Identification of Paintings in Camera-Phone Images

Gabriel M. Hoffmann, Peter W. Kimball, Stephen P. Russell
Department of Aeronautics and Astronautics
Stanford University, Stanford, CA 94305
{gabeh,pkimball,sprussell}@stanford.edu

***Abstract*— This paper presents the algorithm developed, for the EE 368 course project, to identify paintings in query images taken with a camera-phone in the European Gallery of the Cantor Arts Center. The image processing algorithm first extracts the painting of interest, in the center of the photo. It uses edge detection in saturation space, followed by iterative Radon transforms and region removal to find the smallest bounding box enclosing the painting. Using the smallest bounding box, the algorithm computes and performs a projective transform to extract and rectify the painting in the query image. Feature identification is performed on the rectified image using extrema in a difference-of-Gaussians pyramid. At each resulting feature location, feature descriptors are computed using histograms of the gradient distribution among neighboring pixels, without rotation. The resulting features are correlated with a feature database of paintings with aspect ratios similar to that of the painting in the query image. A program to produce the feature database was also developed. This program selects a small set of unique, but repeatable, features from the many available for each painting in the gallery.**

The image processing algorithm correctly identifies paintings in all 99 training images provided by the EE368 teaching staff. It performs well on the additional 93 test images taken by the group, in difficult configurations, to stress test the algorithm. Details and sensitivities of the algorithm are discussed.

I. Introduction

Recognition of objects in images has practical applications across many disciplines. One potential "Augmented Reality" application is the recognition of paintings in an art exhibit. Gallery visitors could take a picture of a painting with a cell phone, and then receive an automated phone call or text message containing information about the painting such as title, artist, historical context, and critical review. This service requires rapid identification of the painting in the image, without strong guarantees on lighting, viewing angle, and size of the painting in the image. Compression artifacts and low light effects, such as blur and noise, corrupt such images to varying degrees. The painting recognition task can be automated using techniques from the fields of digital image processing and computer vision [1].

After consideration and prototyping of several candidate techniques, it was found that this problem has some unique characteristics. First, the low light conditions cause high variation in the quality of the images, making the correlation of many feature descriptor types difficult. Second, there may be multiple paintings, and even other works of art, in the images. Third, the paintings from the same era showed many recurring styles and themes, reducing the uniqueness of features. Visually, patches of some paintings can be seen in other paintings.

This paper presents the algorithm developed for the EE 368 course project, to identify the paintings in the centers of query images taken with a camera-phone. The images under consideration were recorded in the European Gallery of the Cantor Arts Center. The image processing algorithm first extracts the painting of interest, in the center of a query image. It uses edge detection in saturation space, followed by iterative Radon transforms and region removal to find the smallest bounding box enclosing the painting. Using the smallest bounding box, the algorithm computes and performs a projective transform to extract and rectify the painting in the query image. This step reduces the size of the image for feature detection, and reduces the requirements for robustness of the feature descriptors, such as perspective invariance. Feature identification is performed on the rectified image using extrema in a difference-of-Gaussians pyramid. At each resulting feature location, feature descriptors are computed using histograms of the gradient distribution among neighboring pixels, without rotation. The resulting features are correlated with a feature database of paintings with aspect ratios similar to that of the painting in the query image. A program to produce the feature database was also developed. This program selects a small set of unique, but repeatable, features from the many available for each painting in the gallery. Because the paintings are rectified, using the knowledge that they are rectangular, more general correspondence algorithms are not required to perform correspondence in affine or projective spaces.

The image processing algorithm, implemented in Matlab script files, correctly identifies paintings in all 99 training images provided by the EE368 teaching staff, with a mean time of 5.2 seconds per query image on the SCIEN machines. An additional 93 test images were taken by the group in configurations expected to be difficult, to stress test the algorithm, such as with excessive shaking, extreme perspectives, long distances, and more occlusions. This allowed minor modifications to the parameters so that the algorithm could perform well with images not considered during development. Details and sensitivities of the algorithm are discussed.

II. Background

There are two major challenges involved in the art recognition application described above. First, there must be a way to extract the region of interest from the image. Second, it is necessary to describe this region in a distinctive way, one that is repeatable and can be cross referenced to a database of information that uniquely identifies the painting in question.

In order to extract a section of an image, it is necessary to have some information about the region in question. For this application, it is assumed that the painting is in a standard art gallery. That is, the painting is posted on a white wall with some space between it and adjacent pieces of art. It is fine for other works to be present in the query image, so long as there is some white space to distinguish different pieces of art. Also, it is assumed that the desired painting is roughly centered in the image frame. More specifically, the center pixel in the image is assumed to be part of the painting of interest. With these restrictions in place, a particular region in the center of the query image can be sectioned out. Techniques such as the Radon transform [2] and correspondence testing prove extremely useful in performing the image segmentation and rectification.

The image processing field employs many techniques for examining and describing the content of an image [1]. Image transforms are one way to extract information about the content of a digital image. Many times, this can be useful for sorting through noise or other undesirable details and extracting the fundamental behavior of the image. Examples of such transforms include the discrete Fourier transform (DFT) and the discrete cosine transform (DCT). These both classify the frequency content of the discrete intensity image. The DCT was tested for this project, but found to be too susceptible to image rotation error and projective rectification error, which shift frequency content between vertical and horizontal dimensions. Another approach would be to quantify a particular type of content in the image. The red-green-blue (RGB) channels or the hue of an image could be histogrammed, i.e. counting the number of pixels that have particular values. If this histogram is normalized by the number of pixels in the image, this description becomes somewhat scale invariant. In other words, the image is classified based on the percentage of pixels that have particular color/hue values. This vector of percentages creates a sort of "fingerprint" of the image content, which could be referenced to identify an image of the same painting. However, since many paintings are composed of similar paints and colors, it was found that this method is weak at uniquely identifying paintings.

Another method of image classification finds distinct features in an image, rather than looking at global image content. There are many types of features that can be extracted from an image, ranging from simple things like corners and edges to more complicated things like SIFT vectors or SURF vectors. Corners and edges are popular because they are easily recognizable and easily verified as being correctly found in the image. However, by themselves, these features are not especially good for repeatably and uniquely classifying an image [1]. Specifically, prototype algorithms showed corner features to be largely inadequate due to variations in scale and focus in the training images.

It is desirable to have features which have some invariance to common image differences, such as changes in scale and illumination or even affine transformation. The Scale Invariant Feature Transform, or SIFT, method is a very popular algorithm for generating such features [3]. The descriptor vector is created by histogramming gradient values in a scaled window around a keypoint. Using the parameters suggested in [3], this vector lies in 128 dimensional space and thus can be a very unique description of the feature. These features have proven to be quite robust to the above transformations and even to moderate perspective transformation. A large volume of recent work has been based on SIFT features and much support for the use of SIFT exists in the literature. A more recent addition to the field is the Speeded Up Robust Features, or SURF, descriptor. The authors of SURF claim that this method outperforms previously proposed schemes with respect to repeatability, distinctiveness, and robustness, yet can be computed and compared much faster [4]. However, descriptor computation using SIFT descriptors was found to be fast relative to algorithm execution time. The execution time was largely composed of more complex tasks, such as finding extrema in 3 by 3 by 3 cubes, a task required in both the SURF and SIFT keypoint detection algorithms.

III. Image Processing Algorithm

This section details the implemented algorithm for painting recognition. It is robust to changes in lighting and perspective, and has been used to identify paintings from the Cantor Arts Center European Gallery. It works in three sequential parts. First, the painting is segmented out of the image and rectified to a straight-on view. Second, a modified SIFT is performed on the rectified painting. Finally, the resulting feature descriptors are correlated with a database of automatically chosen feature descriptors for all the paintings in the gallery having similar aspect ratios to that of the rectified painting. The correlation results in a numerical score for each painting. The title of the painting with the highest correlation score is returned.

A. Painting Rectification

The query image is first downsampled by averaging square regions, to reduce JPEG artifact effects. Using sparse matrices, this outperforms Matlab's *imresize* function. One copy, to be output, is downsampled by a factor of two in both directions. A second copy, to be used in processing, is downsampled by a factor of eight in both directions (by downsampling the previous image by a factor of 4, for efficiency). The saturation values of the small RGB are computed, according to the formula $S = 1 - \frac{MAX}{MIN+eps}$, where MAX and MIN are with respect to the red, green and blue values, and eps is machine precision, so that the limit for low saturation is zero. Both the saturation and output images are median filtered with a three-by-three kernel to remove noise. See figures 1 and 2.

Next, edge detection is performed on the saturation image using a Laplacian of Gaussian kernel. The saturation image is used instead of a grayscale or color-component image because of the reduced sensitivity to shadow in the saturation space. Note that there is some sensitivity due to the color of whatever ambient light is incident on the shadows. The

Fig. 1. Query image, downsampled by 2x and median filtered.

Fig. 2. Saturation values of the 8x downsampled query image, median filtered.

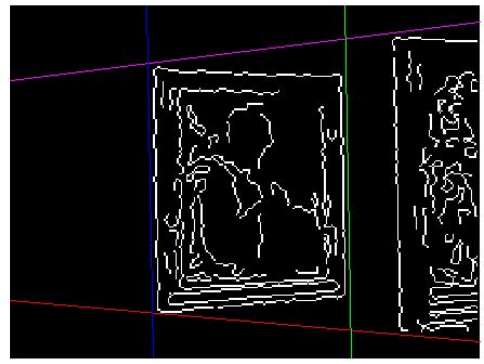

Fig. 3. Bounding Box

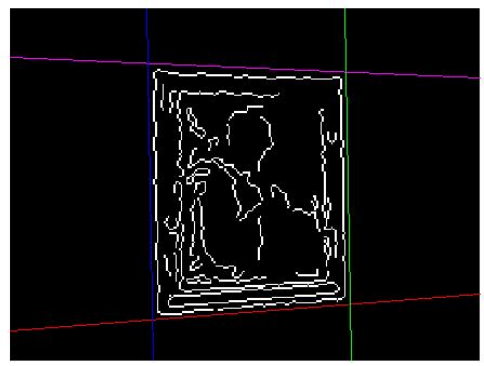

Fig. 4. Bounding Box After Blackout

edge map is thresholded to produce a binary image showing edges in the saturation space of the image. Small regions are removed from this map.

A Radon transform is performed [2], though the objective is not to find lines in the image. Rather, the objective is to find the closest lines to the center which pass through *no* edge pixels. The transform histograms the number of pixels in the edge map which lie on lines parameterized by angle and distance from image center. To improve efficiency, the transform is only performed for angles within 25° of horizontal and vertical.

Using the results of the Radon transform, the line which crosses no edge pixels and is closest to the origin is found on each of the four sides (top, bottom, left, right) of the painting. These four lines define a box which tightly bounds the painting in the image when other objects are not present. However, objects other than the painting of interest often appear in the query images and move the bounding box away from the painting (figure 3). So, edges outside the bounding box must be deleted, and the radon transform performed again. This process is repeated until the lines defining the bounding box do not change (figure 4). The result is not sensitive to local minima or other common line-seeking difficulties, and is able to fit boxes around frames with substantial texture, but few lines. This method is robust to the appearance of other objects in the photo, as shown during testing with challenging images taken by the group in the museum.

The bounding box is scaled from the saturation image to the larger, gray-level output image. A perspective transform is computed using the four points at the intersections of the bounding box lines and applied to output image. The image is then cropped to the bounding box, which has become rectangular under the projective transform. Finally, it is proportionally resized using a bilinear transform to a width of 300 pixels, and output(figure 5). Rectifying the image to a given width is necessary for subsequent stages of the algorithm.

B. Modified SIFT

A modified version of the algorithm in [3], the Scale-Invariant Feature Transform (SIFT), is performed on the rectified query image. This transform generates feature descriptor vectors for keypoints on varying scales in the image.

Fig. 5. Rectified Gray Level Image

Keypoints are defined as the local maxima or minima over all three dimensions in a multi-octave difference-of-Gaussians (DoG) image stack. The algorithm creates the DoG stack, blurring successive levels with broadening Gaussian kernels, and taking the difference between levels. Keypoints are points in the stack whose values are maximal or minimal, compared to their nine neighbors above, eight neighbors around, and nine neighbors below. To find the maxima, each image in the DoG stack is compared with its dilation by a three pixel square structuring element (implemented as the composition of a 1 by 3 element and a 3 by 1 element in the Matlab function). A pixel is a local maximum if and only if its value is equal to the pixel at the same location in the dilated stack and greater than the values immediately above and below it in the dilated stack. Local mimima are found in a similar way, by comparing the DoG stack with its erosion by a three pixel square structuring element. The use of dilation and erosion operations for peak detection allows the use of compiled components of Matlab, rather than costly for-loops. A logical array with the same dimensions as the DoG stack (minus the exterior layer of values) is efficiently created using these definitions.

Keypoints along edges in an image are poorly localized along the direction of the edge. So, the logical array of keypoint locations is element-wise multiplied by zeros at locations along edges, and by ones away from edges, where keypoints are allowed to exist. Non-edge pixels are defined based on ratios of the eigenvalues of the Hessian matrix as defined in [3].

Next, each keypoint in the DoG stack is given a feature descriptor similar to those used by SIFT[3]. Some computation is saved relative to SIFT, however, by leveraging the image rectification. Since this algorithm only processes rectified images, ”right-side-up”, the feature descriptors need not be rotationally-invariant, and feature orientation need not be computed. As in SIFT, each feature descriptor is based on the gradient orientation in a region surrounding the keypoint. The descriptor vector is a stacked series of 16 eight-bin histograms of gradient direction. Each of the 16 histograms describes the gradient direction distribution in a small window near the keypoint. These windows are arranged in a four-by-four grid, with the keypoint at the center.

The output of the modified SIFT is then the list of keypoint locations and their corresponding feature descriptor vectors.

C. Correlation with Database

Once the modified SIFT features have been extracted from the rectified query image, the final step is to correlate this array of feature vectors against the gallery database (section III-D is a discussion of the creation of this database). Two metrics are used to make comparisons with the given training images.

First, the modified SIFT features are compared to each painting in the database, by taking dot products between each descriptor from the test image and each of those for a particular database painting. Correlation is only performed for database paintings whose aspect ratios are near that of the painting in the query image. A threshold value of 0.3 was experimentally found to yield the best tradeoff between correctly rejecting false images and incorrectly rejecting the correct image, due to rectification error from shadows and discretization of the bounding box (a value of 0.1 was effective for the test images provided by the course, but was too strict for the more distorted images recorded by the group). The aspect ratio test typically rejects half of the image database. The subset of features from the test image that have the largest dot products are extracted from the list. These values are squared to give stronger preference to those values close to 1 and suppress those that are smaller.

Second, the squared dot product scores are weighted based on the Euclidean pixel distance between the location of the feature in the test image and the corresponding location in the database image. This step demands that query paintings all be rectified to a given width. Weights are computed according to the following function:

$$w(d) = e^{-\alpha d^2}$$

A good value for α was empirically determined to be 0.001. This function causes the weighting to be less than 0.1 by the time the distance between potentially matching features exceeds 50 pixels. This threshold allows for some inconsistency in the rectified images while still punishing features that have similar descriptor vectors but are in completely different regions of the image.

The weighted scores for each feature are summed to give the cumulative score for a particular painting. The painting with the highest score is declared the match and its name is accessed from the database and returned.

D. Database Creation

It is important that the database of features distinguishes the different paintings. The solution implemented finds feature descriptors that are consistent between the training images of a particular painting, but also unique in that they do not appear strongly in the other paintings. The code written for database creation is general and automated; a

new database could be easily created from a new gallery of training images.

The first step is to find features that are repeatable between multiple images of each single painting. From the three images of a given painting, the one with the largest number of modified SIFT features is examined. A list of strong features from this primary image is created by finding those features that also appear in either one of the other two images. Two criteria are used for "appearance" in another image. The first is descriptor similarity, determined by taking dot products between feature vectors. Only those with dot products above 0.75 are considered further. The second test is proximity to the location of the feature in the primary image. A distance threshold of 20 pixels is used to distinguish acceptable features to keep (again, all images have been rectified to a 300 pixel width).

Having generated a list of features that are consistent between images of a given painting, the next step is to eliminate any features that might be confused with a similar feature in another painting. The same metrics for similarity used above are considered to find similar features across paintings. That is, if two features have a dot product greater than 0.75 and are within 20 pixels of each other (in their respective images), then they are discarded from both paintings' lists.

Finally, the number of features, n, to keep for all paintings is determined by the painting with the fewest number of consistent and unique features. This is so all paintings have the same number of strong, descriptive features; this makes scoring the correspondence with a test image easier. The features for each painting are ordered based on their self-consistent dot product strength and only the top n features and their locations are stored to the database.

IV. Algorithm Performance

The algorithm correctly identifies the 33 paintings in all 99 images of the EE368 training data. The ratio of the correlation score for the correct painting to that for the next-highest-scoring painting exceeds 1.5 in all cases. Additional images of the paintings in the European Gallery were taken to further test the performance of the algorithm. Some images were taken under conditions similar to those in which the training images were taken. Others were taken in more adverse conditions such as from highly-oblique angles, with excessive motion blur or camera defocus, with painting occlusion, and with white-space occlusion. While the algorithm performs with 100% success on the images taken in nominal conditions, it eventually fails under varying degrees of adversity for each of the various degredations. The most sensitive failure mode is occlusion of the white-space surrounding the painting in an image. Under increasing motion blur, the algorithm more successfully identifies paintings with large features than those without.

The algorithm takes a mean of 5.2 seconds to run on Stanford's SCIEN computers for a 3.0 Mpixel JPEG query image, with a standard deviation of 0.5 seconds. Image rectification takes 2-3 seconds depending on painting size. The modified SIFT takes 2-3 seconds, depending on the number of keypoints found in the painting, and the database correlation takes less than 0.01 seconds.

V. Conclusion

Motivated by applications to "Augmented Reality," an algorithm has been developed to identify paintings in the Cantor Arts Center's European Gallery. In the algorithm, the painting in the query image is identified with a bounding box, and rectified using a projective transform. Features in the rectified image are identified, and feature descriptors are formed for each one. These feature descriptors are correlated with those in a feature database. Features in this database are chosen to maximize repeatability over multiple images of the same painting, and uniqueness to images of the correct painting.

The developed algorithm is capable of identifying paintings in the training images as well as in additional images gathered with a digital camera. It leverages the assumptions that the center pixel of a query image lies within the painting of interest, that there is substantial white wall space surrounding the painting, that the frames are predominantly rectangular, and that query images are taken from a viewing angle such that the apparent orientation of the edges deviates no more than 25° from horizontal and vertical. The algorithm is robust to changes in viewing angle and in lighting, as well as to modest amounts of motion blur and camera defocus. The algorithm is especially sensitive to occlusion of the white space surrounding paintings. On Stanford's SCIEN machines, the algorithm runs in 5.2 seconds.

References

[1] Girod, B., "Course Notes," *EE368 - Digital Image Processing*, Spring 2007.

[2] Bracewell, R. N., *Two-Dimensional Imaging*, Prentice Hall, Englewood Cliffs, NJ, 1995.

[3] Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, Vol. 60, No. 2, November 2004, pp. 91–110.

[4] Herbert Bay, Tinne Tuytelaars, L. V. G., "SURF: Speeded Up Robust Features," *Proceedings of the Ninth European Conference on Computer Vision*, May 2006.

Appendix

A. Group Work Log

We split the work up so that each group member had tasks at any given time, and we all contributed a roughly equal amount of time to the project. Portions that we all worked on, close to equally, included brainstorming, a literature search, code optimization and testing, and report writing. The portions that we individually focused on follow, although we helped each other with many of the tasks.

Gabriel's work included: trying image matching using DCT coefficients, using morphology to perform edge detection, a homography transformation estimator, Radon image processing, designing the iterative Radon-blackout algorithm, SIFT keypoint detection, taking pictures at the museum, and using them for tuning.

Peter's work included: cleaning up the Radon image processing code, implementing the iterative Radon-blackout

algorithm, edge detection using Laplacian of Gaussians, small region removal, projective image transformation and cropping, and automatic selection of features to match.

Stephen's work included: trying image matching using RGB histograms, implementing modified SIFT descriptors, implementing feature matching, and automatic selection of features to match.