Rotation Invariant Indexing of Shapes and Line Drawings

Michail Vlachos
IBM Watson Research Center

Zografoula Vagena
UC Riverside

Philip S. Yu
IBM Watson Research Center

Vassilis Athitsos
Boston University

Abstract

We present data representations, distance measures and organizational structures for fast and efficient retrieval of similar shapes in image databases. Using the Hough Transform we extract shape signatures that correspond to important features of an image. The new shape descriptor is robust against line discontinuities and takes into consideration not only the shape boundaries, but also the content inside the object perimeter. The object signatures are eventually projected into a space the renders them invariant to translation, scaling and rotation. In order to provide support for real-time query-by-content, we also introduce an index structure that hierarchically organizes compressed versions of the extracted object signatures. In this manner we can achieve a significant performance boost for multimedia retrieval. Our experiments suggest that by exploiting the proposed framework, similarity search in a database of 100,000 images would require under 1 sec, using an off-the-shelf personal computer.

Categories and Subject Descriptors

H.3 [**Information Systems**]: Information Storage and Retrieval

General Terms

Algorithms, Performance

Keywords

Hough Transform, Image Signature, Metric Tree

1. INTRODUCTION

Recognition of objects or shapes in images and video is a fundamental problem in computer vision encompassing a multitude of applications in medicine (tumor shape recognition) [10], manufacturing (crack identification), multimedia search (shape taxonomy grouping) [11], trademark recognition [9], surveillance (weapon detection), etc.

Effective retrieval of similar shapes in image databases is a complex and time consuming process. In most cases it represents a compromise between accuracy and expected response time. Direct comparison of images is inefficient not only because of the high raw data dimensionality, but also because shape transformations (e.g., due to different viewpoints) can deteriorate the matching process. In order to guarantee high quality matching one has to deploy shape descriptors/signatures that can adequately capture shape characteristics, and are typically tailored to be immune to transformations such as translation, scaling or rotation.

While the efforts of the pattern recognition community are primarily targeted on the accuracy of results with the use of complex similarity measures, there has been little work on expediting the search procedure. Older systems for content-based image retrieval include the QBIC system [4], while more recent implementations for generic multimedia retrieval (images and video) can be found in MARVEL [1]. However, these efforts primarily address the issue of feature selection (such as color, shape, texture and position) rather than the organizational mechanisms of the object signatures.

In this work we are examining the problem of shape retrieval not only from a machine learning perspective (i.e. devising robust shape descriptors), but also from a database viewpoint, providing additionally an efficient *hierarchical indexing* scheme for achieving a fast search mechanism. We accomplish this by intelligently organizing/clustering the object signatures, so that the search procedure is directed toward the most promising object 'clusters'.

The shape signatures that we extract are based on the Hough Transform, which captures important line features of an image. The descriptor offers enhanced robustness against line discontinuities and considers not only the object perimeter, but also its interior content. However, similarity search on the signatures using sequential scan, would significantly hamper the system performance, making it difficult to scale to large datasets. Therefore, we additionally demonstrate how to create the necessary mechanisms that can render search into a real-time process, even using a regular personal computer, on a database containing thousands of images. To achieve this: (i) we exploit the Fourier Transform properties for providing a rotation invariant image signature, (ii) we provide compressed representations of the signatures and also a lower-bounding distance measure which *guarantees* that no qualifying signatures will be discarded, (iii) we present a *hierarchical index* structure which holds the compressed signatures and in conjunction with the lower bounding function can offer real-time search capabilities.

To our best knowledge this is first work that seamlessly integrates all the above components in order to offer a ro-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CIKM'05, October 31–November 5, 2005, Bremen, Germany.
Copyright 2005 ACM 1-59593-140-6/05/0010 ...$5.00.

bust and real-time shape search mechanism. An additional advantage of the proposed framework is its flexibility and adaptability, since it can be used in conjunction with other shape signatures (perimeter or angular based), without any modification, therefore easing the porting of our system in other domains and applications. We envision that this work can provide the data-mining practitioners with a variety of tools that can ease and expedite expensive tasks, such as image clustering and visualization.

2. PROBLEM AND METHODOLOGY

In this work we are interested in providing fast and efficient recognition/classification of shapes or objects in images, given a large database of (possibly annotated) examples. We can achieve this by realizing 3-levels of abstraction for data organization (figure 1). The lowest and most rudimentary level will contain the raw images. The middle level will store image descriptors in the form of signatures that are invariant to various transformations. The upper level will provide the signature organization, and will facilitate an efficient pruning scheme.

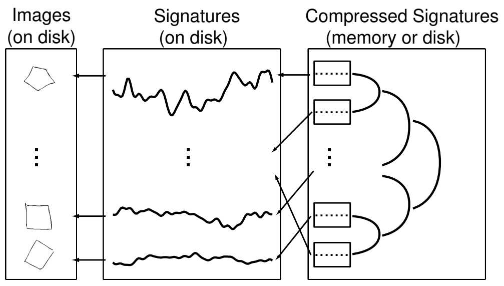


Figure 1: 3-tier hierarchy for efficient image search

The highest level will hold *compressed* versions of the image signatures, with the purpose of providing distance approximations and faster distance calculations. The compressed vectors will also be hierarchically organized, leading to an efficient pruning scheme of signatures (and images) that are *guaranteed* to be very distant to the query image.

3. EXTRACTING SHAPE SIGNATURES

We utilize the Hough Transform (HT) for extracting important features from an image. HT based algorithms can be used for detection of any parametric shape (lines, circles, planes, etc.) and have found applications in many areas such as human iris recognition or fingerprint matching [16]. In this work we are interested in recognizing contours of shapes or line drawings (such as technical drawings [14], symbols, trademarks [9], etc.) therefore we utilize the HT for detecting the basic line structures in an image, a method that is very robust to noise and line discontinuities. An additional advantage of the HT based signature that will be extracted, is the fact that it also takes into consideration the content inside the perimeter of the object, unlike perimeter based methods which only consider the silhouette. In this paper we present a way of *compressing* the information of the Hough Transform and extracting a *rotation invariant* signature. However, the signature transformation and the indexing framework that will be presented in this paper is generic enough to support other shape signatures, such as angular signatures [14]. Due to limited space we provide just a short overview of the HT.

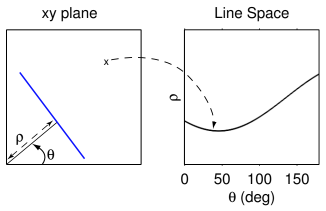


Figure 2: A line in the x-y plane is mapped to a point in the ρ-θ space. A point x (since infinite lines pass through it) is mapped onto a set of ρ and θ pairs, forming a sinusoid in the Hough space.

The HT performs a mapping from the x-y space onto a ρ-θ space, where the ρ, θ parameters represent solutions of the line equation $xcos\theta + ysin\theta = \rho$ (left side, figure 2). Parameter θ records the angle of a segment with one end on the axis origin and the other one perpendicular to the line, while ρ captures the distance of the respective segment. Using the HT each point of the original image is mapped onto a sinusoid in the new ρ-θ space, or 'votes' for the corresponding sinusoid positions in an accumulator array R. After all points have 'voted' on the array R, the local maxima of R correspond to the dominant lines of the image.

Using the entire accumulator $R_{\rho,\theta}$ as an image signature is space prohibitive. In this work we introduce a way of compressing the accumulator by summing the values of R across the θ axis (i.e., summing the columns of the array). In this manner we extract a signature s from every image.

$$s(\theta) = \sum_{p=1}^{n} R^2_{\rho,\theta}, \quad \theta = 0 \dots 180^o$$

The resulting sequence can be normalized by division with its average value. In [5] Fränti et al. also utilize the HT to extract shape descriptors, but retain only the dominant peaks of the HT by thresholding the accumulator array, hence introducing an additional (and not easy to set) parameter. In our setting, all the values of array R are utilized, making our system completely parameterless.

By summing along the parameter ρ we have rendered the shape descriptor robust to translation and scale, since we just record the angles of the lines and not their position. This type of compression however also increases the probability of different shapes having the same signature. This will introduce false alarms during the matching procedure, which can be eliminated in a post-processing phase by looking at the full array R.

We should also note that rotation of an image by *theta* degrees does not change the shape of the signature, but only shifts it (left or right) along the θ axis by an equal amount. In figure 3 we depict examples of signatures extracted from various shapes on grayscale images. The final shape signatures are sequences of length 180, since the Hough Transform records line angles in the range of $0 - 180^o$. In the same figure, we notice that similar shapes produce similar signatures (shifted nonetheless), irrespective of the shape orientation.

4. DISTANCE MEASURE

Suppose that $x = (x_0, x_1, \dots, x_{N-1})$ and $y = (y_0, y_1, \dots, y_{N-1})$ are the signatures extracted from two images. We need to define a distance function that will quantify the degree of similarity. The L_2-Norm should be sufficient to capture such similarities.

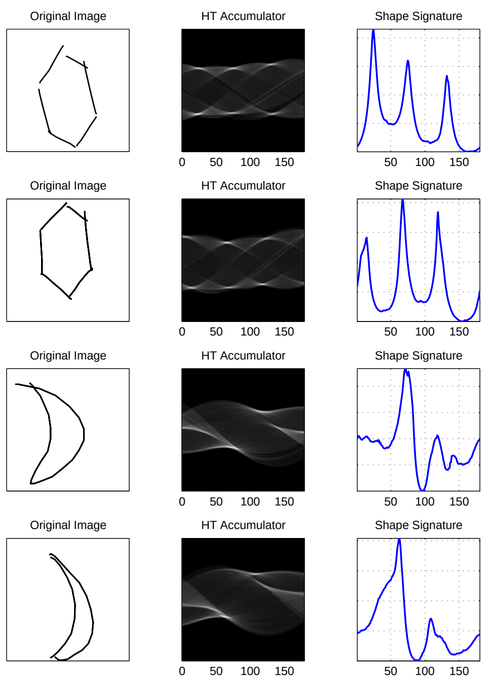


Figure 3: For left to right: image, Hough accumulator, shape signature by vertical summation of accumulator

$$D(x, y) = \|x - y\|$$

However, since we desire our measure to be rotation invariant, we also need to perform a *circular rotation* of one signature along the other and record the minimum distance between the signatures. The rotation invariant distance measure D_{rot} is defined as:

$$D_{rot}(x, y) = \min_{i=0...N-1} D(x, C_i(y)),$$

where C_i is the circular rotation operator. Computation of D_{rot} requires time in the order of $O(N^2)$, which can render retrieval times too slow for many practical applications.

In the following sections, we present a more efficient way of approximating the D_{rot} function. We can avoid the costly execution of the circular rotation operation by exploiting the Fourier Transform properties and rendering the shape signature rotation invariant.

4.1 Magnitude Based Distance

We utilize the normalized Fourier Transform (FT) for providing a rotation invariant signature. The FT of a sequence x is a sequence of complex numbers and is defined as:

$$X(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi kn}{N}}, \quad k = 0, 1 \dots N-1$$

The image signature x is a sequence of real numbers, so each Fourier coefficient $X(k)$ will also have a symmetric in the form of a complex conjugate. The FT essentially records the amplitude and phase of the complex sinusoids $s_f(n) = \frac{e^{j2\pi fn/N}}{\sqrt{N}}$, which represents a lossless decomposition of the original signal. Since we want to provide a measure that is immune to circular rotations it is sufficient to record the *magnitude* of the coefficients and ignore the phase. The magnitude signature is a vector of real and positive numbers. We can also reduce the space in half by recording only the magnitude of the first half coefficients, due to the symmetric property:

$$\mathbf{X} = \|X(k)\|, \quad k = 0, 1 \dots \lceil \tfrac{N-1}{2} \rceil$$

Now, we define the new rotation invariant distance between two signatures x and y, as the L_2-Norm between the magnitude vectors:

$$D(\mathbf{X}, \mathbf{Y}) = \|\mathbf{X} - \mathbf{Y}\|$$

The extraction of the magnitude signatures from the images can be performed offline and therefore the computation of the new rotation invariant distance function requires now time only in the order of $O(N)$ (where N is is the magnitude signature length). These new signatures will be stored on disk and will represent the second level of our 3-tier hierarchy, as described in figure 1. Next we will explain how to create the compressed signatures that will populate the highest level of the database hierarchy.

4.2 Compression and Lower Bound Distance Approximation

We will provide a more compact representation of the magnitude vector and also an approximation of the distance function, between the compressed vectors.

In the magnitude vector not all coefficients are equally important; those with the highest value correspond to the sinusoids that carry most of the signature energy [15]. For any signature $\mathbf{X}$, we will store only the highest k magnitude coefficients, where parameter k depends on the desired compression factor. Let us denote as p^+ a vector indicating the positions that hold the largest k values of signature $\mathbf{X}$ having length N (so $p^+ \subset [1 \dots N]$), and p^- contains the indices of the omitted ones. We also record some information about the omitted magnitude coefficients, which will assist us in providing more tight distance estimation when utilizing the compressed vectors. The additional information will be the square root of the sum of squares of their values ($\epsilon_{\mathbf{X}}$), that represents a measure of the approximation error. The error $\epsilon_{\mathbf{X}}$ is essentially the length of the vector containing the discarded magnitude coefficients:

$$\epsilon_{\mathbf{X}} = \sqrt{\sum_{i \in p^-} \mathbf{X}(i)^2} = \|\mathbf{X}(p^-)\|$$

Therefore, the compressed signature of every image will contain $k+1$ numbers, the k largest coefficients and one additional error value, having the form $[\mathbf{X}(p^+), \epsilon_{\mathbf{X}}]$. Now suppose the user provides a new query image and wants to retrieve the image most similar to it. The query magnitude signature $\mathbf{Q}$ will be calculated, which is an uncompressed magnitude vector. Next, we provide a *lower bound* approximation of the magnitude distance between the original uncompressed vectors $\mathbf{X}$ and $\mathbf{Q}$.

Let us denote as $\mathbf{Q}(p^+)$ the vector holding the equivalent coefficients as $\mathbf{X}(p^+)$. Similarly, $\mathbf{Q}(p^-)$ is the vector holding the analogous elements of $\mathbf{X}(p^-)$.

Example: Suppose that the magnitude vector of x is: $\mathbf{X} = (5, 8, 2, 10, 1, 0, 3, 2)$. If we decide to record just the two best coefficients then $p^+ = (2, 4)$ and therefore $\mathbf{X}(p^+) = (0, 8, 0, 10, 0, 0, 0, 0)$. Now suppose the user presents a query q with magnitude signature $\mathbf{Q} = (8, 2, 10, 5, 2, 2, 1, 3)$. Then we have: $\mathbf{Q}(p^+) = (0, 2, 0, 5, 0, 0, 0, 0)$ which holds the equivalent coefficients as $\mathbf{X}(p^+)$. (Note that the zeros have been placed here just for clarity. In the actual implementation we just need to store the position of the coefficient and its value.)

We can rewrite the squared distance between the compressed vectors:

$$\begin{aligned} D(\mathbf{X}, \mathbf{Q})^2 &= D(\mathbf{X}(p^+), \mathbf{Q}(p^+))^2 + D(\mathbf{X}(p^-), \mathbf{Q}(p^-))^2 \\ &= \|\mathbf{X}(p^+) - \mathbf{Q}(p^+)\|^2 + \|\mathbf{X}(p^-) - \mathbf{Q}(p^-)\|^2 \end{aligned}$$

The left part of the distance calculation can be easily performed since we have all the required data. The right part cannot be computed exactly because we lack the $\mathbf{X}(p^-)$ portion. We will provide an approximation that *always* underestimates the true distance between the original uncompressed signatures. It holds that:

$$\begin{aligned} \|\mathbf{X}(p^-) - \mathbf{Q}(p^-)\|^2 = \|\mathbf{X}(p^-)\|^2 &+ \|\mathbf{Q}(p^-)\|^2 \\ &- 2\langle \mathbf{X}(p^-), \mathbf{Q}(p^-)\rangle \end{aligned}$$

where $\langle \cdot, \cdot \rangle$ is the inner product between two vectors.

However, using the Cauchy-Bunyakovski-Schwarz (CBS) inequality, which dictates that $|\langle a, b\rangle| \leq \|a\| \cdot \|b\|$, we get:

$$\begin{aligned} \|\mathbf{X}(p^-) - \mathbf{Q}(p^-)\|^2 &\geq \|\mathbf{X}(p^-)\|^2 + \|\mathbf{Q}(p^-)\|^2 \\ &\quad - 2\|\mathbf{X}(p^-)\| \cdot \|\mathbf{Q}(p^-)\| \\ &= (\|\mathbf{X}(p^-)\| - \|\mathbf{Q}(p^-)\|)^2 \\ &= (\epsilon_{\mathbf{X}} - \epsilon_{\mathbf{Q}})^2 \end{aligned}$$

and therefore we can utilize the following lower bounding function LB between the compressed magnitude vectors:

$$\begin{aligned} LB(\mathbf{X}, \mathbf{Q}) &= \sqrt{\|\mathbf{X}(p^+) - \mathbf{Q}(p^+)\|^2 + (\epsilon_{\mathbf{X}} - \epsilon_{\mathbf{Q}})^2} \\ &\leq D(\mathbf{X}, \mathbf{Q}) \end{aligned}$$

Essentially, we managed to offer an underestimate of the distance between the original image signatures using their compressed counterparts.

4.3 Utility of Lower Bound function

Let us examine for a moment the usefulness of the lower bounding function. Suppose we only had the uncompressed magnitude vectors and we would like to find the signature that is closest to a query signature $\mathbf{Q}$. The search procedure in this case would proceed by retrieving one-by-one the signatures and comparing them to the query. The image having the smallest signature distance will be returned as the nearest match. Therefore, in this scenario *all* the image signatures would have to be retrieved from disk. This approach is typically called *Linear Scan* (LS) of the data.

When we utilize the compressed signatures, we can avoid examining a portion of the uncompressed shape signatures. In this case we cannot provide exact distances, but we can estimate lower bound approximations. Suppose that we compute the lower bound distance between the query and all compressed vectors. We start retrieving the uncompressed signatures in the order dictated by the lower bound estimates (from smaller to larger distance). The true distance between the query and the currently retrieved uncompressed vector is calculated and the *best-so-far* match is potentially updated. However, when we reach an object with lower-bound distance *greater* than the *best-so-far* distance, the search can be safely terminated, since we are *guaranteed* that all remaining objects have lower bound (and hence true distance) larger than the current best match. If the distance function did not underestimate the actual distance, we could not guarantee the outcome of the results.

Another advantage of this method is that distance computations are much faster using the compressed signatures because of their compact size, meaning that they can typically fit in main memory, enhancing even more the performance.

Using a hierarchical indexing scheme (which will be delineated in the following section), we can also prune from evaluation a large number of compressed signatures, if they are guaranteed to be very distant to the query and as result further improve the efficiency of the search method.

5. INDEXING STRUCTURE

For organizing hierarchically the compressed object signatures, we construct a new index structure which adopts ideas from the family of metric trees. Metric structures provide a clustering of the objects based on the relative distance between the object signatures. Our choice of metric indices is guided by our adaptive compression technique. Since every signature uses a different set of magnitude coefficients (the ones with the highest energy per object), indexing using data-partitioning methods, such as R-trees (which would require the same sets of coefficients for each object signature) would have been impossible. Moreover, the superiority of metric trees against R-Trees has been empirically demonstrated in [6].

In this work we introduce a variation of a well studied metric tree, the Vantage-Point tree (or VP-tree) [6]. The VP-tree exploits the object distances to preselected reference objects (called vantage points) in conjunction with the triangle inequality, in order to prune parts of the search space that are guaranteed to contain objects more distant than the currently discovered best matches. Therefore, the search strategies direct the search toward the most 'promising' partitions, namely the ones that have high probability to contain result candidates.

However, the original structure of a VP-tree uses uncompressed objects as vantage points which simplifies distance calculations, but leads to large tree structures. In this work we present the VPC-structure (where 'C' stands for 'Compressed'), that uses compressed objects as vantage points, leading to reduced index sizes that can be entirely kept in memory, hence leading to a significant performance boost. However, because of its different structure, the VPC-tree employs a different search procedure and pruning methodology than the original vantage-point tree. We continue by presenting the traditional VP-tree and then we explain its differences from the VPC structure.

Every node in the VP-tree (or VPC-tree) is associated with a set of points S and a pivotal or *vantage point* v, which is used to divide all points in that node into two equal sets. After the distances between all node points and the vantage point are sorted, the points with distance less than the median distance μ are placed in the left subtree $S_{\leq}$, while the remaining ones in the right subtree $S_{>}$. This concept is illustrated in Figure 4a. The index tree structure is constructed by recursively performing this operation for all subtrees (figure 4b). The leaf nodes of our index contain the actual compressed signatures, as well as a pointer to their uncompressed version, which resides on disk.

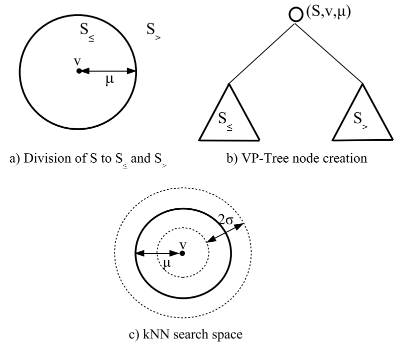


Figure 4: Vantage-Point structure

In order to create a vantage point tree one needs to provide a method for selecting a vantage point for each tree node. As such reference points, are chosen the objects that provide high deviation of distances to the remaining objects in the node, which is an analogous concept to the largest eigenvector in SVD decomposition.

KNN Search in VP-trees. After its construction, the index structure can be used to efficiently answer k-NN queries. In summary, the search is directed to the most promising parts of the data space, potentially pruning objects or clusters of objects from examination. The pruning strategy works as follows; suppose that σ is the distance between the query and its closest match discovered so far and $d(q, v)$ is the distance between the query and vantage point of the currently examined tree node. Using the triangle inequality, it can be shown that the subset $S_\leq$ does not have any candidate nearest neighbor points if $d(q, v) > (\mu+\sigma)$ and therefore the subset can be excluded (pruned) from the search process. Similarly, the subset $S_>$ can be pruned from the search if $d(q, v) \leq (\mu - \sigma)$ (figure 4c). Both subtrees would have to be visited only in the case when $\mu - \sigma < d(q, v) < \mu + \sigma$.

Using those results, the search for the nearest neighbor of a query q proceeds as follows: The threshold σ is set to a large value and the distance $d(q, v)$ between the query point q and the *vantage point* v associated with the root node of the VP-tree is computed. The distance determines which child nodes need to be searched and which can be safely pruned. The same procedure is recursively repeated for each node that is searched, until the leaf nodes are reached where the actual distances of the data from the query point are computed. If at some point a better nearest neighbor is identified, σ is reduced to new distance value and the search resumes with this new value.

KNN search in the VPC-tree. For our shape recognition application, the index uses the compressed magnitude vectors for the vantage points as well as for the leaf data. We will demonstrate in the experimental section that this results in an index structure whose size is a very small fraction of the original data size.

The new index structure necessitates several modifications in both the construction and search phases. During the construction phase, the uncompressed signatures of the shapes are utilized in order to identify the vantage points and compute the median distances. Subsequently, only the compressed version of each object is stored in the index.

During the search phase, we take into consideration the fact that the index maintains only the compressed magnitude vectors, when traversing the structure. As a result, at each step, in order to check whether a visited data sequence is the current nearest neighbor, we first compute the lower bound of the distance between the query sequence and the currently visited compressed representation of the data sequence. If the lower bound is larger than the distance between the query and the current nearest neighbor, then as we have already described in section 4.3, we can safely discard this data sequence. Otherwise, we fetch the actual object signature from disk and we compute the real distance. When the visited data sequence is a vantage point, the discussion as of how to identify the subspaces to search next that we have provided in the previous section, holds only if we load the uncompressed signature from disk. Otherwise, we can only check whether we can discard the subset $S_\leq$, while we have not enough information for the $S_>$ subset. The subset $S_\leq$ can be safely discarded if $LB(q, v) > (\mu + \sigma)$, where LB is the lower bound of the periodic distance between the compressed vantage point and the query point.

```
NNSearch(Q) { /* Input: Uncompressed Signature Query */
  cNN.ID = NULL; cNN.distance = INF;
  Search(ROOT, Q, cNN);
}

Search(NODE, Q, cNN) {
  /* Parameter: Node of VP-tree, Uncompressed Query
                Signature Q, Current Nearest Neighbor */

  /* queue = priority queue of compressed
     shape signatures sorted on LB */
  if NODE.isLeaf {
    for each compressed time-series cT in node {
      LB <- computeLowerBound(cT,Q);
      queue.push(cT,LB); /* sorted by LB */
    }
    while ((!queue.empty()) &&
              (queue.top().LB < cNN.distance)) {
      if (cNN.distance > queue.top().LB) {
        retrieve uncompressed T of queue.top() from disk;
        dist = D(T,Q); /* full distance */
        if dist < cNN.distance {
          cNN.distance = dist; cNN.ID = T;
        }
      }
    }
  } else { /* vantage point */
    LB <- computeLowerBound(VP,Q);
    queue.push(VP,LB);
    if LB < (node.median + cNN.distance) {
      search(NODE.left, Q, cNN);
    }
    search(NODE.right, Q, cNN);
  }
}
```

Figure 5: Indexed Retrieval of Shape Signatures

In figure 5 we provide the pseudo-code for the search algorithm. For simplicity we only include the case where we do not need to load the disk-resident, uncompressed representation of the vantage point. If we have already loaded the uncompressed magnitude vector, the method is identical with the search within the original VP-Tree node, as this is described in [6].

The algorithm invokes the `computeLowerBound(cT,Q)` method, which receives as parameters the compressed representation of a data point and the uncompressed signature of the query point and computes the lower bound of their distances, using the method described in section 4.2. Moreover, it utilizes a priority queue maintaining the compressed signatures of the visited data points, along with the lower bound of their distance. The priorities in the queue are defined by those lower bounds. That way, the most promising data points, i.e. the ones which the smaller lower bounds are visited first.

Due to its hierarchical structure and its small footprint, the VPC-tree is able to provide very fast response rates and exceptional pruning power. This will be demonstrated in more detail in the experimental section.

6. EXPERIMENTS

We provide comprehensive experiments that show the high matching quality of the proposed shape signatures and distance measures. Additionally we indicate the low latency of the indexing scheme. The experiments utilize two datasets; the first contains 160 shapes, with 9 classes of objects (bone, hand, rabbit, etc), which are obtained from various sources (`Mixed-Bag` dataset). The second dataset comes from a symbol recognition database called `HHreco` [1] (Figure 7). The shapes used in our experiments are: *'cube'*, *'square'*, *'pentagon'*, *'hexagon'*, *'parallelogram'*, *'trapezoid'*, *'triangle'*, *'moon'*. These come at different positions, scales and orientations according to the individual writer's style. The database contains a total of 7791 strokes, collected from 19 users. The second dataset (because of its larger size) was also used in order to test the performance of the indexing scheme.

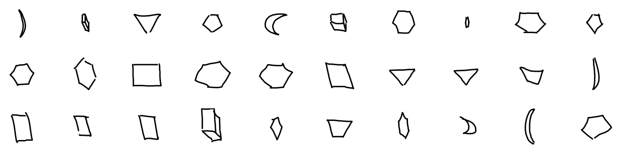


Figure 7: A sample of the HHRECO dataset of symbol strokes used in our experiments

For both datasets, the only preprocessing we did before applying the Hough transform, is to perform an edge detection. After this operation using the proposed algorithms we extracted the shape signature, uncompressed magnitude signature and also its compressed version.

All techniques have been implemented in Java and executed on a desktop computer equipped with a Pentium 4, 2.6GHz processor, 512MB of main memory and a 40GB IDE disk. In our experimental setting the index structure was small enough to be held entirely in main memory, while the uncompressed signatures are disk resident.

6.1 Accuracy and Quality of results

In Figure 6 we demonstrate a small set of results that depict the high perceptual similarity of matches returned by our system for various image queries posed on both datasets. From the retrieved images, it is evident that the extracted shape signatures are robust to transformations such as translation, scale, rotation and even deformation.

We also compare the accuracy of our system against two of the most widely used image matching algorithms in the vision literature, the Hausdorff [8] and the Chamfer [2] distance functions. Both of these measures operate directly on the image space and have complexity of $O(R * ElogE)$, where E is the number of edge pixels and R is the number of rotations that need to being executed (assuming the images have been aligned to have same center and scaling factor). In our experiments R=45. Comparatively our approach has $O(n)$ complexity, with n the length of the compressed Hough signature (where $n << E$).

[1]http://www-cad.eecs.berkeley.edu/Respep/Research/hhreco/

In Table 1 we report the retrieval accuracy from 1-NN to 5-NN for the 3 measures. The accuracy is measured using a leave-one-out classification experiment, where we remove each image from the dataset and then retrieve its k-Nearest-Neighbors ($k = 1 \ldots 5$) from the remaining shapes of the database. The ratio of the number of shapes belonging in the same class as the query image out of the k images, defines the level of accuracy. The presented distance measures on the compressed Hough signatures achieve a 91.25% accuracy for the 1-NN on `Mixed-Bag` dataset, and 83% for the `HHreco` database. The competing distances report slightly higher accuracy (around 2-6%) but this comes at significantly more prolonged execution time. In total, the results of the compressed Hough signatures are very encouraging, achieving high precision/recall, while providing a rotation invariant approach at low computational cost. In future work we plan to explore how these k-NN results could be further refined using more expensive measures, such as the Hausdorff or Chamfer distance (in effect, using a multi-measure hierarchy).

Dataset	Method	1-NN	2-NN	3-NN	4-NN	5-NN
`Mixed-Bag`	HT-Signature	0.91	0.88	0.87	0.84	0.80
	Hausdorff	0.93	0.92	0.91	0.90	0.89
	Chamfer	0.94	0.94	0.93	0.92	0.91
`HHRECO`	HT-Signature	0.83	0.81	0.80	0.79	0.79
	Hausdorff	0.86	0.86	0.86	0.85	0.85
	Chamfer	0.87	0.87	0.86	0.86	0.86

Table 1: System recognition accuracy from 1-NN to 5-NN

6.2 Index Pruning Power

In this section we focus on the index performance. To evaluate the effectiveness of the proposed measure in conjunction with the VPC-tree, we provide a thorough comparison of three methods:

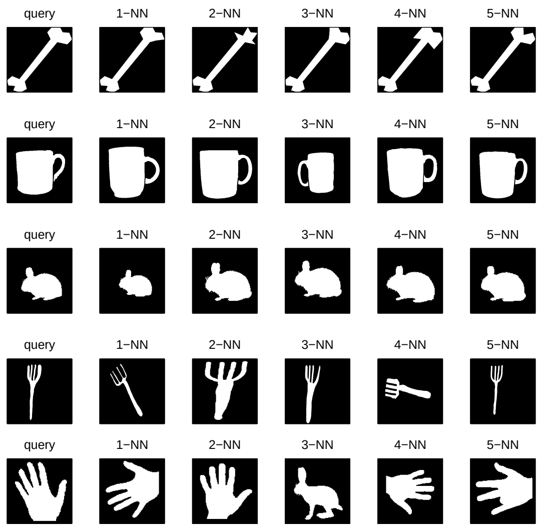
(i) The linear scan of the shape signatures using the brute-force D_{rot} measure

(ii)The linear scan of the uncompressed magnitude vectors, and

(iii) The VPC-tree performance using the compressed HT signatures and the lower bounding distance function described in Section 4.2.

We count the number of the uncompressed magnitude vectors that need to be loaded from secondary storage when the index structure is employed to answer a k-NN query. The experiment is repeated for different values of k, namely 5, 10 and 20 and varying dimensionalities (i.e., number of compressed signature coefficients). The reported results are average values over 100 queries and they are presented in Figure 8(a). It is evident that our system is not only able to retrieve high quality matches, but it also achieves it in a very efficient way, pruning a large number of signatures that do not participate in the final k results.

As expected, the larger the dimensionality of the compressed signatures, the better the pruning power of the index. That is because the estimated lower bounds for the distance values are tighter when more coefficients are preserved. At dimensionality of 16, where the index presents the best pruning power, only 1% of the total uncompressed magnitude signatures are visited by the index. It should be noted, that linear scan methods (which don't employ an index), would have to examine all disk resident signatures. Therefore, our technique demonstrates excellent pruning power of



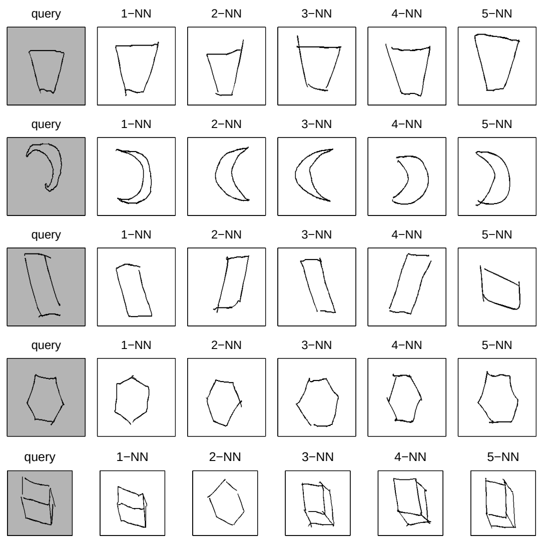


Figure 6: 5-Nearest Neighbors for queries posed on our two datasets. The last row depicts examples of incorrect matches. We observe that rotated versions of the query shape can be effectively recognized.

the data space. These results are also very promising regarding the scalability of our method, as only a very small number of object signatures have to be touched per query.

6.3 Index Response Time and Speedup

Even though the number of retrieved disk-resident signatures is one measure of the index efficiency, a more realistic indicator is the time needed for returning an answer to a k-NN query. In this third experiment we provide a comparison of the time required for each of the three methods under consideration to process and return the results of a k-NN search for 100 queries. The outcome is presented in Figures 8(b) and 8(c). Again the VPC-tree depicts exceptional performance, being more than 40 times faster than the linear scan of the magnitude vectors and up to 300 times more efficient than the linear scan of the shape signatures (using the D_{rot} distance measure). In Table 2 the actual index response times are reported. We observe that the VPC-tree can return the closest matches to a query in ˜50-70msec depending on the signature dimensionality and the number of Nearest-Neighbors that are requested.

	D=2	*D=4*	*D=8*	*D=16*
5-NN	66.59	45.37	45.88	**42.60**
10-NN	75.03	62.47	**49.61**	54.46
20-NN	65.72	59.36	**54.34**	65.32

Table 2: Average time (in msec) for returning the k-NN results in a database of 7791 image signatures. Results shown for different number of Nearest Neighbor searches (*k-NN*) and compressed signature dimensionalities (*D*).

One thing to note here is that for 10-NN and 20-NN searches, the index exhibits a faster response rate for dimensionality of 8, even though more uncompressed signatures are retrieved from disk compared to dimensionality of 16. These results hint on the existence of an upper threshold for the number of coefficients that have to be recorded and clearly indicate that there are diminishing returns by increasing the number of the magnitude coefficients in the index. This outcome shows that signature approximation using 16 coefficients, does not yield a significantly improved lower bound approximation of distances (in comparison to 8 coefficients). Therefore, the overhead of performing the additional magnitude computations surpasses the gains achieved by the tighter distance estimations.

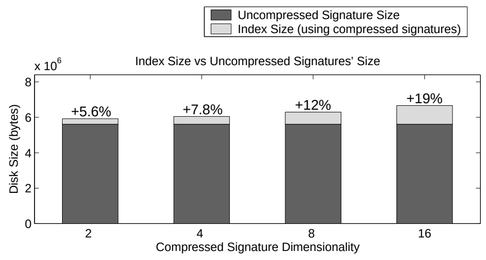


Figure 9: Additional disk size required for storing the index.

Index Size: The VPC-tree index structure holds the compressed signatures, and any additional organizational overhead (vantage points, medians, pointers, etc.), however its size is still a small portion of the uncompressed image signatures. To illustrate this, in Figure 9, we present the total size (data signatures + index structure) occupied by the proposed architecture. The optimal index performance is observed at compressed signature dimensionality of 8. As illustrated in the figure, for this dimensionality, the index incurs a disk size overhead of only 12%. Nonetheless, the speedup achieved by this modest increase exceeds 40 times, compared to the performance of the Linear Scan. Those results justify the strategies proposed in this paper, i.e. to trade some additional disk space, in order to achieve considerably better performance and scalability.

7. RELATED WORK

A significant body of work in the field of computer vision and machine learning has dealt with the problem of shape

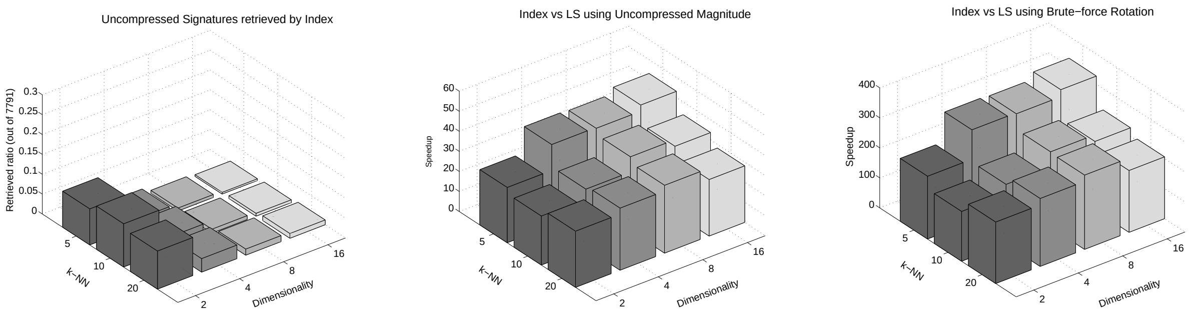


Figure 8: Index performance for various compressed magnitude vector dimensionalities (D=2,4,8,16) and Nearest Neighbor searches (k=5,10,20). *(a)*: Percentage of uncompressed signatures retrieved from disk by the index. *(b)*: Index Speedup compared to time required when performing a sequential scan of the uncompressed magnitude vectors. *(c)*: Speedup offered by index, in comparison to the time required for finding k-NN matches using the original (non-magnitude) signatures and performing a cyclic-rotation distance calculation.

recognition. This section does not intend to be complete in any way, but rather hint on the variety of distance measures that have been used for shape recognition, such as approximate Earth Movers Distance on shape contours [7], Dynamic Time Warping of object boundaries [12], bipartite matching between shape features [3], etc. Most of these approaches completely lack an indexing framework, while some of the distance measures are non-metric and therefore are bound to present false dismissals when combined with an indexing scheme. In [13] Rafiei et. al., present a complete indexing framework for object boundaries using a fixed set of Fourier descriptors in conjunction with R-trees. Besides the different shape signature, our work enjoys the following two advantages: i) the signature compression is significantly more efficient, because it considers a different set of Fourier descriptors per object (the ones holding the highest energy), ii) the metric based tree structure that is used for storing the object descriptors has been shown to be more efficient than R-trees, especially for higher data dimensionalities [6].

A great discriminant of the present work to previous ones is that the HT based signature can easily accommodate for shape discontinuities and takes into account not only the shape boundaries but also the content inside the perimeter, allowing for effective shape matching, even when highly compressed.

8. CONCLUSION

We have presented a complete framework for recognition of shapes and line drawings in image databases and we have compared the proposed image signature and distance measure with other widely used image matching functions (Hausdorff and Chamfer). The contributions of this paper are on different levels; first we have presented an elegant way to extract a rotation invariant image signature using the Hough Transform and then we have rendered the distance calculation between signatures efficient using properties of the Fourier Transform. On the indexing level, we have presented a compressed metric tree variant, coupled with an effective lower bounding scheme, which demonstrates exceptional pruning power and very low latency.

While the focus of this work was primarily on providing real-time matching of similar shapes, accuracy was notably also very high. In future work, we intend to improve even more on the system accuracy and precision, by using our system as an initial prefiltering step for other more expensive matching measures.

9. REFERENCES

[1] Marvel: Multimedia analysis and retrieval system. http://www.research.ibm.com/marvel/.

[2] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *IJCAI*, 1977.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *PAMI*, 24(4), 2002.

[4] M. Flickner and H. Sawhney. Query by Image and video content: The QBIC system. In *IEEE Computer*, 1995.

[5] P. Fränti, A. Mednonogov, V. Kyrki, and H. Kälviäinen. Content-based matching of line-drawing images using the Hough transform. In *IJDAR(3), No. 2*, 2000.

[6] A. Fu, P. Chan, Y.-L. Cheung, and Y. S. Moon. Dynamic VP-Tree Indexing for N-Nearest Neighbor Search Given Pair-Wise Distances. *The VLDB Journal*, 2000.

[7] K. Grauman and T. Darrell. Fast contour matching using approximate earth movers distance. In *CVPR*, 2004.

[8] D. Huttenlocher, D. Klanderman, and A. Rucklige. Comparing images using the Hausdorff distance. *IEEE PAMI*, 15(9):850–863, 1993.

[9] Y.-S. Kim and W.-Y. Kim. Content-based trademark retrieval system using visually salient features. In *Proc. of CVPR*, pages 307–312, 1997.

[10] P. Korn, N. Sidiropoulos, C. Faloutsos, E. Siegel, and Z. Protopapas. Fast and Effective Retrieval of Medical Tumor Shapes. In *IEEE TKDE, 10:6*, pages 889–904, 1998.

[11] C.-L. Lee and S.-Y. Chen. Classification for Leaf Images. In *Proc. of IPPR CVGIP*, 2003.

[12] E. G. M. Petrakis, A. Diplaros, and E. Milios. Matching and retrieval of distorted and occluded shapes using dynamic programming. *IEEE PAMI*, 24(11), 2002.

[13] D. Rafiei and A. Mendelzon. Efficient retrieval of similar shapes. In *The VLDB Journal 11*, pages 17–27, 2002.

[14] S. Tabbone, L. Wendling, and K. Tombre. Matching of graphical symbols in line-drawing images using angular signature information. In *IJDAR(6), No. 2*, 2003.

[15] M. Vlachos, C. Meek, Z. Vagena, and D. Gunopulos. Identification of Similarities, Periodicities & Bursts for Online Search Queries. In *Proc. of SIGMOD*, 2004.

[16] W. Zorski, B. Foxon, J. Blackledge, and M. Turner. Fingerprint and iris identification method based on the hough transform. In *Proc. of Imaging and Digital Image Processing*, pages 69–81, 2000.