order to generate standard license photos eventually. These procedures are categorized as “experimental preparation” or “image preprocessing process”. In this section, we resize the input images into a fixed size that fits our requirements (i.e. each resized image shares the same length and width). Then we manage to obtain facial detection via Haar&Adaboost algorithm, which provides a rough position of face appeared in image. At last we cut the resized image according to the position, leaving the face and shoulders only. Additionally, we restrict the input image to be front view and at least includes face and the upper part of the body. Any input images beyond our restriction may not work out proper cut images.

At the beginning, we need to resize input image because Haar&Adaboost algorithm to be used in the next step requires strict image size. In our experiment we resize any kind of input image to be 800*1200. Here we select three original images, both of which strictly fit all requirements over input image. Then we resize them respectively. All the original images and resized images are shown in table 1 below.

Then we apply Haar&Adaboost algorithm to realize facial detection and work out the rough position of the face. At last we apply equal-ratio algorithm to expand the frame around the position we've got already. In this step we reserve the face and two shoulders only inside the frame. Adjust the frame size as big as the one appeared in identity card, cut the image according to the frame and thus we get the proportional segmentation result shown in the rightmost column.

Table 1. Original Images and Resized Images

No.	Original Image	Resized Image	Cut Image
Test 1			



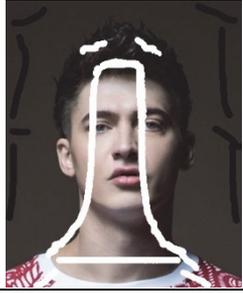
The results above already fit our requirement. In the column of original image, all of the input images are front-view portrait and include the whole face and the upper part of the body. After resizing, all images are reduced to 800*1200. At the same time, the whole face and upper part of the body is completely reserved. The rightmost column demonstrates the final output of preparation process. Consider the basic requirements of license photos that the whole face and the upper part of two shoulders should be included, we adjust the frame size so that the face and shoulders can be included inside the frame as much as possible. Through the rightmost column in table 5 we find all of the cut images conform with the basic requirements. Though the top of the hair in test 2 has been omitted, the proportion of face and shoulders as well as the man size appeared in the whole image seem to be harmonious.

3.2 Previous Work II for Closed-Form Algorithm: Image Scribbling

Scribble is a user-supplied constraint which points out the foreground and background by scribbling over the image. According to formula xxx, foreground and background can be differentiated via α . $\alpha = 1$ indicates foreground and $\alpha = 0$, on the contrary, indicates background. In scribbling process, it is recommended to draw edge information (i.e. to scribble the rough foreground profile) as much as possible. scribbles specify foreground because white color indicates $\alpha = 1$ and black scribbles, similarly, specify background area. In closed form matting and Grabcut, users are required to provide trimap or scribbles as an input so as to obtain initial estimation of alpha matte. In scribbling images, users are only required to draw a couple of scribbles in two colors, one for foreground and the other for background. Considering simplicity, we generate scribbling images where users can easily interact with computer during scribbling.

We complete the scribbling image via Photoshop where users can easily apply their constraints upon input images. We find the output of image preprocessing above is quite satisfactory—almost all the cut images share identical foreground size position and profile, so we attempt to apply the same scribbles towards all the cut images. Table 2 below clearly demonstrates three cut images and their own scribbling images.

Table 2. Cut Images and Scribbling Images

No.	Cut Image	Scribbling Image
Test 1		
Test 2		
Test 3		

For every cut image shown above, we apply the same scribbling image. Though cut images differ from their foreground size and profile, the difference is quite trivial and can be ignored. From the rightmost column above we find that the scribbles fit all the cut images (i.e. normally white scribbles always appear upon the portrait and black scribbles upon background area). Furthermore, the white scribbles can effectively reflect the edge information of the portrait. Overall, using the same scribbling image save a lot of time when applying user-supplied constraint, but its precision is guaranteed.

3.3 Portrait Extraction Results and Analysis

All the previous work mentioned above contributes to the input of image segmentation algorithms. In this paper we apply two different image segmentation algorithms to realize foreground extraction. Here we still use the above three cut images and obtain the foreground extraction images respectively via two algorithms.

3.3.1 Portrait Extraction Result via GrabCut

In this section we use GrabCut algorithm to extract foreground image. GrabCut is an iterative algorithm for image segmentation where users are not required to provide scribbling image or trimap as an input. Thus, GrabCut algorithm is more convenient and easier-to-use. However, it consumes a large amount of computer resources, and thus its performance mostly depends on user's computer level. In addition, it takes a long period of time to segment a small input image. Table 3 below demonstrates the image segmentation result using GrabCut algorithm. The first column from the left lists three cut images as an input and the second column is the output.

Table 3. GrabCut Results

No.	Cut Image	Foreground Extraction Result
Test 1		
Test 2		
Test 3		

As we can see from the result above, the image segmentation result is far from satisfactory. The common fatal problem of GrabCut is the omission of the person’s shoulders. We can deduce from the result that GrabCut takes the shoulders as background content and only the face, the neck and the collar are served as foreground area. Additionally, GrabCut performs badly when foreground color and background color become similar. Just like the result in test 3, the background above two ears is misjudged as foreground area, which leads to an awful segmentation result.

Nevertheless, GrabCut algorithm may still be a possible solution when the foreground color and background color is highly differential. In addition, background area should not contain sharp variation of color so as to avoid misjudgment as much as possible. Compared with test 3, result in test 1 and 2 seems better.

3.3.2 Portrait Extraction Result via Closed-Form

In order to obtain foreground extraction result within a shorter time, we use closed form algorithm to complete image segmentation. In closed form algorithm, a cost function upon α is defined to eliminate F and B . Differed from those complicated iterative algorithms, closed form solution requires less computing time and the result is also good. What’s more, we can obtain a relatively high-quality alpha matte from very simple scribbling images.

Both the cut image and the scribbling image generated from previous work are necessary in closed form solution. Scribbles here contribute to the initial estimation of alpha matte. Considering the computer level and computing time to be consumed, we set the window with a small 3*3 size to run the algorithm and generates alpha matte. Firstly, we use the cut image and corresponding scribbling image as an input, and we obtain the alpha matte via closed form algorithm. We designate red color ($RGB = (255, 0, 0)$) as background color in final license photo in our test, so we replace the black area in alpha matte (i.e. the background area) into red and we get the background processing result. Finally, we fill the foreground into the white area of the background processing image and thus

we obtain the license photo.

In table 4, we select the same three cut images and scribbling images as an input. The respective algorithm result (i.e. the alpha matte) is obtained in the third column from the left. Background processing result is shown in the fourth column from the left. Finally, we filled the white area in background processing image with cut image and we generate license photos.

Table 4. Closed-Form Results

No.	Cut Images	Alpha Matte	Background Processing Result	Final Result (License Photo)
Test 1				
Test 2				
Test 3				

We find the image segmentation result is much better compared with GrabCut. The whole portrait, including the face and the upper part of the body, has been basically preserved during image matting.

Of course, there are still some flaws in the final result. Observe the alpha matte carefully and we find there's a relatively big gray area along the edge of the foreground area, which leads to misjudgments in final result. As we can see in test 2 and test 3, both of the shoulders' edge become gray, thus the background near the shoulders may be misjudged into foreground (see test 3) or the foreground in the shoulders may be misjudged into background (see test 2). Sometimes when background color and foreground color are hard to identify, background may be misjudged as foreground (see test 1).

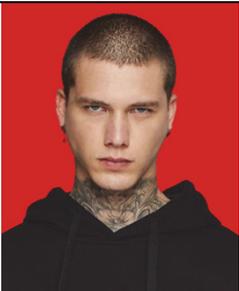
4. Discussion

Image matting plays an essential role in image or video editing and trimming. With the rapid development of computer vision, not only should we pursue high-quality image segmentation result, but we should minimize the computing time as much as possible, especially in video editing domain.

Table 5 again lists five different input images and we obtain the image segmentation results under closed form and GrabCut solution respectively. From the results we summarize and compare with the advantages and shortages of

these two image matting algorithms.

Table 5. Image Segmentation Result Comparison between Closed Form and GrabCut

No.	Cut Image	Image Segmentation Results	
		Closed Form Solution	GrabCut Solution
Test 1			
Test 2			
Test 3			
Test 4			
Test 5			

1. Image segmentation result via GrabCut is far from satisfactory. Sometimes even the eyes and lips are considered as background (see test 1, 2 and 3); for lots of input images, a big part of background area is

- misjudged as foreground (see test 3, 4, 5). Another fatal problem is the omission of two shoulders (see all tests).
2. Compared with GrabCut, closed form solution is much better. Nearly all the edge information can be preserved in all tests. Few of the background is misjudged as foreground. The simpler the background is, the more precise the result will be. However, there's still some problems around two shoulders. Sometimes the background near the neck or the shoulders may be misjudged as foreground, and vice versa (see test 3, 4 and 5). Additionally, it is still hard to identify foreground and background successfully when both of them share similar colors (see test 3 and 5).
 3. Compared with two algorithms, closed form solution is less dependent on computer resources, which contributes to a shorter computing time, but requires user-supplied constraints. In other words, users' interaction with computers is indispensable, while GrabCut algorithm can work automatically.
 4. From group one to five, background complexity increases. When background color remains steady and background color has a great difference to foreground color, the result under closed form solution is pretty good. Even though the background complexity reaches the highest (see test 5), we can still extract most of the foreground correctly. But GrabCut misjudgments is more frequent and seem to be irregular.

Consider the summaries and results above, we find that GrabCut algorithm has a worse manifestation compared with that of Closed Form algorithm due to its high error rate and unregular misjudgments however simple the background is. On the contrary, closed form algorithm has a higher stability. The simpler the background color is, the more ideal the result will be. Also, the more differentiable the background and foreground color are, the more qualified the result will be. Although the background near the left neck is sometimes mistaken as foreground, the problem is more apparent and regular compared with GrabCut.

While the closed form algorithm may work out a nice image segmentation result, we still need to find the inner problem of our algorithm system. Since the problem becomes obvious and regular over a group of tests, it provides a clear direction for us to find out the reason.

Acknowledgments

This research was funded by the National Innovative Experiment Program for Undergraduate Students of Jinan University with the grant number No. 201810559056. The authors are thankful to School of Electrical Information, where the experiment is accomplished, and the significant guidance from our mentors Rui Kong.

References

- Achanta, R., Hemami, S., Estrada, F., & Susstrunk, S. (2009). Frequency-tuned salient region detection. <https://doi.org/10.1109/CVPR.2009.5206596>
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*.
- Boykov, Y. Y., & Jolly, M. P. (2001). Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. *In Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, 1, 105-112. IEEE.
- Chuang, Y. Y., Curless, B., Salesin, D. H., & Szeliski, R. (2001). A Bayesian approach to digital matting. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. IEEE.
- Gonzalez, R. C., & Woods, R. E. (2007). *Digital Image Processing*. Publishing House of Electronics Industry.
- Grady, L. J., & Polimeni, J. R. (2010). *Building a Weighted Complex from Data. Discrete Calculus*.
- Gupta, V., & Raman, S. (2017). Automatic trimap generation for image matting. *International Conference on Signal & Information Processing*. <https://doi.org/10.1109/ICONSIP.2016.7857477>
- Levin, A. (2006). A closed form solution to natural image matting. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. IEEE Computer Society. <https://doi.org/10.1109/CVPR.2006.18>
- Levin, A., Lischinski, D., & Weiss, Y. (2004, August). Colorization using optimization. *In ACM transactions on graphics (tog)*, 23(3), 689-694. ACM. <https://doi.org/10.1145/1015706.1015780>
- 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. <https://doi.org/10.1109/TPAMI.2007.1177>

- Lienhart, R., & Maydt, J. (2002). An Extended Set of Haar-like Features for Rapid Object Detection. *Image Processing. 2002. Proceedings. 2002 International Conference on. IEEE.* <https://doi.org/10.1109/ICIP.2002.1038171>
- Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1), 62-66. <https://doi.org/10.1109/TSMC.1979.4310076>
- Rother, C., Kolmogorov, V., & Blake, A. (2004, August). Grabcut: Interactive foreground extraction using iterated graph cuts. *In ACM transactions on graphics (TOG)*, 23(3), 309-314. ACM. <https://doi.org/10.1145/1015706.1015720>
- Shen, X., Xin, T., Gao, H., Chao, Z., & Jia, J. (2016). Deep Automatic Portrait Matting. *European Conference on Computer Vision*.
- Smith, A. R., & Blinn, J. F. (1996). [acm press the 23rd annual conference - not known (1996. -.)] proceedings of the 23rd annual conference on computer graphics and interactive techniques, - siggraph \96 - blue screen matting. *Proc Siggraph*, 259-268.
- Sun, J., Jia, J., Tang, C. K., & Shum, H. Y. (2004). Poisson matting. *ACM Transactions on Graphics*, 23(3), 315-321. <https://doi.org/10.1145/1015706.1015721>
- Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. null. *IEEE Computer Society*. <https://doi.org/10.1109/CVPR.2001.990517>
- Zheng, Y., & Kambhamettu, C. (2009). Learning based digital matting. *2009 IEEE 12th International Conference on Computer Vision (ICCV). IEEE Computer Society*.

Copyrights

Copyright for this article is retained by the author(s), with first publication rights granted to the journal.

This is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/4.0/>).