Computer Graphics

Course project

UIN: 525007137
Ting-Jui Chang

1. Title:

The implementation of photomosaics with K-means clustering

2. Summary:

a. Problem Description:

To implement image mosaics, two things need to be prepared : a set of images which would be taken as “tiles”, and a target image which we are going to present. Roughly talking the process of producing a photo mosaics, first make a grid over the target image, then for each cell on the grid, calculate its average color and go to the set of reference images to find the nearest image with the same average color of the cell and substitute it in that cell. For the step of searching the nearest image, original way has to go through all set of images, which would take a lot of effort. My idea is that first separate all reference images into several cluster, and each time a cell comes in, I find out which cluster this cell belongs to and return the most representative image of that cluster as the tile for that cell.

b. Importance of problem:

Like what is mentioned above, the traditional way of searching the nearest image for a cell may take too much effort if the set of reference images is quite large. My idea is based on a thinking that if the set of reference images actually consists of several cluster and the difference within a cluster is small while the difference between clusters is large, there is no need to search through all image set. Therefore, by using the representative image of each cluster instead of all images of that cluster, we can accelerate the searching process. But if the reference images are not distributed like the way mentioned above, then the effect of applying this approach may not be very obvious.

3. Previous Work:

When trying to find out the best fit tile for each tile in the target image, there

are three mostly used color metrics for computing the color distance between two different tiles, one is Euclidean distance in terms of RGB average colors, the second one is the linear metric, and the last one is the Riemersma metric, where the Riemiersma metric is considered as the one with the best effect due to its closeness to human eyes detection. In this project, the first metric is chosen because of the simplification. About the part of searching the nearest image for a cell, to speed up this process, some previous works usually use kd-tree structure to decrease the search time (logN, N is the number of reference images). In this project, the number N is actually decreased by selecting some most representative image within the whole reference image set. Figure 1 shows the equations of the metrics mentioned above.

(a)

$$\Delta=\sqrt{(R_1-R_2)^2+(G_1-G_2)^2+(B_1-B_2)^2}$$

(b)

$$ColorNumber=(R\cdot 65536)+(G\cdot 256)+(B\cdot 1)$$

(c)

$$\Delta=\sqrt{(2+\frac{\bar{r}}{256})(\Delta R)^2+4(\Delta G)^2+(2+\frac{255-\bar{r}}{256})(\Delta B)^2}$$

where $\bar{r}=\frac{R_1+R_2}{2}$, $\Delta R=R_1-R_2$, $\Delta G=G_1-G_2$,

and $\Delta B=B_1-B_2$

Fig. 1 The formulas of three distance measurement metric

4. Description of Work:

In this project, the work can be roughly separated into two parts: the first part is doing the image clustering and finding out the representative image for each image cluster through the reference image distribution; the second part is mainly for the part of constructing a mosaic image based on the sample image distribution.

a. First Part:

For doing the clustering job, I applied the most classic unsupervised clustering technique, K-means, which tries to separate the given data set into K different groups. In order to compare the final effects of image mosaic under different conditions, I tried five different K values: 10, 30, 50, 70, 90.

b. Second Part:

After getting the representative images from the first process, we can start to generate the photo mosaic for the target image. To get a better result (higher resolution), we enlarge each side of the target image for eight times, and we

set the size of tile as a 50 pixels by 50 pixels image (To avoid distortion, we first crop each reference image to make sure it is a square, then resize it as a 50 by 50 image. The enlarged target image is also cropped to make sure that each side of it is a multiple of 50 so that it can be filled up with tiles). About finding the nearest reference image for each cell of the target image, to get finer results, instead of using the average RGB color, I use the entire tile to compute the color distance directly (using a vector of size [50, 50, 3] rather than [1, 1, 3]).

c. Challenges in The Project:

When doing the k-means part, since the resolution of the image in the dataset is too high, it takes a lot of time for clustering the dataset. To accelerate the process of k-means, I use a compromise idea, which is shrinking the original images in the dataset before doing the clustering. For the part of building the image mosaic, since it also contains a heavy part of computing the distance (to find the nearest neighbor), I use the multiprocessing package of Python to make the program run faster.

5. Result:

In this project, five different K values (10, 30, 50, 70, 90) are tried for K-means clustering, and we constructed two series of images for two different target image, one has a smaller color scope (closer to blue color region) and the other one has a wider color scope. For mosaic images within each series, we do the comparison in terms of the time cost and the final effect. Note that the reference image distribution consists of 645 images in this project.

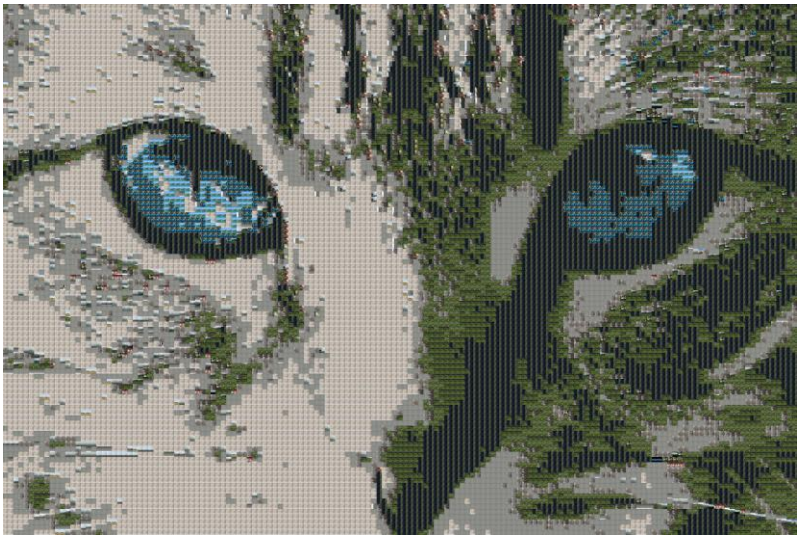
a. Image with the smaller color scope:

Original Image:

K = 10:

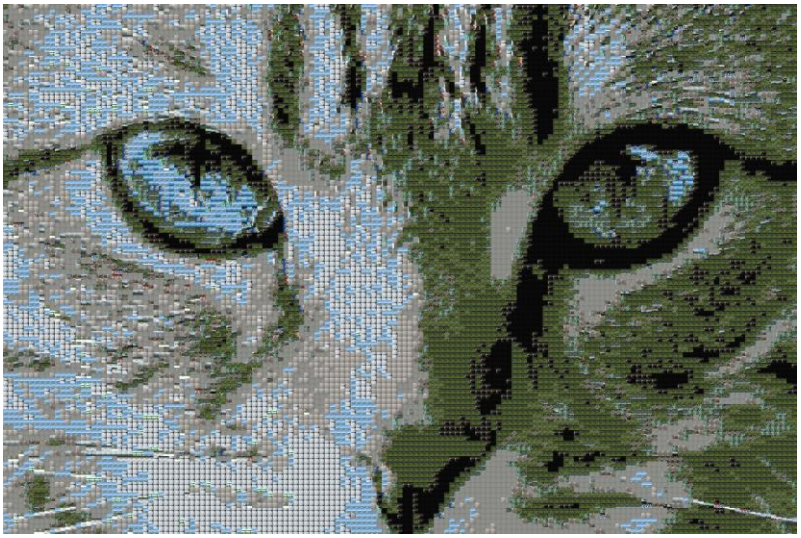
K = 30:

K = 50:



K = 70:

K = 90:



Using the all dataset to construct Image mosaic:

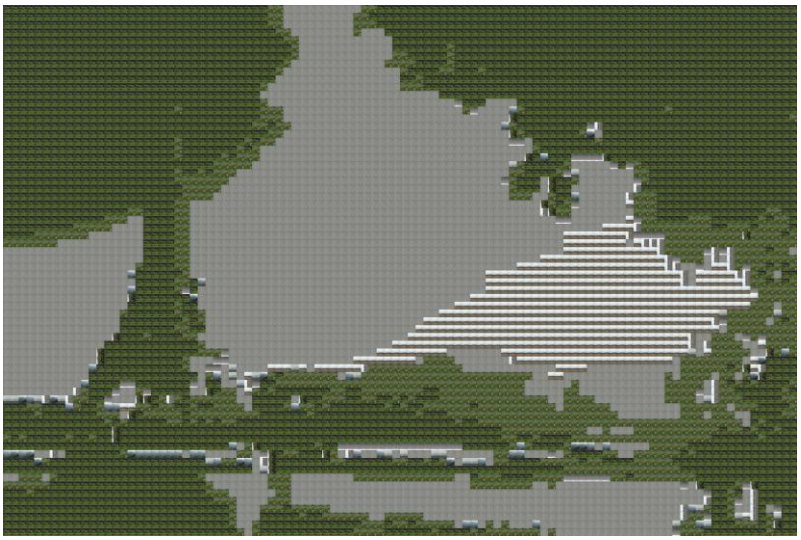
Time Cost:

K = 10	K = 30	K = 50	K = 70	K = 90	All pics
4 secs	6 secs	9 secs	10 secs	14 secs	90 secs

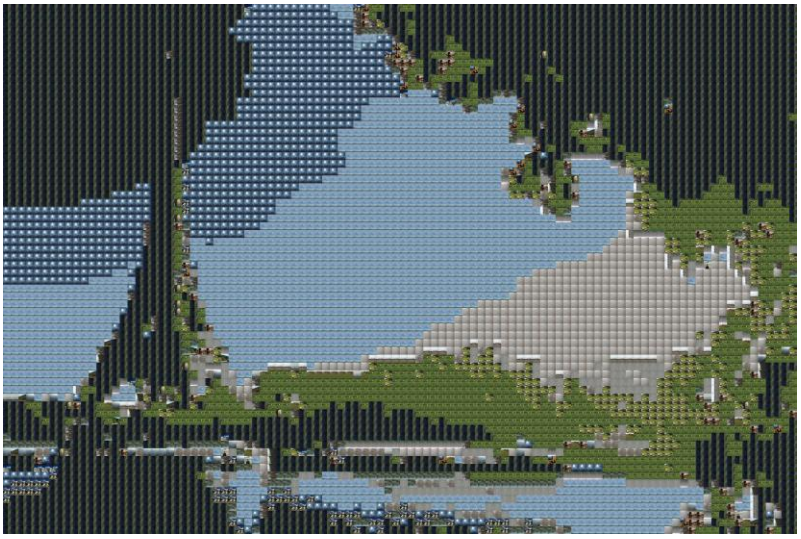
b. Image with the larger color scope:

Original Image:

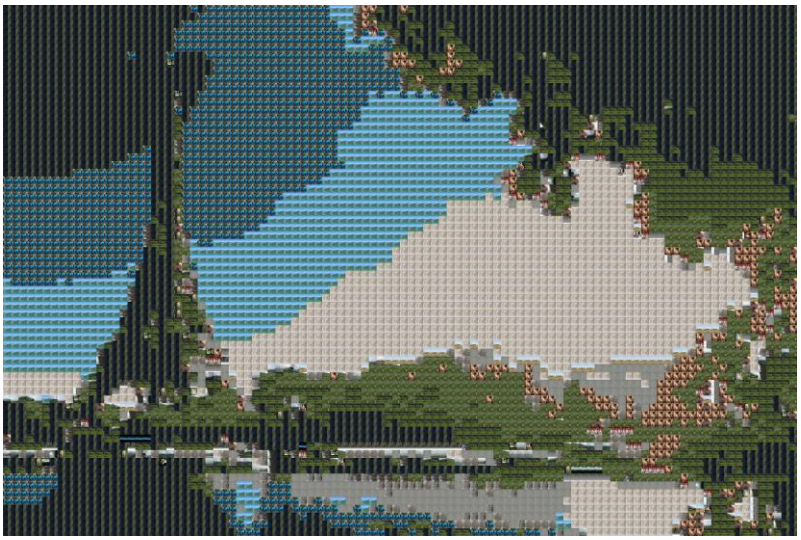
K = 10:



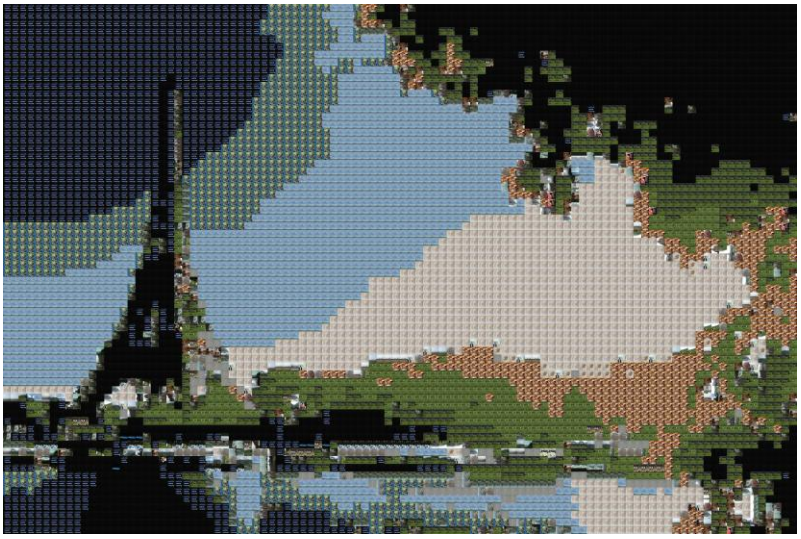
K = 30:



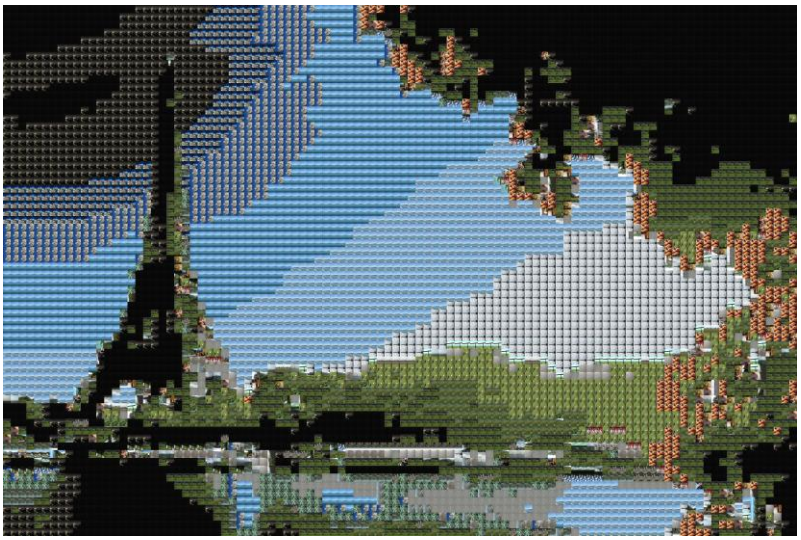
K = 50:



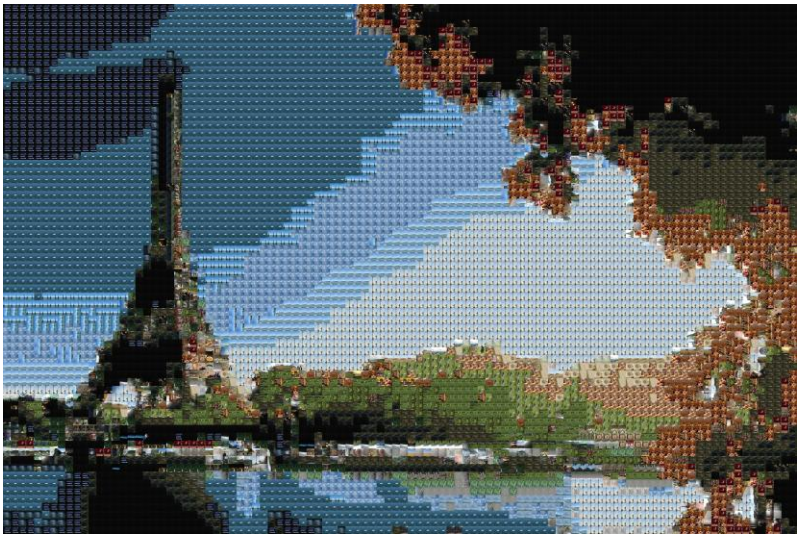
K = 70:



K = 90:



Using the all dataset to construct Image mosaic:



Time Cost:

K = 10	K = 30	K = 50	K = 70	K = 90	All pics
2 secs	3 secs	5 secs	6 secs	7 secs	43 secs

6. Analysis:

a. Meeting goals & Discussion:

Most of the goals in the proposal are completed. Based on the resulted images, we can actually find out that if we use enough number of cluster center to represent the reference image dataset, with less time cost, we can get an image mosaic with almost the same effect as the image mosaic generated by using the whole dataset, but how many representative image we should use really depends on the distribution of the image dataset and the size of the tile we set (In order to easier fit the target image, we would resize each reference image into a square-sized tile. If the tile size is smaller, the resized results of two different reference images may look more similar, which means that the redundancy is increased and the needed amount of representative image is decreased).

b. Future Work:

About the future work, there are two directions for improvement:

1) The clustering algorithm to apply:

K-means clustering has the strong assumption that the data distribution is the mixture of several isotropic Gaussian distributions, however, sometimes the real dataset we got may not fulfill this assumption, so using

k-means technique may not cluster the dataset properly. There are some other unsupervised machine learning technique for clustering with more relaxed assumption. For example, we can assume that the data set distribution is the mixture of general Gaussian distributions, then apply Expectation-maximization algorithms to find out the center of each Gaussian distribution and take the centers as the representative image.

2) The metric used for determining the similarity between two tiles:

   Even though the most frequently used metric for computing the similarity between images is using the Euclidean distance, I think we can actually take more features into account to get a better effect, for example, when comparing the similarity between two images, besides computing their Euclidean distance, we can also compute the orientation of the image and take it into consideration for comparing the similarity. My intuition is that by adding this extra feature, the final result can describe the information of the edges of the original image better.

Reference:

[1] http://www.drdobbs.com/understanding-photomosaics/184404848?pgno=1

[2] http://kodu.ut.ee/~b04866/poster.pdf

[3]http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.37.4998&rep=rep1&type=pdf

[4] http://blog.wolfram.com/2008/05/02/making-photo-mosaics/

[5] https://github.com/codebox/mosaic

[6] https://github.com/DavideA/photomosaic