Pictures at an Exhibition

Stephanie Kwan and Karen Zhu
Department of Electrical Engineering
Stanford University, Stanford, CA 94305
Email: {skwan1, kyzhu}@stanford.edu

***Abstract*—An image processing algorithm is designed and implemented to identify pictures at an exhibition. The algorithm first performs segmentation to obtain the painting in the input image by looking at both the luminance and chrominance components of the input image. The luminance ratios and color histograms are computed to identify the painting according to a set of pre-calculated statistics. The algorithm is able to identify images taken from different angle and different illumination conditions. The algorithm correctly identifies all images in a given training set. The average speed of the algorithm is 20-30 seconds per input image.**

I. Introduction

ADVANCES in mobile imaging technology has enabled many interesting applications of hand-held mobile devices. One example is "mobile augmented reality". By aiming a hand-held device at an object of interest, the user can get information regarding that object from a remote database. This application allows hand-held devices to serve as electronic museum guides. For example, the user can use a camera phone to take a snapshot of a painting of interest, and listen to commentaries about the painting. One of the critical components of such application is the image processing algorithm used to recognize the painting from the picture taken. How the snapshots of the paintings look depend on camera angle, illumination and many other factors. Therefore, a combination of different image processing techniques is required to guarantee successful recognition of the paintings.

In this project, we developed a fast and robust algorithm for recognizing paintings in the European Gallery of the Cantor Arts Center. The input of our system is an RGB image of a painting from the European Gallery, and the output is the title of the painting. We are also given a set of training images, so that we can train our algorithm to set up appropriate parameters and threshold before actual testing. We also tested the segmentation part of our algorithm with other test images that we generated.

II. Image Processing Algorithm

Our algorithm consists of three main stages. A) Segmentation of input RGB image. B) Extraction of luminance and color information of the painting. C) Identification of Painting. We first segment the input JPEG image to obtain the part that only contains the painting. We then divide the luminance component of the input image into four quadrants, and obtain statistics of the spatial distribution of the luminance component of the input image. We also computed the color histograms of the R, G and B color components. Finally, from the information obtained from the image, namely the height to width ratio, luminance statistics and color histograms, we find the best match from the pre-calculated painting statistics in our database, and identify the input painting to be the best matched painting in the database.

A. Segmentation

The objective of the segmentation process is to identify the part of the picture that contains the painting. We try to create a binary mask of the input image, where the painting region is labeled as 1, and the other region labeled as 0.

Among the paintings that the project tries to identify, the painting frames are black, brown or golden. Black frames have luminance values lower than the overall mean of the luminance component of the image, while brown and golden frames have Cb values lower than the overall mean of the Cb component, but Cr values higher than the overall mean of the Cr component of the image. Using these properties, we transform the RGB input image to the YCrCb domain, and binarize the Y, Cr and Cb components of the image, setting potential frame regions to 1, and non-frame regions to 0. From these three resulting binary images, we create two temporary binary masks, one from the binarized luminance image (LuMask), and the other from combining the Cr and Cb binary images (CrCbMask). The choice of one mask over the other is highly dependent on the image under test. Sometimes, the luminance component identifies the painting region more accurately, while other times, the Cr and Cb components identify the painting region better. Fig. 2 shows the LuMask and Fig. 3 shows the CrCbMask for one of the training images.

To make sure the entire painting, but not only the frame are labeled as 1, we fill up holes in connected regions of 1s in the two masks. Then, we label the connected regions in the masks, and identify the largest region. For each of the masks, base on the criteria that the majority of the identified region has to be labeled as 1 in the mask, we identify the largest region that fulfills the criteria to be the painting region. We then set the pixels in the respective identified painting regions to 1, and other region to 0.

As the next step, we choose to use one of the masks by

determining whether LuMask or CrCbMask gives a better estimate of the painting region base on criterions listed below:

1) We know that the painting has to be located at the center of the input image. Therefore, if one of the masks does not contain the center pixel, then the other mask is chosen.

2) Since the painting cannot reside at the corner of the input image, if at least one of the corners of one of the masks is labeled as part of the painting, the other mask is used.

3) If the height of LuMask is shorter than CrCbMask, we choose the LuMask. The rational behind is that the CrCbMask is usually shorter than the LuMask since the LuMask sometimes includes shadows. However, if the CrCbMask is longer, this means that it captured a lot of unnecessary parts either at the top or bottom of the painting, and therefore we do not want to use this mask.

4) If the overlapping area of the identified painting regions in the two masks is greater than 90% of the total identified painting region, then we choose the mask that gives a smaller paining region. Since the two masks are very similar, and it is likely that the smaller mask includes less unwanted regions besides the painting. However, if the overlapping area is less than 90%, then most likely the smaller mask failed to capture the entire painting. Thus, the larger mask is chosen.

After choosing the mask, the mask is further refined to eliminate unwanted regions at the sides of the identified painting area. Fig. 4 shows the final mask of one of our training images. The height to width ratio (HWR) of the painting is also recorded at the end of this process.

Finally, we overlay the mask with the original RGB image, and extract the segmented painting from the RGB image. From this point onwards, we only work with the segmented part of the image. This reduces the size of the image we have to work with, and speeds up the processing time required. Fig. 5 shows an example of a segmented painting.

Fig. 1. Original image of ‘The Painter’.

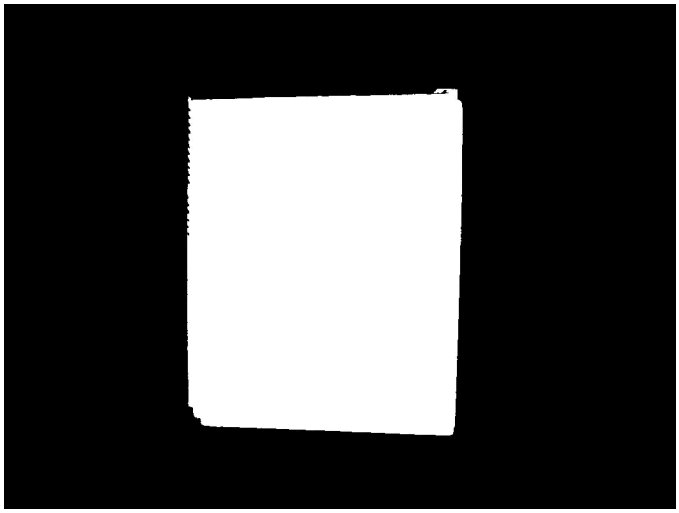

Fig. 2. LuMask of ‘The Painter’.

Fig. 3. CrCbMask of ‘The Painter’.

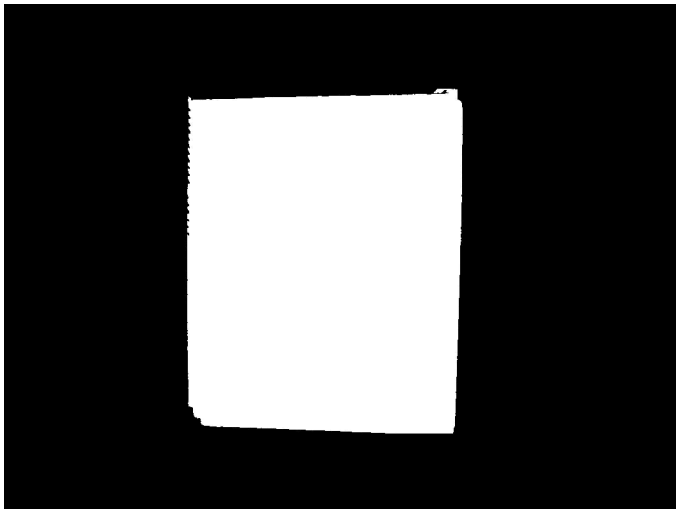

Fig. 4. Final mask of ‘The Painter’.

Fig. 5. Final segmented image of 'The Painter'

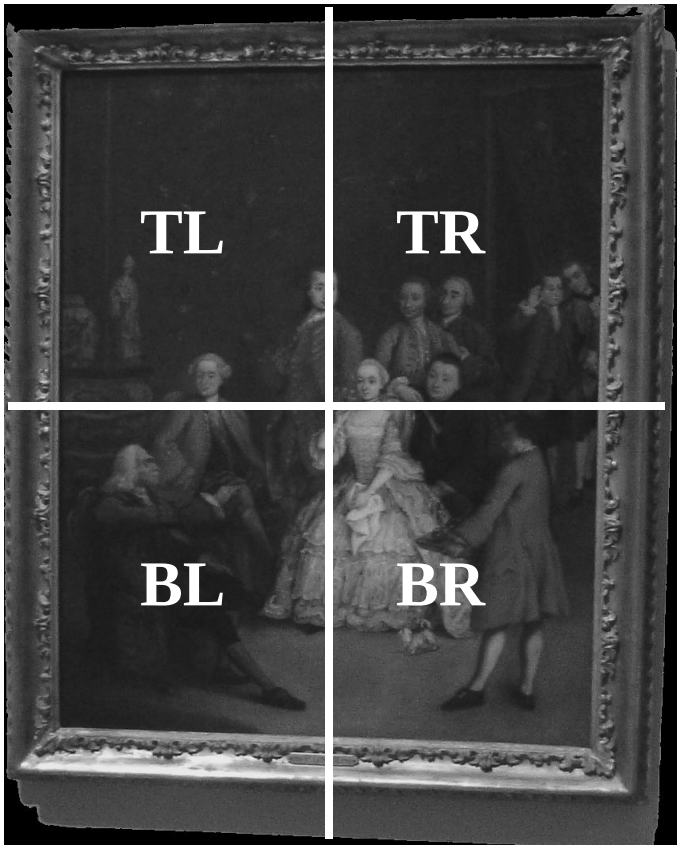

Fig. 6. Illustration of the four quadrants for luminance ratio calculation.

B. Extraction of Luminance and Color Information

The goal of extracting luminance and color information from the painting is to collect reliable statistics about the painting, so that it can be compared to the pre-calculated statistics, and identify the input painting. The main reason for extracting luminance and color information from the painting is that in a museum setting, lighting is controlled. Therefore, by strategically processing luminance and color statistics, variation of these statistics of the same painting due to factors such as different camera angles and distance from painting will be smaller than variation between different paintings. Hence we will be able to identify the correct painting using these statistics.

B1. Luminance Ratio

We separate the luminance component of the segmented image into four quadrants: top left (TL), top right (TR), bottom left (BL), and bottom right (BR) as shown in Fig.6. Then, we calculate six luminance ratios of the luminance image:

1) mean(TL+TR) /mean (BL+BR) (TBR)
2) mean(TL + BL)/mean(TR + LR) (LRR)
3) mean(TL)/mean(entire image) (Q1R)
4) mean(TR)/mean(entire image) (Q2R)
5) mean(BL)/mean(entire image) (Q3R)
6) mean(BR)/mean(entire image) (Q4R)

Although the absolute luminance values of the overall segmented image may vary significantly depending on how the picture was taken, provided that the picture is not taken at an acute angle, the six ratios we collect will remain similar since lighting is controlled. For instance, the TBR of a painting that has a bright top and dark bottom will always be greater than one no matter how the picture is taken. These ratios give spatial information about the painting.

B2. Color Histogram

Color in an image convey a lot of information and is often used to aid computer vision applications. The color histogram method has been first proposed by Swain and Ballard in [1]. Objects can be identified by matching the color histograms of each of the color channels between different images. The drawback of this approach is that any slight variation in lighting condition can significantly influence the shape of the color histogram and weaken its robustness [2].

The Comprehensive Color Image Normalization method is therefore suggested by Finlayson et al. in [3] to discount for lighting geometry and illuminant color together at the same time. This method utilizes an iterative approach, normalizing

lighting geometry in one step and illuminant color in another, until each color channel stabilizes in its values.

$$R(I)_{i,j} = \frac{I_{i,j}}{\sum_{k=1}^{3} I_{i,k}} \quad (1)$$

$$C(I)_{i,j} = \frac{\frac{N}{3} I_{i,j}}{\sum_{k=0}^{N} I_{k,j}} \quad (2)$$

In equation (1) and (2), I is the N*3 image matrix of red, green, and blue values, with each color channel having N values. R(I) is the row-normalization step to discount for lighting geometry. C(I) is the column-normalization step to discount for illuminant color. It has been proven that this process always converges, the convergent image is unique and the convergence is very fast, in only four to five iterations. [3]

In our implementation, we first apply Comprehensive Color Normalization to the segmented images by performing five iterations of a two step process. In step 1, we normalize each pixel value for lighting geometry by applying equation (1). In step 2, we normalize each pixel value for illuminant color by applying equation (2). This normalization process put the pixel values in the range of 0 to 1.

Step1:

$$(r_i, g_i, b_i) = \left(\frac{r_i}{r_i + g_i + b_i}, \frac{g_i}{r_i + g_i + b_i}, \frac{b_i}{r_i + g_i + b_i} \right)$$

Step 2:

$$(r_i, g_i, b_i) = \left(\frac{r_i}{\frac{3}{N}\sum_{i=1}^{N} r_i}, \frac{g_i}{\frac{3}{N}\sum_{i=1}^{N} g_i}, \frac{b_i}{\frac{3}{N}\sum_{i=1}^{N} b_i} \right)$$

$$(r_i, g_i, b_i) = \left(\frac{r_i}{3 \times mean(R)}, \frac{g_i}{3 \times mean(G)}, \frac{b_i}{3 \times mean(B)} \right)$$

After normalization, we compute the color histogram for each of the color channels. The histograms are generated by using 256 bins of equal width in the range of 0 to 1. Finally, we normalize the histogram by dividing each bin in the histogram with the total number of pixels in the segmented image to account for different sizes of images. Three color histograms are saved for each segmented image for identification of the painting.

C. Identification of Painting

After collecting all the necessary statistics from the previous stage, we proceed to comparing these data with statistics we pre-calculated about the paintings we have to identify. In order to reduce the number of false positives, the algorithm first attempts to eliminate as many improbable paintings as possible before choosing the most probable painting in the last stage of the identification process.

C1. Pre-Calculation of Reference Luminance Ratios and Color Histograms from Training Set

Reference luminance ratios and color histograms for each painting are calculated and stored in our database. We compare the luminance ratios and color histograms of our input test image to these reference data to determine which painting the input image belongs to.

To get the six reference luminance ratios and three reference color histograms for each painting, first we run our segmentation algorithm on each of the 99 images in the training set, and calculate the luminance ratios and color histograms for each of these images. Then, we group the images according to the painting they belong to (ie. 33 groups, each with three images), and take the average luminance ratios and color histograms of the three images in the group to be the reference luminance ratios and histograms for the painting.

C2. Pre-Calculation of Luminance Ratios Thresholds for Identifying Paintings

To eliminate paintings that are definite mismatches with our input test image in the luminance ratio test, we have to set thresholds that represent the maximum variations allowed for luminance ratios between pictures of the same painting. In total, we need six threshold values.

To find the thresholds, we first divide our training set (99 images) into three groups, where each contains 33 images, with one image from each painting. We then do three iterations of training and testing as described below.

In each iteration, we use two groups of images as our training set, and the last group as our test set. The three iterations are

1. Training = Group 1,2, Testing = Group 3
2. Training = Group 1,3, Testing = Group 2
3. Training = Group 2,3, Testing = Group 1

For each iteration, we first create our reference luminance ratios $R_{LuminanceRatio}$ using the same method in C1. However, instead of taking the average of three images, we only take the average of the two images in the training set for this current iteration. Then, we compare the luminance ratios of image (i) in our test set with the corresponding reference statistics $R_{LuminanceRatio}(i)$ of the same painting we obtained in this iteration using the equations (3),

$$LuRatioVariation(i,k) = \left| \frac{Test_{LuminanceRatio}(i,k) - R_{LuminanceRatio}(i,k)}{R_{LuminanceRatio}(i,k)} \right| \quad (3)$$

where $i \in \{1...33\}$, representing the painting number, and $k \in \{1...6\}$, representing one of the six luminance ratios.

We then record the maximum variation for each luminance ratio among the 33 paintings in each iteration.

After the three iterations, we get three sets of maximum luminance ratios variations. For each luminance ratio, we then take the maximum value of the three sets, further increase these maximum values by a small amount, and use these numbers as the threshold values that we need.

The reason for separating our initial training set into three groups and obtain thresholds as described above is two folded.

First, if we use all three images of the same painting to set our reference, and obtain the thresholds by calculating the maximum variations of the same three images with the reference, these thresholds are not likely to be robust. The statistics of our 'test images' are part of the reference. By separating the training set and test set, statistics of the reference and the test data are independent, thus the thresholds set using this method are likely to be more robust.

Second, variations observed by averaging two pictures and testing on the third picture are likely to be larger than variations observed by averaging three pictures and testing with a forth one. Therefore, by setting the thresholds to be the maximum variations from the reference found by using two images per painting, the thresholds should be large enough to incorporate variations from forming the reference using three images for each painting, and testing with a forth one.

C3. Identifying the Painting

As a first step of eliminating unlikely paintings, we look at the height to width ratio (HWR) of the input painting. If the HWR of our input painting is less than 1, then we eliminate all the paintings in the training set that have HWR greater than 1. On the other hand, if the HWR is greater than 1.8, then we eliminate all the paintings that have HWR less than 1. Unlike the previous case where the HWR threshold is 1, we set the HWR threshold to be 1.8 in this case to avoid false positive conclusions. When a picture of a painting is taken at an angle, it is possible that the height of the painting in the segmented picture is greater then its width, although the width of the painting is actually greater than its height in reality. Under normal viewing conditions of paintings in a museum, this distortion caused by viewing angle only happens when the length of height is relatively close to the length of width. Therefore, if HWR is greater than 1.8, we can be sure that the painting under test must have its height greater than its width.

In the second step of the identification algorithm, we calculate the variation of the 6 luminance ratios of the input painting from the pre-calculated reference luminance ratios of the 33 paintings. Of the 33 possible candidate paintings, we only perform the six luminance ratios variation calculation if they have not been eliminated in the first step. The luminance ratios variations are calculated using equation (3), and the results are compared to the pre-set threshold. We eliminate the painting if more than one of these luminance ratios exceeds their corresponding thresholds. For paintings that are not eliminated, we store the mean of the resulting luminance ratios variations, and call it LuVar. LuVar is computed using equation (4),

$$LuVar(i) = mean(LuRatioVariation(i,k) - Threshold(k)) \quad (4)$$

where $i \in \{1...33\}$, representing the reference painting number, and $k \in \{1...6\}$, representing one of the 6 luminance ratios.

The larger the LuVar is, the better match the input test image is to the reference painting. Note that we only include LuRatioVariation(i,k) that is smaller than the threshold. So while there are six luminance ratios, if one of the ratios is above the threshold, then we only take the average of the other five luminance ratio variations.

In the third step, we compare the color histograms of the input painting with the reference color histograms of the left over candidate paintings. The variation of the color histograms of the input painting from the reference painting is calculated using equation (5),

$$\begin{aligned} ColorHistogramVariation(i) = \\ & mean\left|Test_{RED\_histogram}(i) - R_{RED\_histogram}(i)\right| + \\ & mean\left|Test_{GREEN\_histogram}(i) - R_{GREEN\_histogram}(i)\right| + \\ & mean\left|Test_{BLUE\_histogram}(i) - R_{BLUE\_histogram}(i)\right| \end{aligned} \quad (5)$$

where $i \in \{1...33\}$, representing the reference painting number. The smaller the ColorHistogramVariation, the better match the input test image is to the reference painting.

Finally, from the remaining reference painting choices, we form a comparison metric by dividing LuVar values with ColorHistograVariation values.

$$ComparisonMetric(i) = \frac{LuVar(i)}{ColorHistogramVariation(i)} \quad (6)$$

We identify the input test image as the reference painting that gives the largest ComparisonMetric value.

III. Results

We ran our algorithm on the given 99 training images, and all of them were recognized accurately. The average processing time for each input image is 20 - 30 seconds.

IV. Conclusion

We have designed and implemented a fast and robust algorithm for gallery painting identification. The robustness is achieved by a combination of different image processing methods. By taking into account both luminance and chrominance information of the input image in creating the mask, segmentation can successfully eliminate unwanted image areas. Height to width ratio, luminance ratio, and color histogram each focuses on a different characteristic of a particular painting. Together they extract critical information from the paintings, and differentiate them effectively. The speed of our algorithm is due to its simplicity. Each of the image processing method is very simple and requires relatively little computation power.

V. Division of work

The following is a rough breakdown of work for the project:

Stephanie:
- Segmentation Algorithm
- Luminance Ratio calculation
- Identification of Painting
- Report

Karen:
- Color Histogram calculation
- Generate new test set and testing
- Report

VI. References

[1] M. J. Swain and D. H. Ballard, “Color Indexing”, International Journal of Computer Vision, 7(1):11 – 32, 1991.

[2] B. Schiele and J. L. Crowley, “Recognition without Correspondence using Multidimensional Receptive Field Histograms”, International Journal of Computer Vision, 36(1): 31 – 35, 2000.

[3] G. D. Finlayson, B. Schiele and L. J. Crowley, “Comprehensive Colour Image Normalization”, ECCV’98 Fifth European Conference on Computer Vision, Volume I: 475 – 490, 1998.

[4] G. D. Finlayson, and G. Y. Tian, “Perfect Colour Constancy vs Colour Normalization for Object Recognition”, SPIE, Vol. 3826, 1999.