HW1 KNN Matting

Image Manipulation Techniques and Visual Effects

Explanation of codes

Parameters

TEST_K

(list) Defaults to [10]

The values inside this list are the count(s) of nearest neighbor(s) that will be found.

IMG_DIR

(str) Defaults to '../img'

This is the path where the input images, trimaps, and backgrounds are stored. Three directories, image, trimap, and background should belong to this directory.

OUT_DIR

(str) Defaults to '../result'

This is the path where the output images will be stored.

ADJ_BG_INTENSITY

(bool) Defaults to True

If set to True, the intensity of the given background image will be adjusted to mimic that of the given foreground image.

Functions

---

knn_matting(K, img, trimap, my_lambda=100)

Returns the optimal alpha

Parameters:
K (int): The count(s) of nearest neighbor(s) that will be found
img (ndarray): The foreground image

trimap (ndarray): The trimap of the foreground image
my_lambda (int, optional): Defaults to 100. The parameter for solving the linear system

Returns:
ndarray: The optimal alpha

This function calculates the optimal alpha by using KNN mating method.
There are mainly 7 sections in this function. The first section prepares for values that will be needed. The second section calculate the feature vector X, where X(i) = (R, G, B, x, y)(i). The third section takes the feature vector X and calculate K nearest neighbors by using sklearn kit. The forth section calculate the kernel function k, where k(i, j) = 1 – |X(i) – X(j)| / C. Note that C = c + 2 since (x, y)s are appended to the feature vector. The fifth section uses spicy kit and takes the kernel function k to calculate the affinity matrix A. The sixth section are preparations for solving the objective function. L, M, and v are calculated here. Finally, the seventh section solves for the linear system, and the optimal alpha will be returned. Note that the exception will be executed if no exact solution exists.

---

adjust_intensity(img, alpha, img_ref)

Returns the intensity-tuned image

Parameters:
img (ndarray): The base image
alpha (ndarray): The mask of img_ref
img_ref (ndarray): The image whose intensity will be referenced and mimicked

Returns:
ndarray: The intensity-tuned image

This function tuned the base image's intensity in order to share a similar lighting condition between the base image and the referenced image. Note that alpha act like a mask of the referenced image. Only when the alpha value of a pixel of the referenced image is greater than 0.5, the intensity of that pixel will be considered. The base image will be multiply by an intensity scale, so that both base and masked references image have the same intensity. Pixel values greater than 255 of tuned image will be replaced by 255. A new image will be returned, and the given base and referenced image will not be affected.
The results of this technic will be shown in the following part of this report.

main()

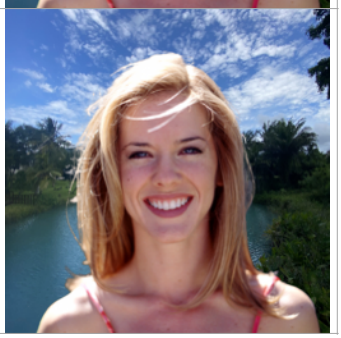
This is the main logic of release.py. A list of image names defines which images to be tested. A loop will go through all values in TEST_K. When processing each image, image and trimap are loaded at first. Then, alpha can be calculated by calling knn_matting(). Alpha is then transformed from shape (h, w) to (h, w, 3) by using numpy.stack(). After that, background is loaded, and if no background image is found for the given image, a general background image will be used. Background will be resized to fit the size of foreground image and intensity adjusted if ADJ_BG_INTENSITY is set to true. Finally, the composed image is saved after the composition is performed.

Experiments and Results

Tuning K

K plays an important role in KNN matting method, so I tried 5 values from 1 to 20. Below are the results on testing images.

K	bear	gandalf	woman
1			
5			
10			

K	bear	gandalf	woman
15			
20			

As we can see, the edges of the foreground image were better detected as K increased. When K is 1, the furry edges of the objects cannot be retained. However, after K is greater than 10, I can hardly tell the differences between these images.

Adjusting intensity

After performing the above test, I realized that the bear image results are those seem to be artifacts. I believed that the main reason is that the background and foreground have different lighting conditions. As the result, the composed image looks unreasonable and strange due to unnatural optics.

I decided to adjust the intensity of the background images to mimic those of the foreground images by calling the adjust_intensity() function. Below are the results of the testing images as well as an additional image. K is set to 10.

	Not adjusted	Adjusted
bear		

	Not adjusted	Adjusted
gandalf		
woman		
GT10		

The bear image and the GT10 image have significant differences after adjusting the lighting condition. The bear image background is tuned lighter than before, and the GT10 image background is dimmed. As a consequence, the result images can be composed by images that shared the same intensity. I personally think that the adjusted backgrounds lead to results that seem lots more natural than before.
Note that the foreground images are not changed. Only the backgrounds are tuned.

Bonus

Working on more images

I downloaded 27 addition images from http://www.alphamatting.com/datasets.php to perform the KNN matting. Only 2 additional images are listed below due to the layout.

K	GT10	GT26
1		
5		
10		
15		
20		

Adjusting intensity

The technics and results are mentioned in the above.

Shooting background images

Instead of downloading, all background images are shot on my phone by myself.