Efficient and Effective Transformed Image Retrieval Using SIFT Descriptors

Mohammad Awrangjeb*, Member, IEEE, and Guojun Lu, Senior Member, IEEE
Gippsland School of IT, Monash University, Churchill Vic 3842, Australia
Phone: +61 3 5122 6138, Fax: +61 3 5122 6879
E-mail: {Mohammad.Awrangjeb, Guojun.Lu}@infotech.monash.edu.au

Abstract

The SIFT (scale invariant feature transform) approach has demonstrated its superior performance in identifying transformed images over many other approaches. However, both of its detection and matching stages are expensive, because a large number of keypoints are detected in the scale-space and each keypoint is described using a 128-dimensional vector. Further research has been carried out in the literature for the dimensionality reduction of the SIFT descriptor, but with the expense of loss of robustness. We present two possible solutions for feature-point reduction. The first solution is to down scale the image before the SIFT keypoint detection and the second solution is to use corners (instead of SIFT keypoints) which are visually significant, more robust, and much smaller in number than the SIFT keypoints. Either the curvature descriptor or the highly distinctive SIFT descriptors at corner locations can be used to represent corners. Our experimental results reveal that two feature-point reduction solutions combined with the SIFT descriptors and our previously proposed feature-point matching technique not only improve the computational efficiency and decrease the storage requirement significantly, but also increase the transformed image identification accuracy (robustness).

Index Terms

Feature detection, Feature representation, Feature matching, Transformed image identification.

EDICS: 4-KEEP Indexing, Searching, Retrieving, Query, and Archiving Databases.

I. INTRODUCTION

In many applications, such as image copyright protection [1] and object recognition [2], a common problem is to identify images which may have undergone unknown transformations. We define this

*corresponding author.
common problem as the transformed image identification (TII), where the goal is to identify the geometric transformed and the signal processed images for a given image. Therefore, the TII is different from conventional content-based image retrieval (CBIR) [3], where all images having the same or similar features, e.g., similar colors, are considered relevant to each other. The TII consists of three main stages [4]: feature detection, feature representation, and feature matching. In the feature detection stage, a set of features, e.g., corners, blobs, T-junctions, are detected. The most valuable property of a feature detector is repeatability, i.e., whether it reliably finds the same feature points after different transformations. In the feature representation stage, each detected feature point is represented by a feature vector calculated from its neighborhood. In the feature matching stage, the feature vectors of the test image and the stored images are compared to identify transformed images for the test image. The matching is often based on a distance, e.g., the Euclidean distance [4], between the feature vectors.

The SIFT (scale invariant feature transform) approach [2] has demonstrated its superior performance in identifying transformed images over many other approaches [9]. However, both of its detection and matching stages are expensive, because a large number of keypoints are detected in the scale-space and each keypoint is described using a 128-dimensional vector. Further research has been carried out in the literature for the dimensionality reduction of the SIFT descriptor, but with the expense of loss of robustness [4], [9], [10]. In this paper, we propose to reduce the number of detected feature points. We present two possible solutions for feature-point reduction. First is to down scale the image before the SIFT keypoint detection and second is to use corners (instead of SIFT keypoints) which are visually significant, more robust, and much smaller in number than the SIFT keypoints. Either the curvature descriptor or the highly distinctive SIFT descriptors at corner locations can be used to represent corners. Consequently, we propose five TII approaches in this paper and will discuss them in Section III-D. The two feature-point reduction solutions combined with the SIFT descriptors and our previously proposed feature-point matching technique [11] not only improve the computational efficiency and decrease the storage requirement significantly, but also increase the transformed image identification accuracy (robustness).

The organization of this paper is as follows. In Section II, we briefly review the existing feature detection, representation, and matching techniques. In Section III, we present the proposed TII approaches. In Section IV, we discuss the performance study and finally, in Section V we conclude the paper.

II. PREVIOUS WORK AND CONTRIBUTIONS OF THIS PAPER

In this section, we first briefly present the existing work in three stages of the TII and then present the contribution of this paper.
A. Feature Detection

A large number of corner and interest-point detectors have been proposed in the literature [2], [12]–[20]. While corner detectors detect image spatial locations where edge segments make significant angles, interest-point detectors not only detect corners, but also image locations that have large gradients in all directions at a predetermined scale [2].

All corner and interest-point detectors can be broadly classified into two groups: single-scale detectors [18]–[20] and multi-scale detectors [2], [12]–[17]. Single-scale detectors work well if the image has similar size features, but ineffective otherwise; because either fine or coarse scale feature is poorly detected, but images may contain both kinds of features. To improve the effectiveness of the detection stage, multi-scale detectors have been proposed.

Corner and interest-point detectors can also be categorized into three groups: intensity-based [2], [12], [13], [19], contour-based [14]–[18], and model or template-based [20] methods. Intensity-based methods estimate a measure which is intended to indicate the presence of an interest-point directly from the image pixel values. Contour-based methods first obtain planar curves using some edge detector and then search for the curvature maxima along those curves. Model or template-based methods find corners by fitting the image signal into a predefined model.

The main drawback with the model-based detectors is that the corner in natural images cannot be approximated by a model of a perfect corner, as it can take any form of the bidirectional signal change [13]. Moreover, they are computationally too expensive [20] and are not used for general purpose [21]. This paper will focus on the intensity and contour-based detectors.

1) Intensity-based Detectors: Probably the most widely used detector is the Harris interest-point detector [19] which is based on the eigen values of the second-moment matrix. However, Harris points are not scale-invariant [4]. Lindeberg [12] introduced the concept of automatic scale selection which allows detecting interest points in an image, each with their own characteristic scale. Mikolajczyk and Schmid [13] refined this technique by creating robust and scale invariant features with high repeatability. They used a scale-adapted Harris measure or the determinant of the Hessian matrix to select the location, and the Laplacian to select the scale. Lowe's [2] approximation of the Laplacian of Gaussian using a Difference of Gaussian (DoG) filter speeds up the feature detection stage significantly. Note the DoG points will be used as SIFT keypoints in the rest of the paper. The recently proposed fast-Hessian detector in the SURF (speeded up robust features) detector-descriptor scheme [4] used a basic approximation of the Hessian matrix and relied on the integral images to reduce the computational cost.

2) Contour-based Detectors: The CSS (curvature scale-space) detector in [16] is one of the earlier contour-based multi-scale detectors. It detected corners at a high scale and tracked them through multiple lower scales in order to improve localization. Since different curves require different
smoothing-scales and there may be different sizes of corners on the same curve, this detector is highly sensitive to the use of a single corner detection scale and a single fixed curvature-threshold. He & Yung [18] improved this detector by using the adaptive curvature-threshold and the dynamic region-of-support on both sides of each curvature extremum point. Zhang et al. [17] further improved it by introducing the idea of curvature-product. In terms of curvature product, the strong corners become more distinguishable from the weak and false corners. Awrangjeb et al. [14] proposed another improvement by selecting three corner detection scales based on the curve’s affine-length.

Awrangjeb and Lu [15] pointed out the two main problems of the above CSS-based detectors. First, the CSS curvature estimation technique is highly sensitive to the local variation and noise on the curve. Second, the CSS corner detection technique requires appropriate Gaussian smoothing-scale selection which is a difficult task. To overcome these two problems, Awrangjeb and Lu [15] proposed a new corner detector based on the chord-to-point distance accumulation (CPDA) for the discrete curvature estimation [22], which is less sensitive to the local variation and noise on the curve and does not require appropriate Gaussian smoothing-scale selection.

B. Feature Representation

In order to facilitate feature-point matching in any subsequent application, each feature-point must be represented with some of its associated information. The more the representation is distinctive, the less the number of false candidate matches will be obtained in the matching stage. The feature representation is also known as the descriptor in the literature.

There are two different types of representations found in the literature: geometric descriptors and local descriptors.

1) Geometric Descriptors: This type of representation [23], [24] is purely geometric, where each corner or feature-point is represented using its curvature, angle, and distances from neighbor corner-points. Zhou et al. [23] used the angles of Delaunay triangles which are formed among the Harris interest-points [19]. These angles are invariant to image translation, rotation, and uniform scaling. Huttenlocher used [24] distance ratios defined by the quadruples of the feature-points. The distance ratios are invariant to affine transformations.

Though this kind of representation is easy to design and requires quite small amount of storage per feature-point, the representation is not unique. As a consequence, either the true correspondences are missed or a huge number of false correspondences are obtained between two images and the matching procedure becomes prohibitively expensive. This is why, the use of geometric representation is quite limited in the literature.

2) Local Descriptors: This type of representation [2], [4], [9], [10] is based on the pixel values in a specific neighborhood around each feature-point. They are harder to design, but more distinctive
than the geometric representations discussed above.

The SIFT-based descriptors, originally proposed by Lowe in [2], have been described as the best among the different types of feature descriptors [9]. The original SIFT descriptor [2] captures a substantial amount of information about the spatial intensity patterns around a feature point. For each descriptor it computes a 3-D histogram of gradient location and orientation, where the location is quantized into a $4 \times 4$ location grid and for each location grid (total 16) the gradient angle is quantized into 8 orientations. The resulting descriptor is a 128-dimensional vector (8 orientation bins for each of $4 \times 4$ location bins). This vector is robust to small deformations or localization errors. To obtain illumination invariance, the descriptor is normalized by the square root of the sum of squared components.

Further research has been carried out in the literature either to reduce the descriptor’s dimension [4], [10] or to make it more distinctive [9], [10]. Though the lower dimensional PCA-SIFT (principal component analysis-SIFT) [10] and SURF [4] are helpful for fast feature matching, PCA-SIFT was proved to be less robust than SIFT under affine transformations [9] and SURF was designed to handle rotation and scale attacks only as a compromise between feature complexity and robustness [4]. The GLOH (gradient location-oriented histogram) [9] is another variant of SIFT which was proved to be more distinctive and robust than the original SIFT [9]. However, it is computationally more expensive [4]. Consequently, in spite of the above refinement versions, the original SIFT is still the most popular.

The above local descriptors are invariant to image rotation, scaling, and translations. The affine invariant local descriptors which are based on the iterative affine-region detection around the feature-points are computationally very expensive [13].

C. Feature Matching

Mikolajczyk and Schmid [9] evaluated three feature matching techniques. In threshold-based matching, two features are matched if the distance between their descriptors is below a predefined threshold. A feature may have several matches in this strategy. In nearest-neighbor-based matching, two features $F_1$ and $F_2$ with descriptors $D_1$ and $D_2$ respectively are matched if the descriptor $D_2$ is the nearest neighbor of $D_1$ and their distance is below a threshold. With this approach a feature should have at most one match. In nearest-neighbor-distance-ratio-based (NNDR matching) matching, which is similar to nearest-neighbor-based matching, the threshold is applied to the distance ratio between the first and second nearest neighbor matches, i.e., two features $F_1$ and $F_2$ with descriptors $D_1$ and $D_2$ respectively are matched if $|D_1 - D_2|/|D_1 - D_3| < t$, where $D_3$ is the descriptor of the second nearest neighbor match $F_3$ of $F_1$. In this approach, a feature has also at most one match.

The worst case (as well as the average case and the best case) running time of these algorithms is $O(mn)$, where $m$ and $n$ are the numbers of features in two images. Note that using these matching
techniques a repeated feature may be missed or there may be some false positive matches.

Awrangjeb and Lu [11] proposed a geometric point matching (GPM) technique where the initial candidate matches are obtained using threshold-based matching discussed above. Then for each combination of three candidate matches, they estimate the affine transformation parameters between the images using an iterative procedure. In each iteration, they transform all the feature-points in one image using the estimated parameters and find matches in the other image. The algorithm tracks the set of parameters that offer the highest number of matches over all the iterations. This algorithm, though offers better matching performance, is more expensive than any of the above three matching techniques. However, in practice, if the feature representation is very distinctive, it costs the same as the above techniques, i.e., $O(mn)$.

D. Contributions of This Paper

In the original SIFT approach [2], a large number of keypoints (normally many hundreds) are detected using DOG filter in the scale-space and each keypoint is represented using a 128-dimensional vector. Therefore, the storage requirement is quite high and all three stages of TII become computationally expensive using the SIFT approach. The reduction in descriptor’s dimension [4], [10] resulted in loss of robustness [4], [9]. A better technique should aim to reduce both storage and computational cost without compromising robustness.

In this paper, as an aim to reduce the number of feature-points we propose two solutions. In the first solution, we down scale the input image before finding the SIFT keypoints. In the second solution, instead of keypoints we use corner-points, which are visually significant, more robust, and much smaller in number than the SIFT keypoints. Either the curvature descriptor or the highly distinctive SIFT descriptors at corner locations can be used to represent corners. Our previously proposed GPM technique [11] can be used for matching both the down-scaled SIFT keypoints and corners detected by a contour-based corner detectors [14], [15]. Experimental results show that the two feature-point reduction solutions combined with the SIFT descriptors and the GPM technique not only improve the computational efficiency and decrease the storage requirement, but also improve the TII accuracy (robustness).

The contributions of the paper can be summarized as follows.

- We describe a complete TII process consisting of three stages – feature detection, representation, and matching (Section III).
- We propose two solutions (with five approaches, see Section III-D) to improve the efficiency of existing SIFT-based approach. We use the contour-based corners or DoG keypoints as feature-points, curvature or SIFT descriptors to represent them, and our previously proposed GPM technique for matching them.
We compare the performance of three feature detectors (Section III-A4). It is observed that corners possess higher repeatability and suffer lower localization error than their keypoints counterparts.

We identify the best TII approaches through experimentation and analysis (Section IV). The two feature-point reduction solutions combined with the SIFT descriptors and the GPM technique not only improve the computational efficiency and decrease the storage requirement, but also improve the TII accuracy.

III. PROPOSED TRANSFORMED IMAGE IDENTIFICATION APPROACHES

For identifying the transformed images for a given test image, all three stages – feature detection, description, and matching – should be efficient and effective. The original SIFT approach detects a several hundreds to a few thousands of keypoints from a medium size image. Its detection stage is slow because of the use of scale-space. Moreover, for many applications, e.g., image copyright protection, this huge number of keypoints also make the later stages unnecessarily slow.

To overcome this problem, we propose two solutions. The first solution is to down scale the input image before DoG keypoint detection. So, the detection stage becomes faster and the number of detected keypoints is reduced exponentially, which also speeds up the later stages. The highly distinctive SIFT descriptors are used to represent the keypoints and the GPM technique is used for keypoint matching.

As a second solution, we propose using corners instead of SIFT keypoints. In general, corners offer the following advantages over keypoints:

- Corners are visually distinguishable and more robust than their keypoints counterparts.
- In an image, the number of corners is much lower than the number of keypoints.
- Corners can be ranked based on their strength like the curvature value or the number of corners can be controlled by changing the detection thresholds. Therefore, a particular number of strong corners can be selected based on the application and it gives further reduction in computational cost during matching. In contrast, it is very hard to rank the SIFT keypoints.
- Corner detection requires less time than keypoint detection in the scale-space.

We use corners detected by our previously proposed two detectors: affine resilient CSS (ARCSS) [14] and CPDA [15] detectors. Either the curvature descriptor or the highly distinctive SIFT descriptors at corner locations can be used to represent corners and the GPM is used for corner matching.

Consequently, in this paper we propose the following five approaches of TII:

- In the first approach, we use the ARCSS corners as feature-points, the curvature descriptors to represent the corners, and the GPM for corner matching.
In the second approach, we use the CPDA corners as feature-points, the curvature descriptors to represent the corners, and the GPM for corner matching.

In the third approach, we use the DoG keypoints as feature-points, the SIFT descriptors to represent the keypoints, and the GPM for keypoint matching, and

In the fourth approach, we use the ARCSS corners as feature-points, the SIFT descriptors to represent the corners, and the GPM for corner matching.

In the fifth approach, we use the CPDA corners as feature-points, the SIFT descriptors to represent the corners, and the GPM for corner matching.

In the following subsections we first briefly present the detectors (Section III-A) which we use with our proposed TII approaches. In Section III-B, we present the descriptors used with our proposed TII approaches. We then present the GPM technique (Section III-C). Finally, we describe the five aforementioned approaches of TII (Section III-D).

A. Feature Detectors

We represent three feature detectors – namely, ARCSS corner detector [14], CPDA corner detector [15], and DoG keypoint detector [2] – used in our proposed TII approaches below. We also present a comparative performance study of these detectors.

1) ARCSS Detector: The ARCSS corner detector [14] extracts edges in the gray-scale image using the Canny edge detector [25]. It then parameterizes each curve (edge) with the affine-length. In order to eliminate noise, it convolves each parameterized curve using the Gaussian kernel in one of three medium scales based on the curve’s affine-length. Thereafter, it calculates absolute curvature value on each point of the smoothed curves and considers curvature maxima points as candidate corners. Both weak and false corners are removed using the appropriate thresholds. Finally, corners are tracked down to the finest scale to improve localization.

The experimental results in [14] showed that in geometric transformations the ARCSS detector outperformed existing CSS detector [16], which had outperformed many other detectors including the Harris interest-point detector [19].

2) CPDA Detector: The CPDA corner detector [15] first extracts planar curves from the edge image detected by the Canny edge detector [25]. Each curve is then smoothed with a small width Gaussian kernel in order to remove quantization noise and trivial details. In order to make strong and weak corners more distinguishable, we first use three chords of different lengths to estimate three normalized discrete curvature values on each point of the smoothed curve. Then we multiply the normalized curvatures to obtain the curvature product (a single estimated curvature) at each point. The maxima of the absolute curvature products along the smoothed curve are then obtained
TABLE I

<table>
<thead>
<tr>
<th>Properties</th>
<th>DoG</th>
<th>scale-DoG</th>
<th>ARCSS</th>
<th>CPDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corners in original images</td>
<td>586</td>
<td>82</td>
<td>30</td>
<td>41</td>
</tr>
<tr>
<td>Corner detection time (seconds)</td>
<td>2.71</td>
<td>0.19</td>
<td>0.17</td>
<td>2.6</td>
</tr>
<tr>
<td>Average repeatability (%)</td>
<td>65.36</td>
<td>70.47</td>
<td>66.40</td>
<td>71.04</td>
</tr>
<tr>
<td>Localization error (pixels)</td>
<td>1.24</td>
<td>1.46</td>
<td>1.21</td>
<td>1.15</td>
</tr>
<tr>
<td>Corners in (transformed) neighborhood</td>
<td>1.93</td>
<td>2.12</td>
<td>1.06</td>
<td>1.06</td>
</tr>
</tbody>
</table>

as candidate corners. Finally, it follows a two-step refinement process that removes weak and false corners using thresholds.

The experimental results in [15] showed that the CPDA detector outperformed many contour-based detectors including CSS [16], multi-scale CSS [17], and adaptive threshold-based CSS [18] detectors.

3) DoG Keypoint Detector: The DoG keypoint detector [2] first blurs the input image with a small width Gaussian filter to prevent aliasing and to increase the stability of the keypoints. It then builds the Gaussian pyramid by repeatedly smoothing the blurred image. The difference-of-gaussian (DoG) pyramid is built by subtracting the adjacent Gaussian smoothed images in the same octave of the Gaussian pyramid. Finally, they obtain the local extrema positions in the DoG pyramid as keypoints. When an extrema is found, two tests are applied before labeling it as a keypoint. First, it must have sufficient contrast and second, it should not be an edge point.

In one of our solutions, we scale down the original images before DoG keypoint detection. We name this modified detector as ‘scale-DoG detector’ in the rest of the paper.

4) Comparing Detectors: We used a database of 23 original images and their more than 8,500 test (transformed and signal processed) images to evaluate the performance of the detectors. In order to find repeated corners between the original and transformed images, we transformed the feature-points detected in the original image using the known transformation matrix and then found their repetitions with the feature-points detected in the test image. In this case, we allowed a mean-square-error (MSE) of $e$ square pixels, which means a feature-point in the original image was considered as repeated if at least one feature-point was found in its (transformed) neighborhood in the test image when the maximum distance was $e$ square pixels. The nearest feature-point in the neighborhood was considered as repeated. We evaluated the performance in terms of average repeatability and localization error.

We also calculated the number of detected feature-points in the (transformed) neighborhood. The detail of the aforementioned database and evaluation metrics can be found in [14], [15].

In the case of the scale-DoG detector, the original $(512 \times 512)$ images were down scaled to
Fig. 1. Comparing Detector’s Performance: (a) average repeatability, (b) localization error, and (c) number of feature-points in the neighborhood.
Fig. 1 shows the detail comparative performance and Table I presents the performance summary (average results). The scale-DoG detector offered the higher average repeatability than the original DoG detector. However, the scale-DoG detector offered the higher localization error with the increase of the neighborhood size. The ARCSS and CPDA detectors showed higher average repeatability than the DoG and scale-DoG detectors respectively when the neighborhood size was small. The opposite scenario was observed when the neighborhood size was large. The reason is, in the case of the DoG and scale-DoG detectors the detected keypoints are very close to each other. Fig. 1(c) shows that for these two detectors the number of keypoints in the (transformed) neighborhood is very high. Consequently, a close but different keypoint could be chosen as repeated offering high repeatability. Moreover, the DoG and scale-DoG detectors suffered from high localization error as shown in Fig. 1(b).

In Table I, we see that the ARCSS detector was the fastest and it detected the lowest number of feature-points among the four. On the other hand, the CPDA detector offered the highest average repeatability but the lowest localization error. However, it was slower than the ARCSS and scale-DoG detectors.

B. Feature Descriptors

We use SIFT descriptors [2] to represent DoG keypoints. Either the curvature descriptor or the SIFT descriptors at corner locations can be used to represent corners. We discussed SIFT descriptors in Section II-B2. Below we discuss how the curvature descriptors are formed.

The ARCSS and CPDA detectors make various information available for later applications. For each corner we get its position, absolute curvature value, angle with its two neighboring corners or with endpoints when the necessary number (two) of neighbor corners are not found, and affine-lengths between neighboring corners on the same curve. We will use all the above information collectively as the ‘curvature descriptor’ to represent the corner. As storing this information per corner takes only a few bytes of memory, the storage requirement for all the descriptors per image is very low in this approach.

C. Geometric Point Matching

The successful application of a feature detector in many applications depends on how to match feature-points between images. We proposed a simple feature-point matching algorithm based on the geometric invariance theory [26]. In order to estimate the affine transformation matrix between two images, where one is an affine-transformed version of another, we need at least three truly matching points. In this section, we first briefly describe our previously proposed GPM technique [11].
1) Iterative Matching Procedure: For given two sets of feature-descriptors of two images, the GPM technique works in following three steps:

- **Step 1:** We first find the candidate point matches by matching the feature descriptors between two images.
- **Step 2:** If three candidate matching points are non-collinear on each image and the ratio of areas of corresponding triangles in both the images is within a specific range (depending on the transformation range we want to consider), we estimate the transformation matrix between these triangles.
- **Step 3:** We transform all the corners in one image using the estimated transformation matrix and determine the number of point matches between the two images allowing a localization error of 3-pixels.

Steps 2 and 3 are repeated over all the combinations of three candidate matches found in Step 1 and track the combination that offers the highest number of point matches.

D. Proposed TII Approaches

We propose the following five approaches of TII using the above feature detectors, descriptors, and matching techniques.

1) **Approach 1: Matching ARCSS corners with Curvature Descriptors:** In this approach, we use the ARCSS corners as feature-points, the curvature descriptors to represent the corners, and the GPM for corner matching. In the first step of the GPM technique, the candidate matches are obtained by matching the curvature descriptors of the query image and a database image. Once we have the candidate point matching set, we apply steps 2 and 3 of the GPM technique to obtain the number of corner matches between the query and database images.

2) **Approach 2: Matching CPDA corners with Curvature Descriptors:** In this approach, we use the CPDA corners as feature-points, the curvature descriptors to represent the corners, and the GPM for corner matching. In the first step of the GPM technique, the candidate matches are obtained by matching the curvature descriptors of the query image and a database image. Once we have the candidate point matching set, we apply steps 2 and 3 of the GPM technique to obtain the number of corner matches between the query and database images.

The only difference of this approach from Approach 1 is, instead of ARCSS corners and curvatures we use the CPDA corners and curvatures.

3) **Approach 3: Matching Down Scaled DOG Keypoints with SIFT Descriptors:** In this approach, we use the scale-DoG keypoints as feature-points, the SIFT descriptors to represent the keypoints, and the GPM for keypoint matching.
In the usual SIFT [2], the DoG detector detects several hundreds to a few thousands of keypoints from a medium size image. Applying the GPM technique to such a huge number of keypoints will be very expensive. Moreover, since each descriptor of a keypoint is a 128-dimensional vector of floating point numbers, storing the descriptor information per image takes a few megabytes of memory, which in turn takes several gigabytes of memory for a moderate image database.

To reduce both the matching and storage costs, the images are down scaled before feature detection to reduce the number of keypoints. In this case, only the coarse scale features will be detected which are more robust than the fine scale features.

The candidate keypoint matches are obtained by matching the SIFT descriptors of the detected keypoints using the NNDR matching technique described in Section II-C.

Once we have the candidate point matching set, we apply steps 2 and 3 of the GPM technique to obtain the number of keypoint matches between the query and database images.

4) Approach 4: Matching ARCSS corners with SIFT Descriptors: In this approach, we use the ARCSS corners as feature-points, the SIFT descriptors to represent the corners, and the GPM for corner matching.

In spite of different measures to reduce the number of candidate matches in the first approach discussed above, we observed that the number of false candidate matches is still high which made the matching stage expensive. To overcome this problem, we use the highly distinctive SIFT descriptor, instead of the curvature descriptor, at each corner position while obtaining the candidate corner matching set. This strategy, though, increases the storage requirement, improves the matching performance highly in terms of both efficiency and effectiveness. Similar to the third approach discussed above, the candidate corner matching set is obtained by matching the SIFT descriptors using the NNDR matching technique.

Once we have the candidate point matching set, we apply steps 2 and 3 of the GPM technique to obtain the number of corner matches between the query and database images.

5) Approach 5: Matching CPDA corners with SIFT Descriptors: In this approach, we use the CPDA corners as feature-points, the SIFT descriptors to represent the corners, and the GPM for corner matching.

The only difference of this approach from Approach 4 is, instead of using the ARCSS corners and curvatures we use the CPDA corners and curvatures.

IV. PERFORMANCE STUDY

We implemented the five proposed approaches on a machine with the following configuration: Dual Core AMD Opteron(tm) Processor (265 × 2), 4GB RAM, Linux 2.6.18 kernel (Debian Distribution).
We used a large database and compared the proposed approaches with the existing most popular SIFT approach [2]. We present the results using the precision-recall graph [3].

A. Database

We randomly selected 1,050 images from David Nister’s recognition database.¹ Each image was of size 480 × 640 and converted to gray-scale. Then, for each of the image we applied following 10 simple transformations:

(i) rotation-crop: \( \theta = 30^\circ \),
(ii) scale: \( s_x = 1.2, s_y = 0.8 \),
(iii) rotation-scale: \( \theta = 20^\circ, s_x = 1.2, s_y = 0.8 \),
(iv) shear: \( s_{hx} = s_{hy} = 0.012 \),
(v) rotation-scale-shear: \( \theta = 10^\circ, s_x = 1.2, s_y = 0.9, s_{hx} = s_{hy} = 0.01 \),
(vi) jpeg: \( \text{quality factor} = 20 \),
(vii) Gaussian noise: \( \text{mean} = 0, \text{variance} = 0.001 \),
(viii) Gaussian blurring: \( \text{sigma} = 3, \text{window} = 3 \times 3 \),
(ix) rotation-scale-jpeg: \( \theta = 20^\circ, s_x = 1.2, s_y = 0.8, \text{quality} = 20 \), and
(x) rotation-scale-shear-jpeg: \( \theta = 20^\circ, s_x = 1.2, s_y = 0.8, s_{hx} = s_{hy} = 0.01, \text{quality} = 20 \).

As a result, we had total 1,050 groups of images, each group having 10 relevant images, and in total 10,500 images in the database.

B. Evaluation Metrics

We used precision and recall [3] collectively to measure the identification performance. Recall measures the system capacity to retrieve the relevant images from the database. It is defined as the ratio between the number of retrieved relevant images \( r \) and the total number of relevant images \( T \) (group size) in the database:

\[
\text{Recall} = \frac{r}{T}. \tag{1}
\]

Precision measures the retrieval accuracy. It is defined as the ratio between \( r \) and the number of retrieved images \( R \):

\[
\text{Precision} = \frac{r}{R}. \tag{2}
\]

In practice, the performance of an information retrieval system is presented using the precision-recall graph, where the higher the precision at a given recall value the better the performance of the retrieval system [3].

¹http://www.vis.uky.edu/~dnister/
In order to calculate the precision and recall for a given query image using (1) and (2), we first rank all the database images based on the number of feature-point matches with the query image. We then consider the minimum top $R$ images as the retrieved images such that they include all the relevant images of the query image.

C. Approaches to be Compared

In our experiments, we considered seven different approaches including the five proposed ones as shown in Table II. Approaches 1 to 5 in Table II are those discussed in Section III-D. For the Approach 3, we down-scaled each original image of size $480 \times 640$ to $60 \times 80$. Approaches 6 and 7 were implemented to compare the proposed approaches with the existing schemes. Approach 6 is the usual SIFT [2] which used the DoG keypoints, SIFT descriptors, and NNDR matching. Approach 7 is similar to Approach 3, but we used the NNDR matching instead of the GPM technique to compare the performance of the GPM and NNDR matching techniques.

Note that we found applying the GPM technique to the usual SIFT (Approach 6) was computationally very expensive due to the large number of keypoints, where each keypoint was represented using a 128-dimensional SIFT feature-vector.

D. Experimental Results and Discussions

Fig. 2 shows the TII performance of different approaches. The graphs are averaged on 177 random queries. Due to high computational complexity of some of the approaches, we could not consider more queries. We present and discuss our experimental results in the following sub-sections.

1) Scale-DoG vs Usual DoG Keypoint Detectors: Approach 7 performed worse than the usual SIFT (Approach 6) for the following two reasons: (i) in Approach 7, though the number of feature-points is decreased to speed up both the detection and matching stages, the number of matching
Fig. 2. Transformed image identification performance by different approaches.

Fig. 3. Image matching (object recognition) in arbitrary affine transformation and occlusion by Approach 1 (ARCSS detector, curvature descriptor, and GPM technique). The detected corners inside the white square of the left image were given as features of the object we want to recognize. The corresponding matched corners (total 19) are connected with lines and shown inside the white quadrangle of the right image.
features using the NNDR matching is also decreased; and \( (ii) \) in contrast, the main reason behind the success of the usual SIFT is, it detects a lot of features and a large number of them are found matched between the relevant images. The performance was greatly increased by using the GPM matching (Approach 3) instead of NNDR matching (Approach 7). This shows the superior performance of the GPM technique compared with the NNDR matching.

2) **Corner-Curvature vs Keypoint-SIFT Combinations:** Approach 1 required lower storage and total computational costs, but showed slightly higher identification accuracy than the Approach 6. While the former used the curvature descriptor with the GPM technique, the latter used the SIFT descriptors with the NNDR matching technique. Approach 2 offered higher identification accuracy than Approach 1, but required the higher computational cost. This is because the number of CPDA corners was higher than the ARCSS corners and, thus, there might be more false candidate matches in Approach 2 than in Approach 1. By using the top corners we could overcome this problem.

However, Approach 6 outperformed Approach 1 at higher recall values as shown in Fig. 2. The reason is, as the average number of detected corners was low using the ARCSS detector (Approaches 1), some irrelevant images (usually with higher number of detected corners than the relevant images) may also be retrieved. The performance was increased at higher recall values by increasing the number of corners, as evident from the performance of Approach 2 (using CPDA corners). However, this made the matching procedure expensive.

The SIFT descriptors are well known for object recognition [2]. The high accuracy of Approaches 1 and 2 indicates that they should be suitable for object recognition too. As an example, Fig. 3 shows an image matching (object recognition) by Approach 1. Similar performance was found by Approach 2. The specific part of ‘Fun house’ in the left image is recognized in the right image where it is arbitrary affine transformed and occluded.

3) **Corner-Curvature vs Corner-SIFT Combinations:** When we used the SIFT descriptors to represent the corners the matching cost reduced drastically as the number of false candidate matches decreased considerably due to high distinctive nature of the SIFT descriptor [2] (see Approaches 4 and 5).

4) **Summary of Results:** Table III\(^2\) summarizes the performance of different approaches by comparing the average number of detected features per image; storage requirement for the descriptors; average feature detection and representation time (in sec.) per image; average feature matching time per image pair; total time for detection, representation, and matching; and retrieval accuracy (precision) which

\(^2\) Features = number of features per image; Det.&Rep. time = feature detection and representation time per image; Mat. time = time to match a pair of images (a query image and a database image); all times are in second; Accuracy is the average precision over all the recall values shown in Fig. 2; total number of features and DR time were averaged over the whole database discussed in Section IV-A; and Mat. time was averaged over 177 random queries.
TABLE III
COMPARING DIFFERENT TRANSFORMED IMAGE IDENTIFICATION APPROACHES OF TABLE II.  

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Features</th>
<th>Storage</th>
<th>Det.&amp;Rep. time</th>
<th>Mat. time</th>
<th>Total time</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approach 1</td>
<td>46</td>
<td>low</td>
<td>0.19</td>
<td>4.330</td>
<td>4.52</td>
<td>94.8</td>
</tr>
<tr>
<td>Approach 2</td>
<td>58</td>
<td>low</td>
<td>2.65</td>
<td>5.36</td>
<td>8.01</td>
<td>97.4</td>
</tr>
<tr>
<td>Approach 3</td>
<td>37</td>
<td>moderate</td>
<td>0.54</td>
<td>0.004</td>
<td>0.544</td>
<td>97.9</td>
</tr>
<tr>
<td>Approach 4</td>
<td>71</td>
<td>moderate</td>
<td>0.19</td>
<td>0.045</td>
<td>0.235</td>
<td>98.0</td>
</tr>
<tr>
<td>Approach 5</td>
<td>90</td>
<td>moderate</td>
<td>2.65</td>
<td>0.17</td>
<td>2.82</td>
<td>99.1</td>
</tr>
<tr>
<td>Approach 6</td>
<td>1644</td>
<td>high</td>
<td>3.45</td>
<td>3.380</td>
<td>6.83</td>
<td>94.2</td>
</tr>
<tr>
<td>Approach 7</td>
<td>37</td>
<td>moderate</td>
<td>0.54</td>
<td>0.002</td>
<td>0.542</td>
<td>90.9</td>
</tr>
</tbody>
</table>

was averaged over all the recall values in Fig. 2. Note that we used the machine code provided by D. Lowe3 for feature detection and representation stages of Approaches 3, 6, and 7. While finding the SIFT descriptors at a corner location for Approaches 4 (ARCSS detector) and 5 (CPDA detector), we considered all the SIFT descriptors (detected by the usual SIFT) within the 3-pixel neighborhood around the corner in question. Therefore, the number of features was increased in Approaches 4 and 5 compared to Approaches 1 and 2 respectively. Since we were unable to differentiate between keypoint detection and representation times using the original SIFT’s machine code, we could not add the time of corner representation for Approaches 4 and 5. However, we believe that the SIFT feature representation time is much lower than its keypoint detection time.

5) The Best TII Approaches: As shown in Table III, Approach 5 performed the best in terms of identification accuracy, followed by Approaches 3 and 4. However, Approach 5 was much slower than Approaches 3 and 5 due to its slow corner detector. Approach 6 required very high storage for descriptors and its all the stages were quite expensive. The identification accuracies by Approaches 1 and 2 were slightly better than the usual SIFT. Moreover, Approaches 1 and 2 required much lower storage than Approach 6. The storage requirement, total time, and the identification accuracy of Approach 6 were significantly improved by the application of the proposed first feature-point reduction solution (see Approach 3). The total time and the identification accuracy of Approaches 4 and 5 (second solution) were also considerably improved by using the distinctive SIFT descriptors at the corner locations, though the storage requirement was increased moderately. This proves the high distinctiveness of the SIFT descriptor over the curvature descriptor.

Therefore, by combining the advantages of corners, SIFT descriptors, and the GPM technique, Approaches 3, 4, and 5 can be considered as the overall best.

3http://www.cs.ubc.ca/~lowe/keypoints/
Fig. 4. Transformed image retrieval examples by Approaches 1, 3, 5, and 6. Approach 4 performed the same (no irrelevant images were retrieved) as Approach 1 shown in (a) above. And Approach 2 performed the same (no irrelevant images were retrieved) as Approach 5 shown in (c) above. The number of detected features in the query image by these Approaches were 19, 37, 66, and 488 respectively. The roman number (inside parentheses) under each image indicates a particular transformation discussed in Section IV-A and the number after it is the number of feature matches with the query image in (a).
E. Transformed Image Identification Examples

Fig. 4 presents a transformed image retrieval example by Approaches 1, 3, 5, and 6. Approach 4 performed the same (no irrelevant images were retrieved) as Approach 1 as shown in (a) above. And Approach 2 performed the same (no irrelevant images were retrieved) as Approach 5 as shown in (c) above. In this example, while Approach 6 retrieved 16 and Approach 3 retrieved 2 irrelevant images, Approaches 1 and 5 retrieved no irrelevant images. The top 10 retrieved images are shown in Fig. 4 for each Approach, where 3 irrelevant retrieved images by Approach 6 and 2 by Approach 3 are also shown.

V. Conclusion

We have shown that both the feature detection and matching stages of TII can be significantly speeded up by reducing the number of detected feature-points. We have presented two feature-point reduction solutions.

The first solution is to down scale the image before the DoG keypoint detection. We have found that this solution significantly increases the TII performance when the scale-DoG keypoints are matched using the GPM technique, which finds more repeated features than the existing NNDR matching technique with the expense of little more computational time. However, this additional expense can be considered negligible because of high improvement in the identification accuracy (comparing accuracy of Approach 3 with that of Approaches 6 and 7 in Table III).

The second solution is to use corners instead of DoG keypoints. Either the curvature descriptor or the highly distinctive SIFT descriptors at corner locations can be used to represent corners. The corner-curvature combinations, though require higher matching time, perform better than the keypoint-SIFT combinations (comparing performance of Approaches 1 and 2 with that of Approaches 6 and 7 in Table III). However, when the corners are represented using the SIFT descriptors (corner-SIFT combinations), not only the matching time is significantly reduced, but also the TII accuracy (robustness) is improved (comparing performance of Approaches 4 and 5 with that of Approaches 1 and 2 in Table III).

Therefore, the two feature-point reduction solutions combined with the SIFT descriptors and the GPM technique not only improve the computational efficiency and decrease the storage requirement, but also improve the TII accuracy.

REFERENCES


