MULTIPLE COLOR TRANSFER: EXAMPLE-BASED PHOTO ENHANCEMENT

[1]*Chang-Hsi Hsieh* (謝昌熹), [2]*Bing-Yu Chen* (陳炳宇), [3]*Ming Ouhyoung* (歐陽明)

National Taiwan University
E-mail: [1]isaddo@cmlab.csie.ntu.edu.tw, [2]robin@ntu.edu.tw, [3]ming@csie.ntu.edu.tw

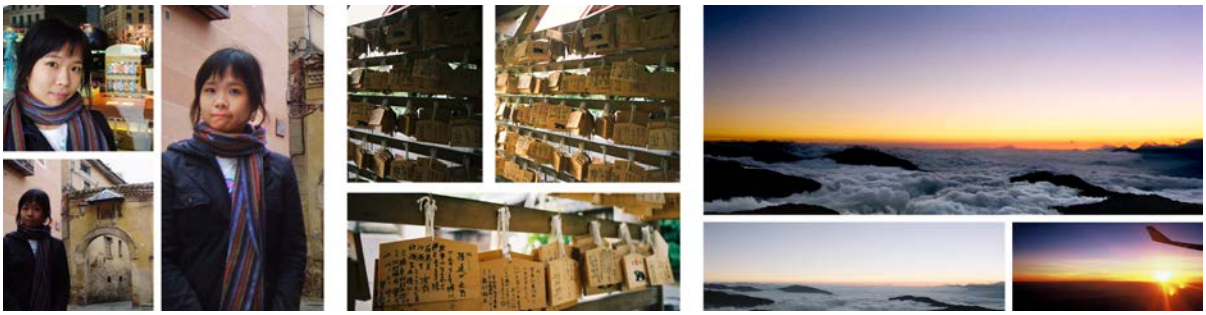


Fig. 1: Here are three sets of examples of why a user might be motivated to use this tool. The left smaller two photos are taken at two different places in one day. The middle two are the same scene taken from different views. The right two are captured a similar object on an airplane and on a mountain peak respectively. The bigger images are results enhanced by using this tool.

ABSTRACT

One of the most common situations in photo editing is that users want some specific effects but don't know how to achieve them. Nevertheless, they have an image in their mind about how the photo should be, which is derived from a better photo on similar subjects, or from their imagination. With this user motivation in mind, this thesis proposes a method for altering a photo's color based on the coloring of another exemplary photo. Users simply set the requirement by drawing some pairs of brush strokes in corresponding regions and then the tool will perform the editing automatically.

1. INTRODUCTION

Nowadays, it costs less than before to take a photo. With the ease of taking photos, a person can have several photos of the same subject shot by one or more cameras. For example, when people take a photographic portrait or a group photo, they usually take two or three times to avoid unexpected results. In a famous or interesting place, the photos taken may have high similarity. A photographer often takes one scene from different views to find different interpretations. Fig. 1 shows three examples of common cases. However, users always find out some unsatisfying photos in a sequence of photos because of it's against the light, the incorrect camera setting or other reasons. Even so, users can easily find a satisfying photo of the similar subject. Can they use the information of one similar photo to edit the other? Actually, as photo editors, we have rich materials to rely on for editing photos. We can easily exploit those data to create a better photo. This is the goal of our photo editing tool.

Related work Several methods have been developed recently to enhance the ability of digital photography by combining multiple photographs of one scene into an image of better quality. For example, Jia [9] considered a pair of photos in a dark environment: one is motion blurred but accurate in color; the other is underexposed but edge-preserved. One can repair the latter by using histogram matching and spatial constraints simultaneously. Similarly, Eisemann [8] and Petschnigg [14] developed another approach which can produce a picture of rich details and original lighting by combining one photo with flash to capture details with the other without flash to capture ambient illumination. Moreover, as Agarwala [1] shows an interactive digital photomontage method, an interactive framework can

compound selected parts from a set of photographs into a single picture. Besides, Agarwala [2] also used a flash and no-flash image pair to remove the reflection and highlights from the flash image.

However, these methods usually need extra equipment, such as a tripod or a high-quality flash lamp. Another problem is there are various pre-process methods. That is, users need to realize they will use those techniques when taking the photo, and sometimes the knowledge of photography is required.

There are many factors that can affect feelings of an image. Bae [3] propose an approach to transfer the "look" of an image from one to another on black and white image or luminance channel. About altering an image's color, Reinhard [15] developed an easy and successful method to transfer colors from one image to another. However, this method is not suitable for all cases. Their approach is a global one; the all pixels apply the same transfer function. The problem is if the source and target contain different color regions, the colors may embroider to the other regions (Fig. 5). Some researches [17, 6, 7] tried to solve the problem of how to match the correct corresponding region. To connect the corresponding regions accurately is the key of transferring color naturally, but also a very difficult task.

Our approach This thesis chooses an interactive method to get the information of how the photo should be from users to solve the problem of match the correct corresponding region, and the information is used to build multiple color transfer functions. Then, we use weighted average to set the color transfer parameters of each pixel. Next, after an optimization process, each pixel is altered by a unique transfer function. Lischinski [13] presented a general optimization approach to optimize the transfer function on an image. We use the same function for minimization but design a new method to set full constraints on the image, instead of sparse constraints.

Besides, usually users are just dissatisfied with a part of the photo, so this thesis uses the image cutout technique to clamp the editing region for altering photo partially. Wang [18] analyzed the user's intention behind the interaction, and added a user intention term in their frame work. However, they only use the direct proportion of distance of the stroke to estimate the energy when the user adds the additional stroke. We have designed a new, adaptable energy function based on another users' drawing habits to estimate the importance of spatial and color information of strokes. Finally, we used the joint bilateral upsampling [10] to speed up our system.

In summary, we propose an easy-to-learn and easy-to-use tool for photo editing. All users need to do is to find another photo they prefer and simply set their requirement by an intuitional interface. Then, the desired colors will be transferred to the expected regions. No extra photographic equipment and knowledge is required, and users don't need to change their habits of taking photos.

2. PREVIOUS WORK

Before introducing our approach, we review some previous works closely related to the core of our approach.

Color transfer between images Reinhard [15] showed a simple algorithm to transfer the color characteristics between images. Their core strategy is to choose a suitable color space and then to apply simple operations there. They chose the $l\alpha\beta$ space, a low correlation color space, so they can alter color value in each channel independently by this transfer function,

$$T(c_p) = (c_p - \mu_s)\frac{\sigma_t}{\sigma_s} + \mu_t. \quad (1)$$

Here c_p is the color value of the pixel p. μ_s and μ_t are the means of the source image and the target image; σ_s and σ_t are their standard deviations. Actually, it is a linear transfer to match the mean and variance of target image. We can think of this function as three separated processes: shifting, scaling and then shifting. That is, there are three parameters, μ_s, σ_t/σ_s and μ_t in the processes.

Image-guided optimization Lischinski [13] presented a general approach that is able to apply effects locally. In their framework, the user sets some constraints by drawing some strokes and then the effects will propagate to the region having similar luminance. They performed this task by minimizing this quadratic function,

$$f = \arg\min_f \left\{ \sum_{\mathbf{x}} w(\mathbf{x})(f(\mathbf{x}) - g(\mathbf{x}))^2 + \lambda \sum_{\mathbf{x}} h(\nabla f, \nabla L) \right\}, \quad (2)$$

$$h(\nabla f, \nabla L) = \frac{|f_x|^2}{|L_x|^\alpha + \varepsilon} + \frac{|f_y|^2}{|L_y|^\alpha + \varepsilon}.$$

Here, x is pixels in the image; $g(\mathbf{x})$ is users' constraints. The first term is the square of f minus g so it maintains the solution f will be as close g as possible, and $w(\mathbf{x})$ is the weight of constraints, $g(\mathbf{x})$. The second term is the smoothing term; L is the log-luminance channel of the image, and the subscripts x and y denote spatial differentiation. It ensures the consistency between neighbor values in f, but it is allowed to have a rapid change across significant edges. Therefore, it can have different effects in different regions.

Energy minimization via graph cuts Boykov [5, 4] developed a graph cut optimization algorithm to solve the energy minimization problem for a energy function E, which is typically formulated as

$$E(f) = E_{smooth}(f) + E_{data}(f). \quad (3)$$

Each possible labeling f is assigned a value of energy according to the energy function, $E(f)$. The goal of this technique is to find a labeling f that assigns each pixel a label

and minimize the value of energy. This technique has been used for a variety of tasks, including image segmentation, stereo matching, optical flow, and photomontage. You can get the desired result by designing a suitable energy function carefully. These papers [16, 1, 12, 11, 18] perform the task of extracting a part of image by this technique. The difference between them is they designed the different energy functions respectively for their difference objectives of application.

3. EXAMPLE-BASED PHOTO ENHANCEMENT

3.1. User interface design

The goal of our tool is to enable the photo editor to easily alter the dissatisfied photo by referring the other preferable photo. A suitable user interface has designed for drawing the corresponding regions between a source and one or more targets. Users are allowed to edit their photo completely or partially by the same work flow, and the difference is only whether you add the label in source but not in target. Our algorithm can preserve the satisfying or exediting region successfully. Therefore, the user can edit in a large scale first and correct the details further (Fig. 7).

3.2. User requirement

The user is allowed to use arbitrary color to label the region. After the user draws strokes, we first analyze the different meaning of each stroke. We can think of each color as a label and set all the used labels as a set $\mathcal{L}$. The subsets, $\mathcal{S} \subset \mathcal{L}$ and $\mathcal{T} \subset \mathcal{L}$, express the labels used in source and target. Furthermore, our tool defines if you use a color to label in source but not in target, it means you want to preserve this region (Fig. 2). That is,

$$\mathcal{F} = \mathcal{S} \cap \mathcal{T}, \quad \mathcal{B} = \mathcal{S} \setminus \mathcal{T}.$$

Here $\mathcal{F}$ includes the labels that mark the editing region, and $\mathcal{B}$ includes the labels that mark the preserving region. In addition, we use p as the notation for a pixel; c_p and l_p are its color and label, and I_s is the source image. For clamping the editing region, we need to find some seeds to represent the editing region and the preserving region. Therefore, we collect the pixels under strokes and build the sets,

$$P_F = \{p \mid p \in I_s \text{ and } l_p \in \mathcal{F}\},$$

$$P_B = \{p \mid p \in I_s \text{ and } l_p \in \mathcal{B}\}.$$

Here P_F and P_B are the editing region seeds and preserving region seeds. We will use the graph cuts algorithm, which is often used in the image cutout task. Therefore, for understanding easily and matching the general names, we may call the editing region foreground and call the preserving region background sometimes; the strokes color in $\mathcal{F}$ calls the foreground strokes, and others are background strokes.

Fig. 2: (a) The strokes in the source (b) The strokes in the target The light blue stroke was drawn for preserving the background. (c) The result that only constrains the pixels under the strokes. (d) our result, the background is preserved completely, and the scarf is successfully recovered. (e) The result of progressive cut in first step. The color of the wall is similar a part of the face, and the shadows like the cloths, the T-shirt is white, so the result have some errors. (f) Our editing region. It has a satisfying result in one step.

3.3. Clamp editing region

Before alteration, this is a pre-process to clamp the editing region for two reasons. One is to get the more precise editing region to avoid producing the unexpected effect in the region user want to preserve. The other is we can set constraints largely and accurately in the optimization stage. Setting constraints as many as possible is helpful to create a better result, although the equation (2) can propagate the effect. As the Fig. 2 shows, if we only constrain the pixel under the strokes; the constraints is too few, and the result may be mainly influenced by the smooth term in the equation (2).

Energy function We use the well-know technique, graph cuts, to minimize the following energy function to segment the image into the editing region and the preserving region.

$$E(X) = \sum_{p \in I_s} E_c(p) E_p(p) + \alpha \sum_{(p,q) \in N_s} E_s(p,q). \quad (4)$$

Here X is a possible labeling on the image; $E_c(p)$ and $E_p(p)$ are color energy and positional energy. $E_c(p)$ is used to measure the conformity of the color of pixel p

to the foreground/background color model, and $E_p(p)$ is used to measure the ratio of spatial distances to the foreground/background strokes, and it can also estimate the importance of the spatial information of strokes. The data energy is $E_c(p)$ times $E_p(p)$, it means we consider two terms simultaneously to create a more robust result. $E_s(p,q)$ is a smooth term used to maintain the spatial coherence and tend to cut the image on the edge. N_s is a set which includes all neighbor pairs, (p,q), in the image.

Color term Like most previous works [16, 11, 18], we build 3D GMMs, M_F and M_B, with K components to describe the distribution of P_F and P_B. In practice, we use K-mean cluster to cluster the pixels into K groups to build GMMs. $E_c(p)$ is defined as follows.

$$E_c(p) = \frac{\log(\mathrm{D}(c_p, M_{x_p}))}{\log(\mathrm{D}(c_p, M_F)) + \log(\mathrm{D}(c_p, M_B))}, \quad (5)$$

$$\mathrm{D}(c, M_x) = \sum_{k=1}^{K} \omega_{xk} \mathrm{Pr}(c, G_{xk}).$$

The function $\mathrm{Pr}(\cdot)$ is a Gaussian probability distribution; ω_{xk} is mixture weighting coefficients; G_{xk} is the k-th components in M_x, and $x_p \in \{F, B\}$ is the mark of p for foreground or background. We use this term to estimate a pixel's color is closer to foreground or background, and the sum of energy of marking a pixel as foreground and background is 1.

Positional term Next, we introduce our new method to estimate the importance of spatial information. After analyzing some users' drawing behavior, we have two observations. One is that the pixel which is closer the foreground/background strokes is more probable to belong to the foreground/background. The other is that the pixel between foreground strokes and background strokes is in the region of the focus of attention, that is, the user requires the program to decide where the edge is. Furthermore, when the distances to the foreground strokes and the background strokes are close and the sum is small, the accurate cluster usually depends on the color. Otherwise, if the pixel is located in the same side of two kind of stroke and the sum of distances are large, there is high probability that the accurate cluster belongs to the closer stroke group. Briefly, when the sum of distances is small, the main decision strategy depends on the color value, and when the sum of distances is large, which kind of stroke is closer will become important. We use the following equations to model this statement.

$$E_p(p) = \frac{r - 0.5}{|r - 0.5|} \times 0.5 \times (2 \times |r - 0.5|)^n + 0.5, \quad (6)$$

$$n = a \exp(-b \frac{\mathrm{Dist}(p, P_F) + \mathrm{Dist}(p, P_B)}{\max(\mathrm{width}, \mathrm{height})}),$$

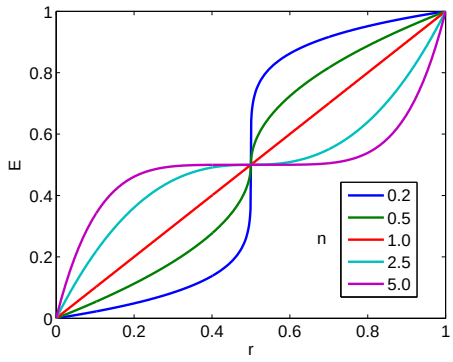


Fig. 3: This figures shows the tendency of the positional energy function, $E_p(\cdot)$, with parameter r and n. This figure is drawn by set a and b are 5 in Eq.(6). We can discover that when the n is larger, there is a broader flat region in the middle of figure. In the flat region, the energy is closer 0.5, which means the ratio of the spatial information is less important. Similarly, the steep line means the color information is less important relatively.

$$r = \frac{\mathrm{Dist}(p, P_{x_p})}{\mathrm{Dist}(p, P_F) + \mathrm{Dist}(p, P_B)},$$

$$\mathrm{Dist}(p, P) = \min_{p_i} \|p - p_i\|, \quad \forall p_i \in P,$$

here r is the ratio of distances of two kinds of stroke, and The value of n increase with the sum of distances to the foreground/background strokes. The positional energy becomes higher with the ratio r, n is the variable to decide how the importance of spatial information. When n become larger, the importance of spatial information is less. Moreover, since our data term is $E_c(p)$ times $E_p(p)$, and they both have a property that is their value keep a relationship of v and $1 - v$ when the x_p is different. Therefore, if one of them is 0.5, it means it is depends on the other to decide on which cost is higher, foreground or background. Therefore, when the n become larger; the energy is near 0.5 on most value; the importance of spatial information is less. We can refer the Fig. 3.

Smoothness term Next, we define the $E_s(p)$ as

$$E_s(p,q) = \begin{cases} 1/\|c_p - c_q\|^2 & , \text{if } x_p \neq x_q \\ 0 & , \text{if } x_p = x_q \end{cases}, \quad (7)$$

here $\|\cdot\|$ is L_2-norm distance in the $L^*a^*b^*$ color space. This term is a penalty term, That is, if you want to mark different mark on the pixel and its neighbor, you need to pay the additional energy which is inverse proportional to the color difference between the two pixels. When the similarity of the two pixels is high, you need to pay large energy to separate them. Finally, after the graph cut process, each $p \in I_s$ will get one mark $x_p \in \{F, B\}$.

3.4. Multiple color transfer

Our approach for altering color is to set a suitable constraint for each pixels, and then using an optimization process to build a final result. In addition, we work in the $l\alpha\beta$ color space, which is low correlation between channels, so we can perform the task separately in three channels [15]. Therefore, the following description is for one channel, and the other two is the same.

We have owned the user's requirements of how the regions should be. We first build the Gaussian Color Model pair, $G_{sj}(\mu_{sj}, \sigma_{sj})$ and $G_{tj}(\mu_{tj}, \sigma_{tj})$, using the pixels under the same strokes in the source and the target respectively. μ is the mean and σ is the standard deviation in the model G; j represents the different stroke. Besides, we also build the Gaussian Color Model for the preserving strokes. We can think of each stroke pair as a color transfer function. Therefore, we have the number of color transfer functions as many as $|\mathcal{F}|$. Now, we need to decide on how much ratio a pixel should be influenced from each color transfer function, and we set the constraints by accumulating the different influence of each function on the pixel. Therefore, we use the following equations to set the constraints.

$$\hat{f}(p) = \begin{cases} \sum_{j\in\mathcal{F}} \mathrm{Wt}(c_p, j)\frac{\sigma_{tj}}{\sigma_{sj}} + \sum_{j\in\mathcal{B}} \mathrm{Wt}(c_p, j) & , \text{if } x_p = F \\ 1 & , \text{if } x_p = B \end{cases} \tag{8}$$

$$\hat{u}(p) = \begin{cases} \sum_{j\in\mathcal{F}} \mathrm{Wt}(c_p, j)\mu_{sj} + \sum_{j\in\mathcal{B}} \mathrm{Wt}(c_p, j)c_p & , \text{if } x_p = F \\ c_p & , \text{if } x_p = B \end{cases} \tag{9}$$

$$\hat{v}(p) = \begin{cases} \sum_{j\in\mathcal{F}} \mathrm{Wt}(c_p, j)\mu_{tj} + \sum_{j\in\mathcal{B}} \mathrm{Wt}(c_p, j)c_p & , \text{if } x_p = F \\ c_p & , \text{if } x_p = B \end{cases} \tag{10}$$

$$\mathrm{Wt}(c, j) = \frac{\Pr(c, G_{sj})}{\sum_{i\in\mathcal{S}} \Pr(c, G_{si})}$$

We have mentioned that we treat the color transfer function Eq.(1) as three linear processes: shifting, scaling and then shifting. We use $\hat{f}(p)$, $\hat{u}(p)$ and $\hat{v}(p)$ to represent the parameter value in the three processes respectively; $\mathrm{Wt(c, j)}$ indicates how much ratio the color c should be influenced from the j-th color transfer function. In addition, if p is in the editing region, the weighted average of each functions is set to p as a constraint, but in the preserving region, we set itself color as shift parameter and 1 as scale one to preserve the original color value. The later term of the case $x_p = F$ is designed to prevent the clamp region error in the preserving region to produce unexpected change; especially, at edges and some fragmentary background between the foreground.

3.5. Global optimization

If we don't apply the global optimization, the result of multiple color transfer is showed in Fig. 4. There are some artifacts on edges, so we use the image-guided optimization technique to eliminate them. Before running the optimization, we need to set the weight of each constraints, $w(\mathbf{x}$ in Eq.(2). Actually, setting all of the weight as 1 is acceptable

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

Fig. 4: (a) The result before optimization. (b) The enlargement of the red square in (a). (c) The weight of constraints. (d) The difference between the source and the final result.

for most cases, or it can be set more reasonable by

$$w(p) = \begin{cases} \max_{j\in\mathcal{S}} \mathrm{Wt}(c_p, j) & , \text{if } x_p = F \\ \max_{j\in\mathcal{B}} \mathrm{Wt}(c_p, j) & , \text{if } x_p = B \end{cases}. \tag{11}$$

We find the maximal influence ratio of transfer functions as the weight of the constraint, and the sum of influence ratio sequence is normalized to 1. Therefore, if there is a influence ratio larger than others, it means the constraint of the pixel is close one of user's requirements (We can think of each user's requirement as a color transfer function). We set a larger weight for the constraint which matches a specific transfer function. However, in the preserving region, we just search the maximal influence ratio as weight in the set $\mathcal{B}$ for eliminating the clamp region error of making the editing region become the preserving region. Fig. 4 (c) shows the weight of the constraints. Then, we run the optimization process to get the final transfer functions, f ,u, and v, by setting the $\hat{f}$, $\hat{u}$ and $\hat{v}$ as f of the Eq.(2).

Finally, we renew the source photo via f, u and v. That is,

$$c_p = (c_p - u(p))f(p) + v(p), \quad \forall p \in I_s. \tag{12}$$

Here, each pixel is altered by different and suitable transfer parameters. Besides, the optimization doesn't influence the detail. Fig. 4 (d) shows the difference between the source and the final result. The background is preserved completely and no detail is lost in optimization.

4. RESULT

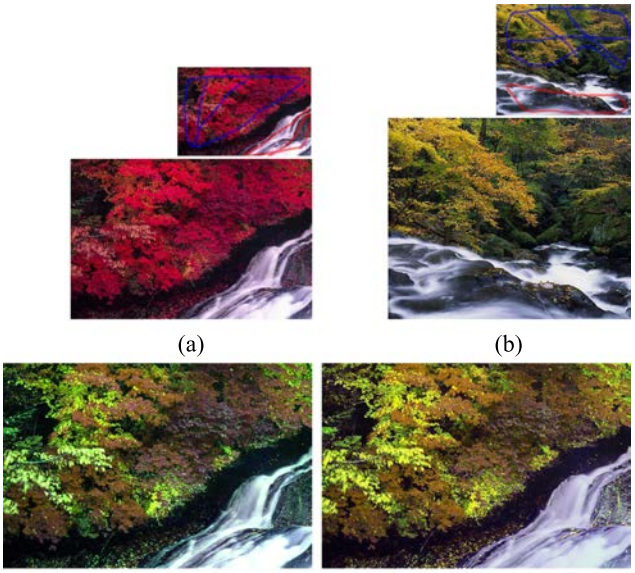
In this chapter, we display a variety application of our tool and discuss some implementation and application problems.

4.1. Application

Fig. 2 shows a common case that the photo is taken against the light. We found the other photographs taken in the same day to edit it. The cloth and the scarf in the source photo have approximate color because of underexposure. However, they can be recovered to original color by setting different transfer functions.

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

Fig. 5: Illustrated here is a suitable case to apply global color transfer. (a) The source image is an overexposed photo. (b) our result. (c) The target image is taken by other person. (d) Reinhard's [15] result.



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

Fig. 6: We compare our result with the Local Color Transfer [17] (a) The source image. (b) The target image. (c) Our result. (d) Local Color Transfer result.

Fig. 5 is a suitable case to apply global color transfer. We compare our result with Reinhard's [15] in this case. The source is an overexposed photo, and the target is taken by other people during the same trip. They can perform the task automatically, and we need to draw some strokes on the sky, mountain, lake and path. But their result embroiders light blue on the lake.

Fig. 6 compares our result with Tai's result of Local Color Transfer [17]. The source is taken with the red tree so the surface of the river reflects a little red light. They successfully separate the trees and the river when transferring colors. Despite the green trees, however, the river still includes a little red color. Our river looks more natural because it reflects light green from the river.

Fig. 7 shows a progressive editing process by using our tool. The source is taken under serious sunlight and in a high contrast scene. In this situation, it is hard to decide on the exposure and to take a photo that is satisfying for the whole region. Like this example, the background is over-

(a) (b) (c)

Fig. 7: Presented here is a progressive method by using our tool. (a) The source image is taken in a high contrast scene. (b) The result after altering background. (c) The result of editing face further. Upper is the strokes for editing. The red stroke successful preserves the background in (b).

exposed already, but the face is still a little underexposed. Users can use our tool, however, to easily create a better photo. To do so, we first chose a photo with a beautiful sea and coast in the same album, then perform the background editing. The face is still not clear enough, however, so we chose another photo to edit the skin color of the face. To exploit the preserving stroke, the sea and coast are not influenced in the further editing. Finally, we have a photo with living expression and a beautiful background.

Fig. 8 shows a case using multiple targets to edit one photograph. The source is taken at dusk and the sunlight only lightens the mountaintop. We found three photographs to edit this one for the sky, the mountain, and the forest respectively. We choose the photograph which was taken from the same subject to add some nature blue in the sky and found a high contrast mountaintop photo to accentuate the main object. Then, we added some green color at the bottom of the photo to rich the photo further.

Fig. 9 shows two photographs taken on the early evening. One photograph is taken by long exposure for recording the track of the car light, darker roads, and houses, but the far sea and sky become too bright because of overexposure. However, the photographer takes the other one to capture the mood of the sea in the evening. We can use our tool to combine the two objects of taking the photographs to create a better photo.

The right Fig. 1 shows another situation that is difficult to take a satisfying photo by a specific exposure. The source image is taken by the exposure which is suitable to capture the sea of clouds. For the overexposed sky, we chose another photo taken on the airplane at sunrise and transfer the

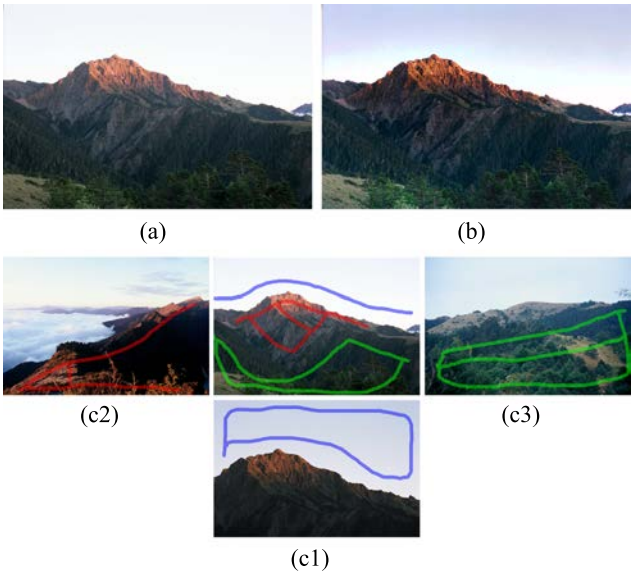


Fig. 8: This figure illustrates a result by using multiple target images to edit one photo. (a) The source image. (b) The result. (c) The target images and the corresponding strokes in the source and three targets. (c1) The target for the sky. (c2) The target for the mountain. (c3) The target for the forest.

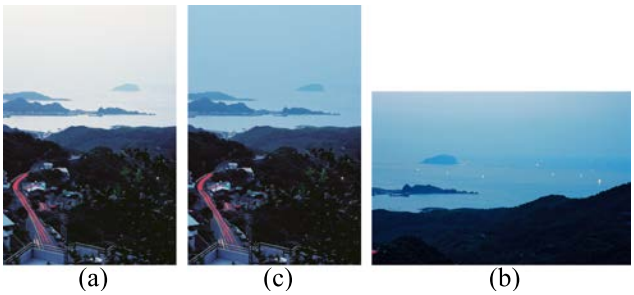


Fig. 9: (a) The source image is taken by long exposure for recording the track of the car. (b) The target image record the mood in the evening. (c) The result is created by combining the two objects.

color of the sky to enhance it.

Fig. 10 shows a result of editing the foreground which mixes with the background. We enhance the underexposed foreground, but the small background regions surrounded by the foreground are not influenced.

4.2. IMPLEMENTATION

Now, we discuss some implementation and application problems. The biggest problem of the optimization process is time length, and our approach need to run it nine times. It is impossible to wait for ten minutes for editing a photo. Typically, downsample and a better preconditioner are needed. There is an additional advantage from assigning the constraints for all pixels, that is to get a better preconditioner in solving a big sparse linear system.

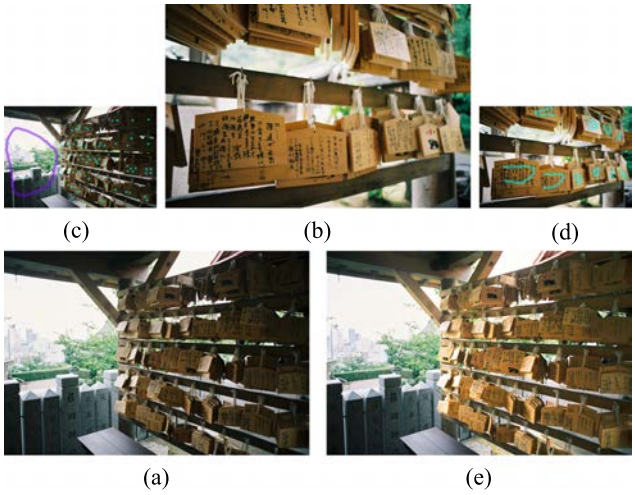


Fig. 10: (a) The source image. (b) The target image. (c) The strokes on the source. (d) The strokes on the target. (e) The result.

	Scarf	Coast	Sunrise
Convert color space	1.133	1.133	0.978
Graph cuts	2.984	3.094	3.454
Optimization	3.969	2.046	3.781
Upsampling	3.453	3.579	2.891
Other processes	1.148	0.992	0.830
Total	12.687	10.844	11.953

Table 1: Processing time in each stage, the unit is second.

Downsample is quite useful for speeding up the process, but upsample is another problem. Traditionally upsample approaches, such as Bicubic or Gaussian, may produce some artifacts around the hard edges. Fortunately, Kopf [10] developed a method, Joint Bilateral Upsampling, recently. This technique is suitable for our application. They joint the region information of the full resolution source image in the upsampling process. The result is smooth and edge-preserved, this property is fit the requirement of our application.

In our case, the solution calculated on eighth size in width and height is enough to get an acceptable approximate solution. One optimization process takes less than 1 second and To upsample the nine parameter functions takes around 4 seconds for a 0.3 mega-pixel image in our implementation. Table 1 lists the processing time in each stage of some results in this thesis. In some cases, if it is hard to get an accurate editing region in one step, and the result have unexpected effects because of the error editing region. Users may need to add some strokes to get a accurate edit-

ing region.

All results in this thesis were produced by following parameter setting: $\lambda = 0.2$, $\alpha = 1$ and $\varepsilon = 0.0001$ in optimization process Eq.(2); $\alpha = 16$ in graph cut Eq.(4); $K = 6$ in building G_F and G_B; $a = 5$ and $b = 5$ in Eq.(6).

5. CONCLUSION AND FUTURE WORK

In this thesis, we have described an example-based photo editing tool. Users can describe the requirements on an abstract level via an intuitive, stroke-based user interface. Then, we perform the editing by setting appropriate constraints and using the image-guided optimization process. This method can produce accurate results that match users' expectations. More importantly, the tool is very easy to learn; no photography or photo editing knowledge is required. Besides, we have presented many different applications using this tool. Along with convenience and efficiency, it is also a powerful way to complete creative tasks.

We have also introduced a new energy function for graph cut. According to where the pixel is located, the function can estimate which is more important: color or spatial information. In our framework, the tool can be used to clamp the region for editing; however, it can also produce more expected results in image cutout task. In future work, we would like to develop other editing function under the same scenario.

ACKNOWLEDGEMENT

This paper was partially supported by the National Science Council of Taiwan under NSC 95-2622-E-002-018 and also by the Excellent Research Projects of National Taiwan University under 95R0062-AE00-02.

REFERENCES

[1] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive digital photomontage. *International Conference on Computer Graphics and Interactive Techniques*, pages 294–302, 2004.

[2] A. Agrawal, R. Raskar, S. Nayar, and Y. Li. Removing photography artifacts using gradient projection and flash-exposure sampling. *Proceedings of ACM SIGGRAPH 2005*, 24(3):828–835, 2005.

[3] S. Bae, S. Paris, and F. Durand. Two-scale tone management for photographic look. *International Conference on Computer Graphics and Interactive Techniques*, pages 637–645, 2006.

[4] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004.

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[6] Y. Chang, S. Saito, and M. Nakajima. Example-Based Color Transformation of Image and Video Using Basic Color Categories. *Image Processing, IEEE Transactions on*, 16(2):329–336, 2007.

[7] Y. Chang, K. Uchikawa, and S. Saito. Example-based color stylization based on categorical perception. *Proceedings of the 1st Symposium on Applied perception in graphics and visualization*, pages 91–98, 2004.

[8] E. Eisemann and F. Durand. Flash photography enhancement via intrinsic relighting. *International Conference on Computer Graphics and Interactive Techniques*, pages 673–678, 2004.

[9] J. Jia, J. Sun, C. Tang, and H. Shum. Bayesian correction of image intensity with spatial consideration. *ECCV 2004*, pages 342–354, 2004.

[10] J. Kopf, M. Cohen, D. Lischinski, and M. Uyttendaele. Joint bilateral upsampling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, 26(3):to appear, 2007.

[11] Y. Li, J. Sun, and H. Shum. Video object cut and paste. *Proceedings of ACM SIGGRAPH 2005*, 24(3):595–600, 2005.

[12] Y. Li, J. Sun, C. Tang, and H. Shum. Lazy snapping. *ACM Transactions on Graphics (TOG)*, 23(3):303–308, 2004.

[13] D. Lischinski, Z. Farbman, M. Uyttendaele, and R. Szeliski. Interactive local adjustment of tonal values. *ACM Transactions on Graphics (TOG)*, 25(3):646–653, 2006.

[14] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama. Digital photography with flash and no-flash image pairs. *ACM Transactions on Graphics (TOG)*, 23(3):664–672, 2004.

[15] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley. Color transfer between images. *Computer Graphics and Applications, IEEE*, 21(5):34–41, 2001.

[16] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3):309–314, 2004.

[17] Y. Tai, J. Jia, and C. Tang. Local color transfer via probabilistic segmentation by expectation-maximization. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1, 2005.

[18] C. Wang, Q. Yang, M. Chen, X. Tang, and Z. Ye. Progressive cut. *Proceedings of the 14th annual ACM international conference on Multimedia*, pages 251–260, 2006.