Recognition of Paintings at an Exhibition

Sara Bolouki
EE368 course project, spring 2007

Abstract – This report presents an automatic method for recognition of paintings in a museum. The developed method recognizes the given input picture among 33 classes of paintings and provides the name of the painting as output. The proposed method can be used as an interactive museum guide, which provides more information of the paining to the visitors.

I. Introduction

Recognition is considered to be one of the most important applications of image processing and computer vision. Human vision system performs this task for us in an outstanding manner in everyday life. However, when it comes to computer vision, there are some challenges to overcome; these challenges can be categorized according to different cameras characteristics, change in environment lightening, affine transformations, noise, and changes in point of view. Therefore any proposed recognition method should be as robust as possible to tolerate these changes. In this project a robust recognition method is proposed, implemented and evaluated on a set of 33 classes of paintings. The rest of the report is organized as follows: in Section 2 an overview of the algorithm is presented. In Section 3 feature detection algorithm is described while Section 4 contains details about generating descriptors. In section 5 the evaluation methodology and results are presented and Section 6 concludes the report.

II. Algorithm Overview

The proposed algorithm consists of four stages: preprocessing, feature detection, descriptor calculation and matching. These stages are shown in Figure 1. Since the images are taken with a mobile phone camera and in low light condition of the exhibition, they are suffer a low signal to noise ratio. In order to remove this noise, images are first converted to gray-level representation and then passed through a low pass filter. Sine the resolution of the input image is 2048 x1536, in order to speed up the processing, it is first down-sampled to one fourth of its original size. All these steps are shown as preprocessing in Figure 1. In the next step, feature detection, all the feature points of the preprocessed image are extracted. In the employed algorithm, feature points are defined as corners since corners can be considered as features that remain unchanged by variation of the scale and lightening condition, as well as applying affine transformation. In this project Harris method [1] is used for detecting corners. The third step of our recognition algorithm is calculating a descriptor for each feature point. We use SURF [2] algorithm to describe each feature. Finally the last step of the recognition procedure is to match the descriptors of the given image to the training data, and making the decision; the class with the highest number of matches will be considered as the final match.

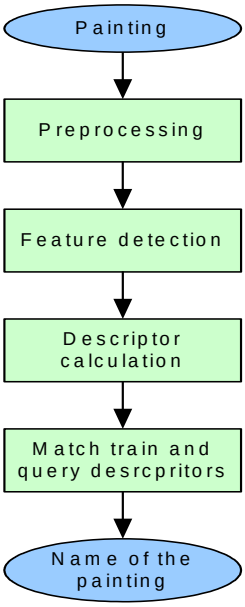

Figure 1 Algorithm overview

III. Feature Detection

As mentioned earlier, we consider corners as robust features in the recognition process. According to Noble [3] and Harris [1],

normalized value of the determinant of the matrix shown in equation 1, describes the corner strength at each pixel.

$$M(x,y)=\begin{bmatrix} \left\langle \left(\frac{\partial I(x,y)}{\partial x}\right)^2 \right\rangle & \left\langle \left(\frac{\partial I(x,y)}{\partial x}\right)\left(\frac{\partial I(x,y)}{\partial y}\right) \right\rangle \\ \left\langle \left(\frac{\partial I(x,y)}{\partial x}\right)\left(\frac{\partial I(x,y)}{\partial y}\right) \right\rangle & \left\langle \left(\frac{\partial I(x,y)}{\partial y}\right)^2 \right\rangle \end{bmatrix}$$

Equation 1

For approximating the gradient at each pixel, Prewitt filter has been used. For noise resilience purposes, the gradients are smoothed by a Gaussian filter. $\langle I \rangle$ in Equation 1 denotes the smoothing operation on I. The local maxima of the determinant of M, is considered as the corner points of the image. Considering the normalized determinant of M as a gray level image, we can use gray-level dilation to calculate the local maxima at a specific neighborhood. The dilated image is then threshold-ed and the points larger than the threshold are considered as corners. Figure 2 shows the detected corners on two sample images. The paintings are the same but the scale is different. Also, the first image is slightly rotated compared to the second one. However, as demonstrated in Figure 2, recognized feature points are consistent.

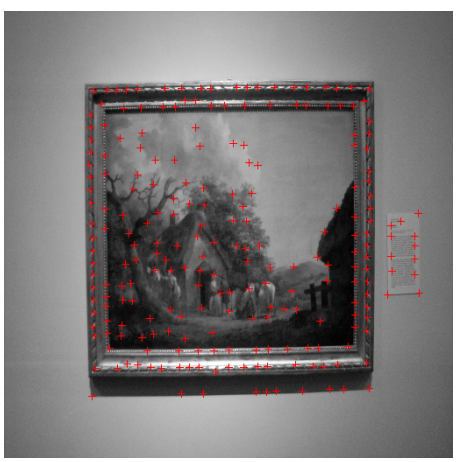

Figure 2 (a) Corner detection results on smaller scale and slightly rotated

Figure 2 (b) Corner detection result on larger scale

In order to be independent of the changes in the scale of input, feature detection is performed in two different scales. The second scale of the input image is calculated using Haar wavelet, and LL part is considered as the second scale.

IV. Descriptor Generation

The next step of the algorithm is to calculate a descriptor for each feature point. The descriptor is desired to be invariant regarding the scale and the orientation. We used a simpler version of the SURF [2] descriptor for describing the characteristics of each feature point. Since in this specific case the method is applied to paintings mounted on a wall, there is not a lot of variation in the orientation. Therefore the dominant orientation is not considered in order to specify the window in which SURF descriptor is calculated. As described in [2] $\sum dx$, $\sum |dx|$, $\sum dy$, $\sum |dy|$, are calculated in a sub-window of size 4x4 in the window around the feature point. This descriptor represents the intensity pattern [2] around each feature point by calculating the gradient in small regions.

As described before, the feature points are detected in two different scales, where the size of the window for each scale is different. If the feature point is detected in the first scale, the window size is 64x64, and if it was selected in the second scale, the descriptor will be calculated in a 32x32 window. Since we want to have an equal number of samples in each scale, in the first scale the 64x64 window is sub-sampled by a factor of two. The size of descriptor is 256 and Sobel filter is used to calculate the gradient in each window.

V. Matching

The final step of the algorithm is to match the descriptors of the query image to the descriptors of the training data, and find the class with highest number of matches. To find the distance between the descriptor vectors, euclidean distance between descriptors are calculated. In order to accept or reject a descriptor as a match, two different methods have been evaluated. In the first method, two descriptors are considered to be matched if their euclidean distance is less than a given threshold. The second method, which was introduced in [2], finds the first and second closest descriptors and declares a match only if the norm value of the first descriptor is grater than 80 percent of the second descriptor. Using this method, there will not be any matches for a desrcriptors if there are many descriptors close to that in the training data. This make the algorithm more robust and also results in a higher recognition percentage compared to using a threshold value.

VI. Evaluation

The implemented algorithm was evaluated on the provided images. Three different experiments were performed to decide the best method and features for the final recognition algorithm. In the first experiment, using a threshold was compared to the nearest neighbor rejection method as described above. This experiments uses 64x64 windows in only one scale. Figure 3 compares the recognition results of methods. As shown in the figure, success rate is very sensitive to the chosen threshold. Therefore, in order to reduce the sensitivity, in the final implementation, first and the second nearest points were compared and the match is recognized based on their distance. In Figure 3 this method is called the nearest point. Each experiment uses two out of three images in each class as the training data, and tests the algorithm on the third image.

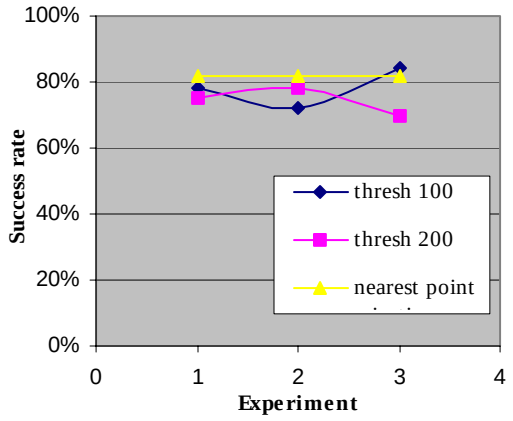

Figure 3 Comparing threshold methods to nearest point match rejection

The second experiment was aimed to compare the role of number of scales in the success rate. The window sizes for the first and second scale are 64x64 and 32x32 respectively. Nearest point method is used for descriptor matching. Figure 4 shows the results of this experiment.

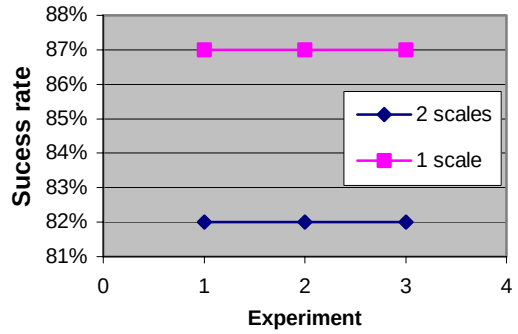

Figure 4 Comparison between number of scales

Considering the results of this experiment, in both training and test data the feature points are calculated in two different scales.

The final experiment compares the running time of the algorithm while varying the number of sampled features of the test data. This experiment is performed using two scales and nearest point descriptor rejection method. During training, all the feature points are extracted and their corresponding descriptors are stored. However during the matching process, only certain number of features are selected randomly, and success rate of the algorithm and the average running time are measured.

Table 1 Run time and success rate comparison according to number of sampled features

Num features	Run time (sec)	Success rate
200	70	87%
150	52	87%
135	46	87%
100	34	82%

Considering the results of the Table 1, 135 features were randomly sampled in the recognition algorithm to speed up the procedure while maintaining the performance.

VII. Conclusion

In this project a method for recognizing a painting was proposed. The proposed method uses the ideas of Harris corner detection and SURF descriptor while trying to optimize it for the specific application. This method can successfully recognize 87% of the paintings, while being trained on two images in each class.

VIII. References

[1]1 C. Harris, M. Stephens, "A combined corner and edge detector", *Proc. of 4th Alvey Vision Conference*, pp. 147-151, 1988.

[2] H. Bay, T. Tuytelaars, L.Van Cool" SURF: Speeded up robust features", *Proc. 9th European Conf on Computer Vision*, 2006

[3] J.A. Noble, "Finding corners", *Image and Vision Computing*, pp. 121-128, 1988.