Low-Complexity Arrays of Contour Signatures for Exact Shape Retrieval

Florian Lardeux, Sylvain Marchand, Petra Gomez-Krämer

Pattern Recognition 118 (2021) 108000

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Low-complexity arrays of contour signatures for exact shape retrieval

Florian Lardeux*, Sylvain Marchand, Petra Gomez-Krämer

L3i laboratory, Faculté des Sciences et Technologies, La Rochelle University, Bâtiment Pascal Avenue Michel Crépeau, La Rochelle Cedex 1 17042, France

ARTICLE INFO

Article history:
Received 23 May 2020
Revised 24 March 2021
Accepted 18 April 2021
Available online 1 May 2021

Keywords:
Shape recognition
Contour signature
Fast retrieval
Associative arrays

ABSTRACT

We propose a framework for a fast exact shape retrieval called Low-complexity Arrays of Contour Signatures. The purposes are to match a shape against a database in constant time and to retrieve correct shapes very close to the query, while the latter may have undergone rigid transformations and noise. We present a shape signature based on prior works as well as a compact characterization of such signatures, a system of associative arrays allowing a short search time for retrieval and a technique of pairwise alignment. This method shows a good resilience to perturbations and is performed in a constant computational time.

1. Introduction

Shape recognition, classification, retrieval or even just building tools for shape analysis can still be a challenge nowadays, as the exigencies of technological efforts continuously increase. There are many ways to formulate goals within this field: classifying shapes visually resembling each other, finding matching parts of shapes, perform a matching that is invariant or robust to transformations, etc. A significant part of the literature focuses on shape classification considering occlusions, perturbations due to noise, scaling, rotation and even sometimes large deformations. Depending on the application, the intra-class variance in a dataset is sometimes such that it can be considered relevant to admit that some shapes in the same class are not the same. Therefore a large part of the existing state of the art cannot discriminate visually similar but differently labeled shapes, as they admit large amounts of deformations. Also, most of the current literature revolves around the use of direct descriptor comparisons, which can lead to high computational times for retrieval.

In this paper, we decide to tackle the problem of exact shape retrieval in a large database, in constant computational time.

For instance, these shapes can be contours extracted from quasi-flat objects such as ancient coins [1]. Ancient coin databases can be very large, i.e. the Online Coins of the Roman Empire (OCRE) database[1] totals over 100,000 specimens. The objective is to retrieve only coins that are produced by the same matrix. This is a problem of exact shape retrieval different from typical shape retrieval approaches, which would retrieve all coins of the same type.

Hence, two shapes are considered here to be the same if one can be mapped onto the other by applying a rigid transformation or noise, as presented in Fig. 1.

The intuition for this work was given by a conjunction of ideas from both the audio field and the shape recognition community. The work of [2] is a very simple and compact way of describing the shape by its curvature. However, the cross-correlation is not robust and quite slow. Wang et al. introduced in [3] the Shazam system for musical audio recognition, where retrieval is performed using a hash map to efficiently store feature pairs of the signal as well as a binning technique to ensure the alignment between similar parts. We therefore investigate this idea in the context of shape recognition, without the need for a hash key for the creation of the feature pairs. Dealing with small deformations involves nonetheless the use of a robust signature. Since the cumulative sum [2] accumulates noise along the signature, this representation is not at all robust to perturbations. Our signature is a modified version of the Triangle Centroid Distances (TCD) [4].

This article involves the following contributions: 1) A signature based on a modified version of the TCD [4], which makes a simple yet very robust 1D representation of the shape. 2) A selection of feature points and feature pairs extracted from the signature. 3) An efficient storage system via associative arrays. 4) A general algorithm of constant time retrieval using pairwise geometric alignment. We call this whole method Low-complexity Arrays of Contour Signatures, shortened in LACS.

The use of associative arrays for exact shape matching resembles geometric hashing. However, geometric hashing was originally

* Corresponding author.
E-mail address: florian.lardeux1@univ-lr.fr (F. Lardeux).

[1] http://numismatics.org/ocre/.

https://doi.org/10.1016/j.patcog.2021.108000
0031-3203/© 2021 Elsevier Ltd. All rights reserved.

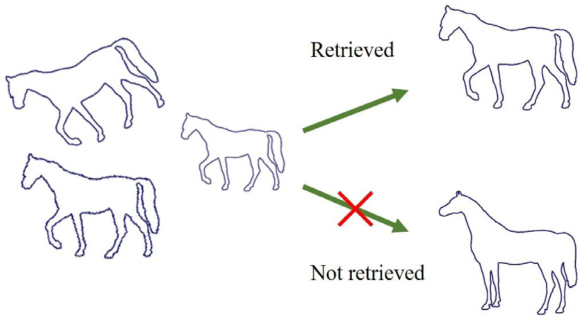


Fig. 1. Exact shape retrieval. A shape is retrieved if and only if there is a rigid transformation or noise between it and the query. If the shape seems visually to belong to the same class, but is in fact geometrically different, it will not be retrieved.

developed in computer vision for matching point features [5], such as vertices of shapes, and not shape signatures.

Related works on shape recognition are listed in Section 2. The definition of the signature is presented in Section 3. In Section 4, the feature points and the feature pairs are described and the process of storage to form a database is described in Section 5. Section 6 explains the mechanism of pairwise geometric alignment to ensure geometric consistency in the retrieval. Finally, experiments on robustness and runtime are presented in Section 7. A conclusion and discussion of our work are outlined in Section 8.

2. Related work

There are two main perspectives to extract information: region based and contour based. Region-based methods, such as Fourier descriptors [6] or image moments [7], provide a global shape description by computing values related to the shape as a whole. However, these techniques suffer from the global descriptive data since they can analyze only parts of the shape.

Contour-based methods are widely considered in the literature, because they are invariant to lighting conditions and do not take into account colors or textures [8]. The main strategy to characterize the contour is the use of a shape descriptor, which can characterize globally or locally the contour. The corner stone of shape similarity techniques is the shape context descriptor [9], which locally features the distribution of contour points by the means of radial histograms. Although it allows to deal with scaling, rotation and deformations, the similarity cost is rather heavy and cannot cope with partial occlusions. Many works are based on this technique and propose the use of Dynamic Programming to match points [10], a hierarchical strategy to deal with affine and projective transformations [11] or the use of the earth mover's distance to perform comparisons [12]. Chang et al. [13] propose a descriptor for open and closed shapes based on the Triangle-Area representation. Their work efficiently deal with scaling by combining several scales and can tackle occlusions. Similar properties can be found in [14,15], which propose multiscale descriptors based on invariant aspects of shapes. They also demonstrate invariance to articulated deformation and intra-class variations. Wang et al. [16] introduce another type of descriptor relying on height functions. The works of [17] and [8] use a descriptor which can efficiently match parts of contours. The general method uses a fixed sampling of the contour from which they extract angles between chords within the shape. Yang et al. introduced the Triangle Cendroid Distance (TCD) descriptor [4], based on distances between each point of the shape and Fourier descriptors. They achieve fine results in both part-to-whole and part-to-part matching while dealing with significant non-rigid deformations. However, their matching technique is not invariant to the position of the starting point and therefore needs as many comparisons per shape as there are points in it. The work of [18] also propose the use of Fourier descriptors with a multiscale technique using group features, which uses combination of spatial features to improve matching accuracy. Finally, to refine shape description, the work of [19] combines shape contour and skeleton by analyzing the geometry of contour parts in relation with their associated skeleton.

Alternatively, several other works use signature-based representations to describe the shape, which synthesize the contour information into a compact representation. Commonly used signatures are, for instance, centroid distances, the chord length signature, the area function, the cumulative angular signature and the turning angle signature. Curvature is a paramount element in human shape recognition [20]. Cui et al. [2] formulate a scale-invariant signature based on this principle. The abscissa is calculated as the integral of unsigned curvatures. This mathematically shows to be indeed scale-invariant in theory but in practice fails to take into account the changes in sampling of the contour when perturbations are applied to the image itself. To compare two shapes, they use normalized cross-correlation. In [21], a Distance Interior Ratio signature is introduced. It is invariant to rotation and scaling and yield promising retrieval results on popular datasets.

Various frameworks can be formulated for the matching step, depending on what the desired goal is. Research works describing novel methods for shape representation generally adopt distances or similarity measures such as the edit distance [10], the earth mover's distance [22], the χ^2 distance [9], or dissimilarity costs [4,17]. In any of the aforementioned cases, pairwise comparisons must be performed if one desires to find the best match among a database of shapes which leads to a linear complexity when performing retrieval. Several authors propose alternatives to standard comparisons that can involve space search reduction via statistical tools [23], or the use of Bayesian inference [24]. However, the speed gain is achieved for each comparison, and not for the database as a whole, which is still a limiting factor for a fast retrieval. Other works rely on learning to perform classification. The classic method of Bag-of-Words model is used by [25–27]. Bai et al. [28] extended this principle by using both local information and global properties of the shape to gain discriminative power for the recognition. The same idea of contour segments is investigated in other works such as [29] which uses Bayesian inference for learning. Learning methodologies can be very effective for shape classification, but require a specific training set, which cannot account for large databases of very different shapes.

3. The shape as a signature

Our choice is to convert the shape into a 1D signature so as to get a compact representation which contains relevant information of the shape. Attneave [20] showed in 1954 that the human mind seems to privilege a very small, but really precise quantity of data, gathered in its boundary curvature.

In order to comply with this psychological background, a natural choice is to study signatures which feature this kind of attributes. Most naive signatures [30] are not invariant to scale and features such as curvature also suffer heavily from noise [2]. Namely, the main issue with the computation of a good signature is its stability under perturbations.

3.1. Description of the signature

We modify the TCD descriptor designed by [4] and extract a signature from it.

The TCD is based on the Fourier transform of distances between points of the shape. Namely, if the shape has n points, for each pair of points P_i and P_j, $j \neq i$, let g_{ij} be the barycenter between P_i, P_j and the total center of gravity of the shape G. Then, the distance

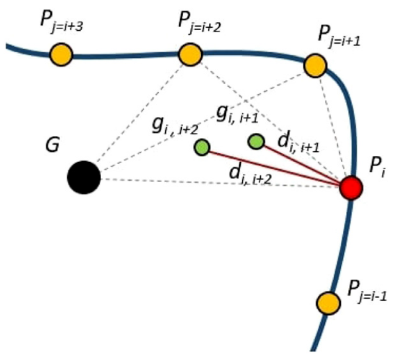


Fig. 2. Graphical explanation of the TCD. d_{ij} is the distance between P_i (red) and g_{ij} (green). G is the center of gravity of the whole shape. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

d_{ij} is defined as the distance between P_i and g_{ij}. Fig. 2 provides a illustration of this calculation.

Each distance is normalized by the maximum of each row, giving Eq. (1).

$$\forall (i,j) \in [0, n-1] \times [0, n-2], \;\; \hat{d_{ij}} = \frac{d_{ij}}{\max_i d_{ij}} \tag{1}$$

Finally, a Fourier transform is applied for each column of (d_{ij}) for 16 frequencies, from which the absolute value is taken, which leads to the final descriptor $(f_{wj}) \in \mathcal{M}_{16,n-1}(\mathbb{R}^+)$ in which each element is given by Eq. (2).

$$\forall j = 0, \ldots, n-2, \quad \forall \omega = 0, \ldots, 15$$

$$f_{\omega j} = \left| \frac{1}{(N-1) \cdot \max_n d_{nj}} \cdot \sum_{n \neq j} d_{nj} \cdot \exp\left(-\frac{2i\pi n\omega}{N-1} \right) \right| \tag{2}$$

where $i^2 = -1$ and $|.|$ is the absolute value.

The difference in description compared to the original TCD is that we consider the d_{nj} to be the direct norm of the vector $\overrightarrow{P_n P_j}$. This formulation is much simpler and yields better results because it deals with exact positioning. Our principal signature, which is sufficient for a large number of cases, is the case of zero frequency in Eq. (2), and is simply the ratio of the average and the maximum of the d_{nj}.

$$f_j = \frac{1}{(N-1) \cdot \max_n d_{nj}} \cdot \sum_{n \neq j} d_{nj} = \frac{\overline{d_{nj}}}{\max_n d_{nj}} \tag{3}$$

This signature is robust to rigid transformations as well as noise and characterizes the shape. As explained below in Section 5.3, one can nonetheless use at least the first few rows of (f_{wj}) to improve the robustness. It should be noted that each shape is read forward and backward, since there is a priori no way to select a stable direction. The query shape will thus have two signatures to compare.

3.2. Behavior of the signature

This simple formulation is very efficient as a signature since it is, in theory, invariant to shift, scale, rotation, starting point (resulting in a shift in abscissa) and very robust to noise, being the low frequency of the distance signal. Here we focus on the formulation presented in Eq. (3). Three types of behavior regarding transformations are cited below: adding noise the to shape, scaling the source and rotating the source (changing the starting point). For illustration, the first shape of the Kimia99 dataset [31], displayed in Fig. 4, is used. All original images have an average resolution of around 120 × 120. A binary blob is extracted from the shape and the contour is extracted as a contiguous sequence of pixels.

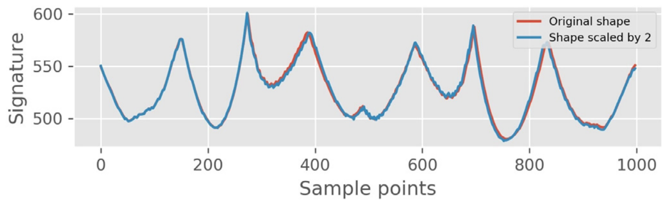


Fig. 3. Signatures obtained with two different scalings of the shape. There is a slight difference, but overall the signatures are almost identical.

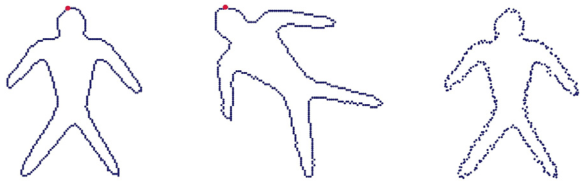


Fig. 4. Left and middle: Impact of rotation on the starting point (red dot). The shape here is rotated by $\frac{\pi}{4}$. Right: Shape with Gaussian noise of standard deviation 0.5. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

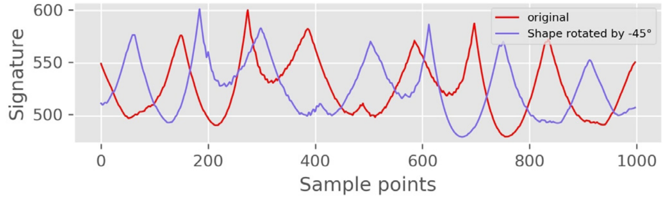


Fig. 5. Impact of rotation on the signature. The signature shifts by the same amount as the shape curvilinear abscissa.

3.2.1. Scaling the image

Scaling is applied to the image prior to the shape extraction and the calculation of its signature. The larger the image, the more precise the shape details. Fig. 3 presents the two versions of the signature of the same shape having undergone a scaling by 2. It is paramount that the signature is as precise as possible so that its extrema stay very stable (see Section 4). The work in [4] uses a uniform interpolation of 100 points whereas we choose 1000 to gain precision.

3.2.2. Rotating the image

Rotation changes the location of the starting point within a closed shape. Given a binary blob image from which the contour is extracted, there is no way to determine robustly which point to start the contour with, and therefore it changes with rotation. This is particularly visible in Fig. 4, where the red dot denotes the starting point of the shape. Fig. 5 reveals that the signature shifts accordingly. The only major issue with this phenomenon is that it can hide a potentially relevant feature point being located at the break point that would appear as a peak or a trough in the signature.

3.2.3. Adding noise to the shape

In order to simulate noise, we simply add a Gaussian noise to both coordinates of the shape pixels independently. Fig. 6 shows the impact of adding noise on the signature. A Gaussian noise of standard deviation 1.0 is rather strong for the size of this shape, therefore it can be considered as a limit case. The signature can be very corrupted by noise, the low frequency signal is used. This is considered to be a limit case, since we do not intend to deal with large deformations.

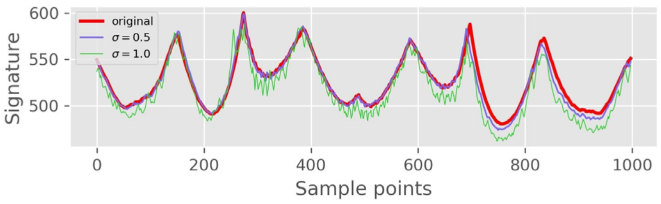


Fig. 6. Impact of noise on the signature. Noise in the shape translates in noise in the signature. Strong noise can significantly corrupt the signal and disturb the local extrema locations.

4. Feature points and feature pairs

To reduce both spatial and time complexities, feature points are extracted from the signature values and organized in pairs. That way, we ensure to keep the model very compact while maintaining a geometrical consistency.

4.1. Feature point selection

Feature points of our signature are defined as the points which exhibit a distinct variation in signature.

The feature points are chosen to be the local extrema of the signature of coordinates (x_j, f_j) such that:

$$P_{feature} = \left\{ (x_j, f_j) \mid f_j > f_{j-1} \text{ and } f_j > f_{j+1} \quad \text{or} \quad f_j < f_{j-1} \text{ and } f_j < f_{j+1} \right\} \quad (4)$$

Fig. 7 shows these feature points in our example in both the signature and the shape. A simple analysis of Eq. (3) reveals that extrema in the signature appear where points are the closest and the furthest from the rest of the shape. This may occur at various positions in the shape, but this set of feature points noticeably include high curvature points. However, they carry more information than the sole extrema of the curvature. This is at the heart of a relevant and compact representation of the shape for an efficient recognition; shapes that carry very little curvature extrema cannot be analyzed properly. Here, on the contrary, there is a nice distribution of peaks and troughs that enable a rich feature representation of the shape. Finally, by keeping only the extrema, a large part of the information carried by the other parts of the signature are eliminated. This pruning is relevant because we postulate that the informative content is gathered within the relative positions of these extrema.

4.2. Generating pairs of feature points

Out of the selected feature points, our idea is to generate feature pairs. The main argument behind such a choice is the following: points by themselves do not carry sufficient information; there is no way to compare two points if there is a shift between them. Pairs of points add structural information about the signature and the shape. Because the signature f is very robust to rigid deformations, the pairs will remain very stable.

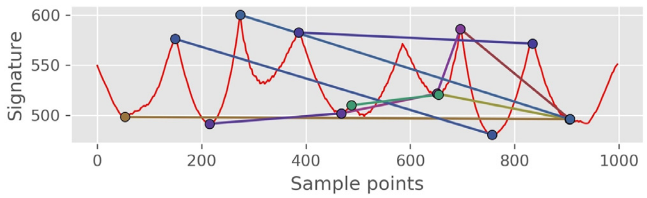


Fig. 8. Examples of pairs of feature points (10 are presented). Pairs add structural information to the positional information of the feature points.

The generation of pairs is done by reading the feature points in the signature from left to right. For each point, the algorithm looks ahead within a window of size m inside which the second point is selected. An example of such pairs can be found in Fig. 8. In practice, we choose $m = 20$.

To handle the case of the rotation of a closed shape and thus a shift of the signature, one needs to take into account the points in a circular manner. The list of n feature points is appended with the first m elements, making it a list of $n + m$ feature points. This way, when processing the last points of the list, we create pairs with the first feature points of the list, thus creating a continuity in the generation of pairs. In the same fashion, we calculate another list of feature pairs for the backward signature.

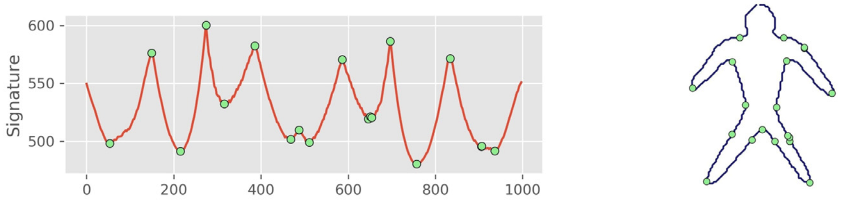
5. Storing data in associative arrays

We choose to use the pairs to store information in a system of associative arrays that serves as a tool for an efficient retrieval method (see Section 6). The main idea behind this form of storage is that one can store a whole database of feature pairs in it, each belonging to a certain shape. In essence, this is very close to the concept of hash tables. Some works have used hashing for shape recognition and clustering [32,33], or for finding unusual shapes [34]. However, our technique allows to retrieve shapes among a large database and does not require learning any clustering pattern. Although the hashing technique has also been used in the Shazam algorithm [3] by which our method was inspired, the difference is that we do not explicitly create hash functions *per se*, because the associative functions described below are not injective.

5.1. Associative array functions

An associative array is a set of buckets indexed by a key function and containing a certain value. Let $h = \{h_0, h_1, \ldots, h_{p-1}\}$ $(p \in \mathbb{N}^*)$ be the set of associative arrays. Each array does not have a fixed number of buckets. In the following, each array is assimilated to its key function so that $\forall i = 0, \ldots, p-2, \; h_i \in \mathbb{Z}^{\mathbb{R}}$.

Every time a pair is stored in h, the bucket indexed by the key might not exist and is therefore created in the process. In prac-



(a) Feature points in signature.

(b) Corresponding positions in the shape.

Fig. 7. Feature points in our example shape. Note that these extrema are closely related to the high curvature points, but give however far more information than a simple curvature signature.

```
Algorithm 1: Adding a pair to associative arrays.
Input    : $h$: associative arrays
           $\{a_0, a_1, \dots, a_{p-1}\}$ quantization parameters
           $\{d_0, d_1, \dots, d_{p-1}\}$: input descriptor
           $\{x_0, x_1, \dots, x_{p-1}\}$: list of abscissas of the first
 point of the query pairs
           $idx$: index of the shape
$L \leftarrow h_0$
for $i$ from 0 to $p-1$ do
    $\hat{d}_i = h_i(d_i)$
    if $i = p-1$ then
        if $L[\hat{d}_i]$ does not exist then
            $L[\hat{d}_i] \leftarrow$ empty list
        $L[\hat{d}_i] \xleftarrow[\text{append}]{} (x_i, idx)$
    else
        if $L[\hat{d}_i]$ does not exist then
            $L[\hat{d}_i] \leftarrow$ empty array
        $L \leftarrow L[\hat{d}_i]$
```

tice, we assign an index to the shape (idx in Algorithm 1), which can be of any form as long as it allows to identify the shape. For each pair, there is a descriptor $\mathcal{D} = \{d_0, d_1, \cdots, d_{p-1}\}$ with p components. Each key function takes as an input respectively each component of the descriptor $\mathcal{D}$. Let $\mathcal{A} = \{a_0, a_1, \dots, a_{p-1}\}$ be a set of p positive values. These values are quantization parameters of the key functions defined below. They virtually allow to increase or decrease the "size" of each bucket in the arrays. This way, one can manipulate those parameters in order to get a finer or coarser storage. Small perturbations can be taken into account by setting a sufficient coarseness. For our application and for any pair of extrema $\{(x_1, f_1), (x_2, f_2)\}$ defined in Eq. (4), we use a descriptor of three components defined by $\mathcal{D} = \{f_1, f_2, x_2 - x_1\}$ and a set of parameters $\{a_0, a_1, a_2\}$ such that $a_1 = 0.1$, $a_2 = 0.1$ and $a_3 = 0.5$. The values of the arrays are fetched by finding bins corresponding to $h_i(d_i) = \lfloor a_i \cdot d_i \rfloor$, $\forall i = 0, \dots, p-1$.

Because the signature is deemed sufficiently stable, one can discriminate the ordinates easily, hence the choice of the first two components. Moreover, the third component of $\mathcal{D}$ allows to neglect the impact of shift along the signature abscissa, thereby making the descriptor invariant to the starting point. The descriptors are thus very reliable, provided the signature f is sufficiently robust.

5.2. Structure

As evoked, h is a *system* of associative arrays. Each bucket (not empty) of the first array h_0 contains one independent array, which is indexed by the second key function h_1, and so on. This system of nested arrays is akin to a dense tree structure, in which there the retrieval is progressively refined. Fig. 9 explains graphically the structure of our system and presents a query example. In our case, we use three different keys and therefore the system has a depth of 3.

The last array buckets should contain the identification of the shape. We store a list of tuples consisting of the abscissa of the first point x_1 and the index of the shape idx. Therefore, each time a new entry is added to the whole array (Algorithm 1), it is possible to append its tuple (x_1, idx) to a list already containing previously added tuples. The associative array h_i does not handle collisions and the whole storage system allows to retrieve several potential shape indices for each query pair (Algorithm 3). Also, if no corresponding pair was found (which should often occur if the shape is unknown in the database), then the value None is returned.

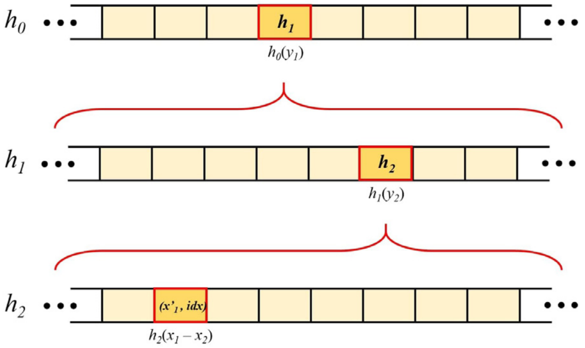


Fig. 9. Structure of our system of associative arrays. Each bin of each array, indexed by its key function, contains a sub-array. For a query pair descriptor $(f_1, f_2, x_1 - x_2)$, the result is found by successively finding the corresponding bin of the arrays, as long as it exists. The last array contains the result of the retrieval, which is the abscissa of the first point of the retrieved feature pair x'_1 initially stored (Algorithm 1) and the index of the corresponding shape idx.

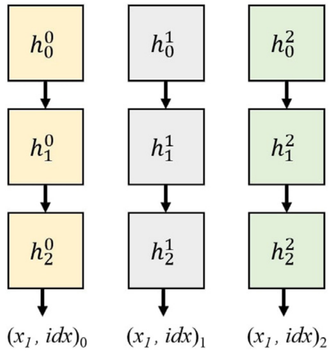


Fig. 10. Example of $\mathcal{H} = \{h^u_v \mid u \in [0, 2]\ \ v \in [0, 2]\}$. Each h^u is a system of associative arrays composed of v nested arrays. The retrieve values (x_1, idx) may not concur, as they are pulled from a different signature (even though they characterize the same shape, they are independently processed).

5.3. Parallel systems of arrays

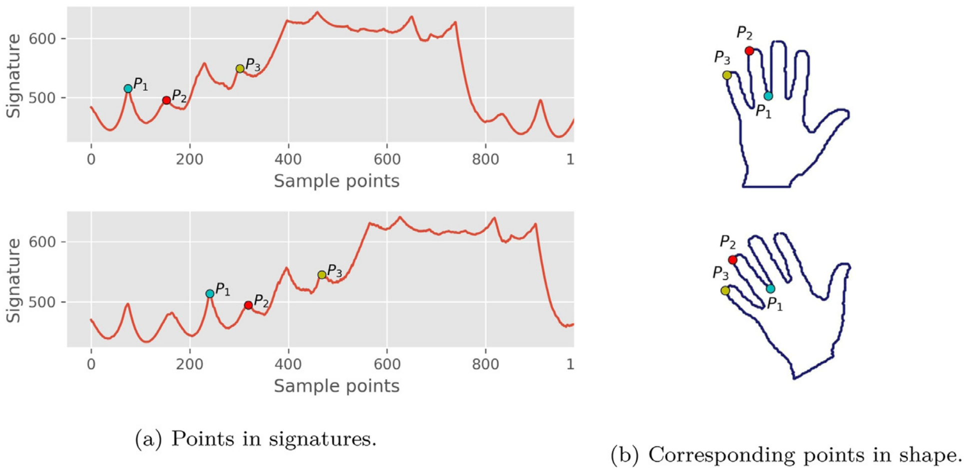
It is possible to use several systems of associative arrays at once. Namely, we can use several signatures, all extracted from the descriptor of [4] and processed separately, all in the same fashion, both for storage and retrieval. Let $\mathcal{H}$ be a system of associative arrays. The process of retrieval is to consider a set of signatures characterizing the shape at several levels of description $f_j(\omega = 0)$, $f_j(\omega = 1), \dots$ and perform retrieval on each pair of each signature. The complexity of retrieval increases linearly with the size of $\mathcal{H}$, but adding more systems of arrays leads to very robust results. An example of such set of arrays is presented in Fig. 10. In the following, we will continue to use only one array to explain the technique to avoid heavy notations.

6. Retrieval

This section explains the complete mechanism of retrieval. It consists of two general steps: finding the matches in the database and aligning them with a binning technique.

6.1. Pulling correspondences from h

The simple scenario to consider here is having a query $\mathcal{Q}$, which is a collection of pair descriptors for a query shape, and compar-



(a) Points in signatures.

(b) Corresponding points in shape.

Fig. 11. Example of the calculation of δx for a rotation of $\frac{\pi}{6}$. The given points P_1, P_2 and P_3 have been shifted due to rotation. Here $\delta x = 166$ in each case. Even if there were slight variations (bounded by +2.0 (excluded) with $a_3 = 0.5$), those values would still belong to same bin in the histogram.

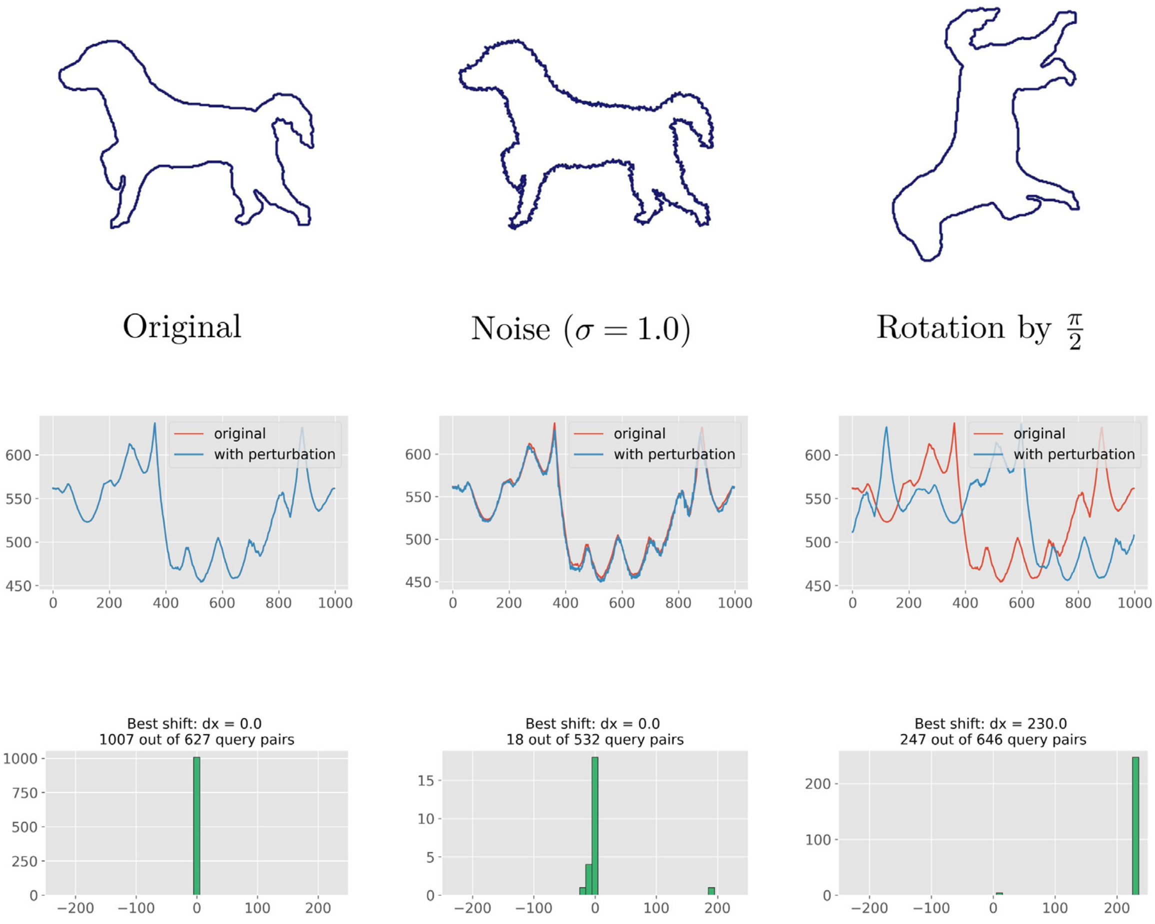


Fig. 12. Results of the alignment for different perturbations. The first row displays the contour extracted from the transformed shape, the second row shows the corresponding signatures and the third row presents the resulting histograms. It should be mentioned that the noise in this case is quite large and represents a worst-case scenario.

ing it with the shapes already processed and stored in h. Let $\mathcal{S}$ be the database of shapes stored in h. The shape k in the database is $\mathcal{S}(k)$, $k \in \{0, 1, \ldots, n-1\}$, where n is the number of shapes in the database.

The algorithm goes through all pair descriptors in $\mathcal{Q}$, and for each of them, we retrieve a list of tuples of abscissa and index that we write $\{(x, k)\}$ for clarity. If the retrieved value is None, the next descriptor of $\mathcal{Q}$ is processed, ending up retrieving all the potential shape candidates for each investigated pair. The retrieved values lack information (only one abscissa value x along with its corresponding shape index k) and may be numerous for each query pair. Therefore an additional strategy is employed to select the relevant ones and prune the acquired set.

6.2. Pairwise geometric alignment

If x and x_Q are respectively the abscissa in the current retrieved tuple and the first point of the current pair in the query, we define:

$$\delta x \triangleq x - x_Q \tag{5}$$

Eq. (5) refers to the shift needed to map x_Q onto x. An example of such calculation is provided in Fig. 11. For any part of the signature which matches the query, the resulting values of δx should be close to another one. The values of δx are quantized (Eq. (6)) and we calculate histograms of δx per shape of index k in the database. The bins of the histograms range from $\delta_{\min} = -1000$ to $\delta_{\max} = +1000$ and have N_{bins} bins. The resulting binning is expressed by Eq. (6). The number of bins is again a parameter for the finesse of the matching; values ranging from 100 to 500 give fairly good results and here we choose 200 bins. We determine the quantized values of δx *on the fly* for each retrieved tuple, and we successively identify the score of the current shape *score*[k] as the value of the largest bin. The scores are stored in a dictionary indexed by the shape indices.

$$\Delta x = \left\lfloor N_{bins} \cdot \frac{\delta x - \delta_{min}}{\delta_{max} - \delta_{min}} \right\rfloor \tag{6}$$

Fig. 12 illustrates the behavior of such histograms. If the original shape is matched against itself, the correspondences are perfect, and therefore the retrieval yields a maximum number of pairs. This number is far larger than the number of pairs in the query because it is likely to find several results corresponding to one input descriptor. As explained in Section 3.2, noisy shapes disturb the signature signal and can thus corrupt the retrieval and rotation is the reason for an important shift in the signatures which can be seen in the histogram. There is a significant drop in the number of pairs retrieved, but nonetheless a very coherent shift is found.

6.3. Selection of the best shape

The aforementioned pipeline of techniques is compiled into a single Algorithm 3. There are essentially two passes to compute the scores: one for the forward query and one for the backward query. At each iteration, we retrieve the correspondences from the database and update the score and the best shape associated using the steps described in Section 6.2. We only keep the largest score for each bin of each shape. The best score, corresponding to the largest number of pairs retrieved over all shapes in $\mathcal{S}$ is $n_{best} = \max_k(score[k])$. Finally, the index of the best shape is given by $k_{best} = \arg\max_k(score[k])$.

6.4. Computational complexity

The descriptors and the quantization produce collisions during storage. This allows to get more than one possible shape corresponding to the query and also several pairs belonging to the same shape. Of course, the main issue is that it causes a linear computational complexity instead of a purely constant one. In theory, given a set of associative arrays, each having b_i buckets, the mean number of collisions can be assumed to be uniformly distributed and equals to:

$$\gamma = N / \prod_{i=0}^{p-1} b_i \tag{7}$$

where N is the number of pairs stored. The number of pairs N is substituted by the total number of shapes in the database n since the former increases linearly with n. Using Eq. (7), the complexity becomes:

$$C(n) = \mathcal{O}(\gamma \cdot N_Q) = \mathcal{O}\left(\frac{n}{b^p} \cdot N_Q\right) \tag{8}$$

where N_Q is the number of pairs in the query and if we suppose $\prod_i b_i \approx b^p$. Eq. (8) shows that there is a linear relation between the complexity and the number of pairs N, but there is also a factor which, if well-chosen, can severely diminish the overall computational time. The value of b depends on the descriptor definition and also on the parameters $\mathcal{A}$; a carefully chosen set of parameters thus allows to limit collisions, to ensure that γ is relatively constant when N (and the number of shapes) varies and to only pick pairs closely related to the ones investigated as queries. We will demonstrate experimentally in Section 7.2 that we manage to keep the runtime low and γ quasi-constant.

Table 1
Parameters for each perturbation. A certain amount of values are uniformly chosen within a reasonable range. For the scaling, we include the factor 1.0, so that there are 11 values.

	Scaling (*factor*)	Rotation (*angle*)	Noise (σ)
Range	[0.5, 2]	$[0, 2\pi)$	[0, 1]
Amount	10 (+1)	16	11

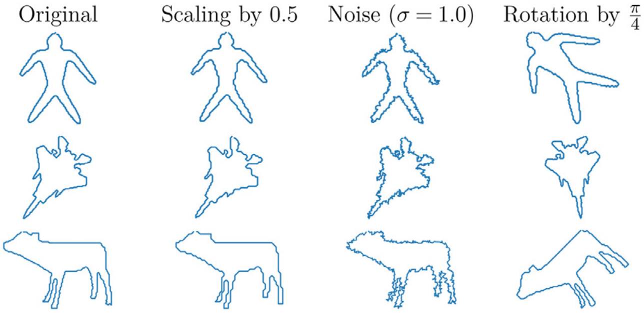


Fig. 13. Original and perturbed shapes of the Kimia99 dataset.

7. Experiments and results

To assess the performance of exact shape retrieval, studies on both robustness to perturbations and retrieval time are provided. Our technique is compared to several methods from the state of the art, including [9] as a standard baseline strategy with descriptors, [2] to evaluate the use of cross-correlation and [4] because their method inspired our signature. In the graphs described below, our method is labeled "LACS u", where u corresponds to the number of parallel systems of arrays used (see Section 5.3).

7.1. Robustness

This subsection presents an analysis of the resilience with respect to scaling, rotation and noise. These perturbations were generated by a process explained in Section 3.2. As a general protocol for this study, each transformed shape query is matched against the original shapes. Parameters for the perturbations are presented in Table 1. We choose to use the Kimia99 dataset [31] as a database for two reasons: it features a reasonable variety of inter-class shape complexities, each class sometimes containing very similar shapes, albeit without being a too large dataset, which leads to prohibitive computational times, especially for [2,4,9]. Fig. 13 shows some original and perturbed shapes of the Kimia99 dataset. The final values are accuracy scores corresponding to the bull's eye score for the retrieval of each independent shape, the retrieval per class and per parameter of perturbation.

Table 2
Total accuracy of robustness (%) by method including various amounts of parallel associative arrays.

	Shape Context [9]	Cui et al. [2]	TCD [4]	**LACS (Ours)** (number of parallel arrays)				
Kimia99				1	2	3	4	5
Scaling	66.6	13.7	71.5	80.2	90.3	90.5	92.4	92.8
Rotation	35.0	13.9	88.9	88.7	94.0	95.0	95.6	96.3
Noise	95.8	26.8	63.5	66.4	77.3	82.4	85.5	86.9
MPEG-7				**1**	**2**	**3**	**4**	**5**
Scaling			70.0	86.7	89.9	90.5	91.3	91.1
Rotation			74.6	83.5	87.4	87.9	88.4	88.6
Noise			63.7	68.2	76.2	79.1	80.8	82.8
Tari-1000				**1**	**2**	**3**	**4**	**5**
Scaling			72.6	93.5	95.7	95.8	96.2	96.3
Rotation			87.4	86.0	89.9	90.4	91.2	91.5
Noise			75.3	65.6	77.1	80.7	83.0	85.1

Table 3
Runtimes of retrieval (in seconds) for several methods and different amounts n of shapes in the database. Collisions γ for each n is also displayed between parentheses.

Database size n	Shape Context [9]	Cui et al. [2]	TCD [4]	LACS 1	LACS 2	LACS 3	LACS 4	LACS 5
1	0.93	0.066	0.010	0.003 (1.31)	0.005 (1.43)	0.011 (1.50)	0.020 (1.53)	0.018 (1.55)
10	8.95	0.47	0.070	0.002 (1.47)	0.006 (1.50)	0.011 (1.63)	0.022 (1.66)	0.025 (1.68)
100	86.34	5.19	0.61	0.013 (1.52)	0.009 (1.52)	0.030 (1.61)	0.045 (1.93)	0.121 (2.31)
1000	823.61	65.82	6.44	0.011 (3.42)	0.011 (3.42)	0.193 (3.91)	0.241 (5.18)	0.901 (6.57)

Results in Table 2 show that our method outperforms both [2] and [4], even with one associative array. Considerable improvements in robustness can be made by adding more associative arrays, but at the cost of more computational time. Shape context [9] achieves impressive results of robustness to noise, but is much less resilient for scaling and rotation. The TCD is designed to deal with non-rigid deformations, which obviously hinder its discrimination for very close shapes. The accuracy per class is however still very high in general, meaning that mistakes were mostly made between very close shapes (Fig. 14). Cross-correlation of [2] suffers from a very poor stability, as predicted. Our method achieves very good outcomes in general for the accuracies per transformation parameter. Scaling mistakes correlate with factors higher than 1.0. Image rotation involves drops in robustness that are more likely to appear for angles $\theta \not\equiv 0 \pmod{\pi/2}$. Finally, there is a rapid decline in robustness for higher noise standard deviations for all methods, except for the shape context, but we still manage to reach 68% for $\sigma = 1.0$.

We show some further results in Table 2 for the MPEG-7 dataset [35] and the Tari-1000 dataset [36]. They are usual datasets for the evaluation of shape retrieval methods and contain respectively 1400 and 1000 shapes. The results for LACS and TCD are comparable to those obtained on the Kimia99 dataset. As shown below, shape context and the method of Cui et al. have a quadratic time complexity. Hence, it is not possible to execute them on such a large dataset in a reasonable time interval.

7.2. Runtime analysis

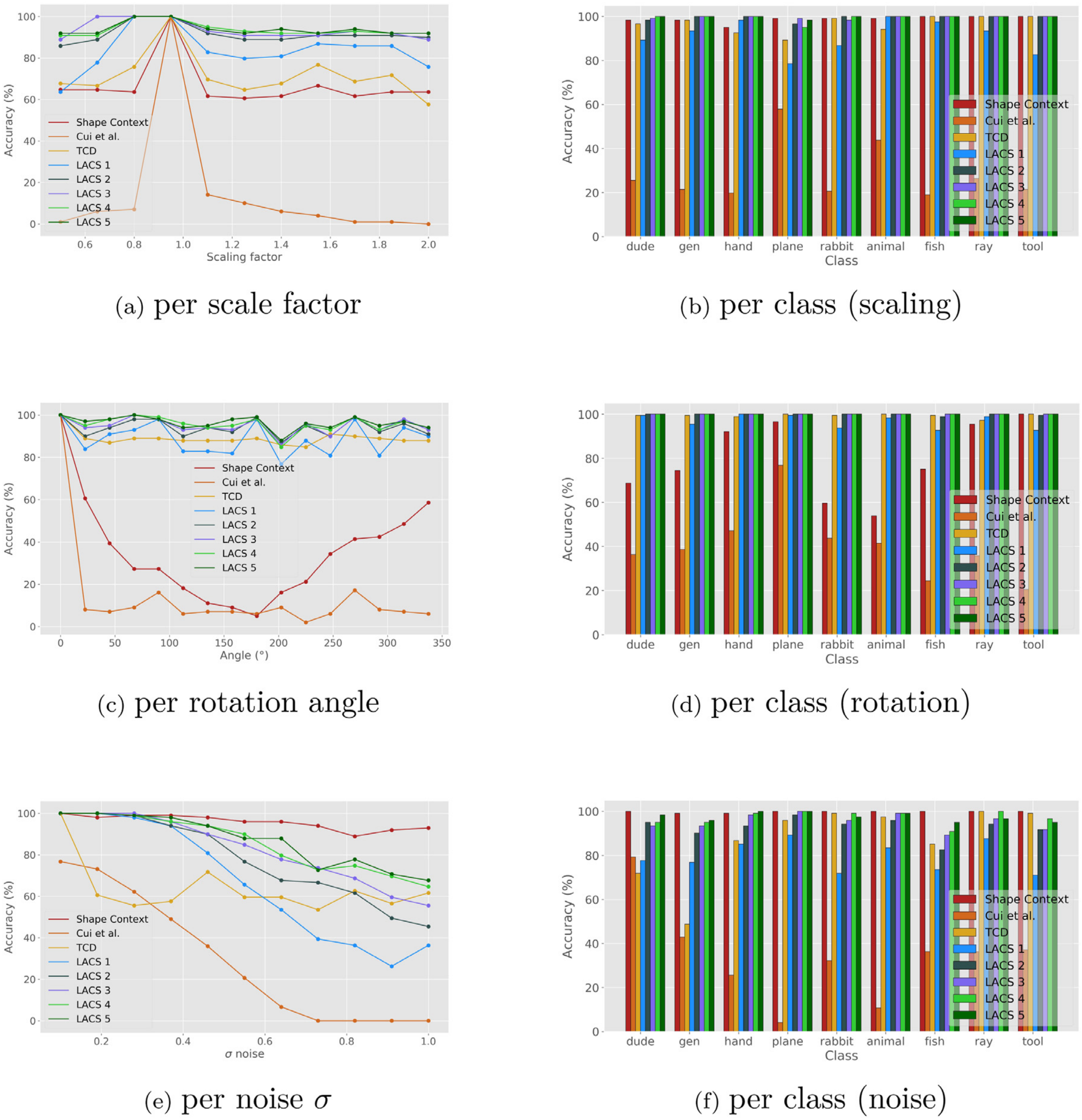
A runtime analysis is performed to evaluate the behavior of the associative arrays when changing the size of the database and the size of the query. We use the MPEG-7 dataset [35], which contains 1400 shapes, because we want to evaluate the runtime for large databases. For the first experiment, we select a specific query containing $N_Q = 380$ pairs, a value close to the mean number of pairs $\overline{N_Q} = 374$. We perform successive retrievals against a database in which we iteratively increase the number of shapes n; the same process is followed for every method. The second experiment is done by using the largest database (1400 shapes are stored) and by computing the retrieval time for an increasing number of pairs in the shape. To do so, we take the shape with the largest number of pairs and perform a process similar to the first experiment by adding pairs. In each experiment, we do not take into account the creation of the descriptors, the signatures or the associative arrays; the database is already already built and we examine only the retrieval time. Computations were run with an Intel® Core$^{\text{TM}}$ i7-7820HQ CPU @ 2.90GHz with Python 3.6.5 and the results are presented in Fig. 15.

Although we indeed observe a linear tendency, the curve is very flat, making the computational time of technique akin to a constant complexity. When the database is very small, there is a high probability of retrieving nothing (None in Algorithm 2) and so the retrieval time is negligible. This explains the soaring at first, which evens out at around 100 shapes. The average number of collisions (calculated with Eq. (7)) remains between $\gamma = 1.31$ and $\gamma = 2.92$, with a general mean of $\gamma = 2.27$. This implies the necessity to check, in average, 2.27 values r in Algorithm 3. The result is a very low retrieval time which spans between 3 ms and 13 ms, with an average of 10 ms. One can extrapolate that only 1 s is required to perform a retrieval with a query of 380 pairs among a database of around $n(t = 1s) = 350000$ shapes. Table 3 demonstrates the speed of our method with respect to more traditional approaches. In av-

Algorithm 2: Request on the associative array.

```
Input  : h: associative array
         $\{a_0, a_1, \ldots, a_{p-1}\}$ quantization parameters
         $\{d_0, d_1, \ldots, d_{p-1}\}$: input descriptor
Output : list of tuples

$L \leftarrow h_0$
for $i$ from 0 to $p-1$ do
    $\hat{d}_i = h_i(d_i)$
    if $L[\hat{d}_i]$ does not exist then
        return None
    $L \leftarrow L[\hat{d}_i]$
return $L$
```



(a) per scale factor

(b) per class (scaling)

(c) per rotation angle

(d) per class (rotation)

(e) per noise σ

(f) per class (noise)

Fig. 14. Detailed results of the robustness experiments. The left column shows accuracy against the perturbation parameters. The right column shows accuracy per class for each perturbation.

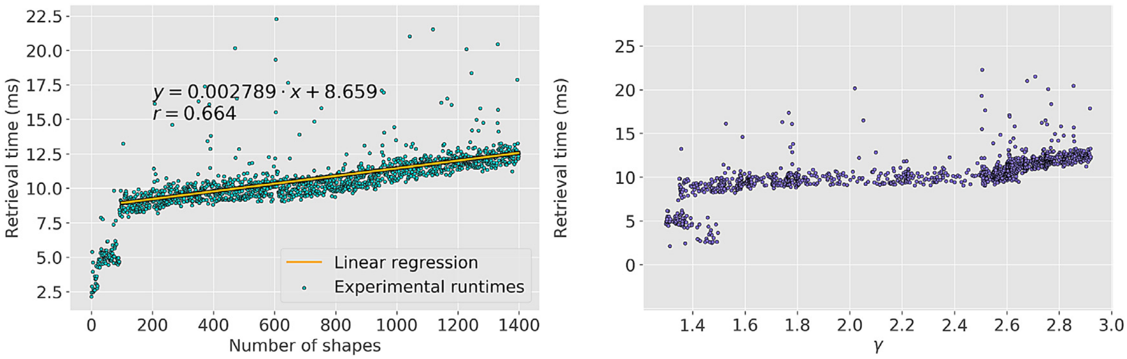


Fig. 15. Left: Retrieval time evaluations with respect to the size of the database for our technique with one array, using same query of $N_Q = 380$ pairs. A linear regression is shown beginning with $n = 100$. Right: Retrieval time per collisions γ.

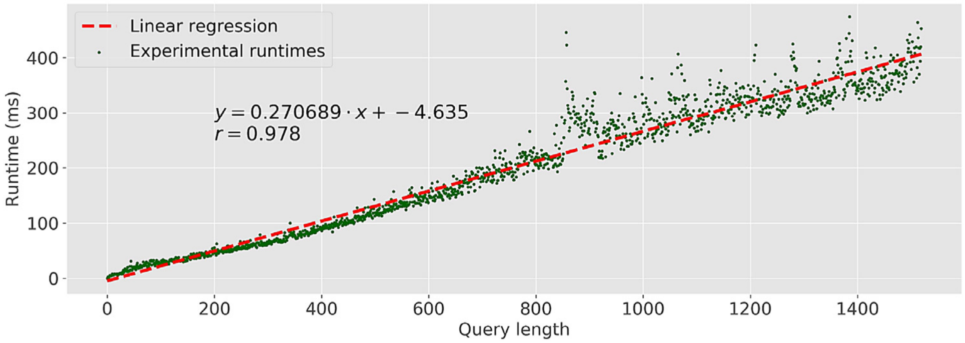


Fig. 16. Retrieval time evaluations with respect to the size of the query, using the largest database. Runtimes are very low, showing a near constant complexity.

Algorithm 3: Finding matches in the database from a shape query.

```
Input   : $h$: associative arrays
          $\mathcal{X} = \{x_0, \ldots, x_{p-1}\}$: list of abscissas of the first point of the query pairs
          $\mathcal{Q} = \{\mathcal{D}_0, \ldots, \mathcal{D}_{p-1}\}$: list of pair descriptors from the query shapes
          $\mathcal{A}$: quantization parameters
Output  : $k$: best shape index
          $score$: best score

$hist \leftarrow$ empty dictionary
$score \leftarrow 0$
$k \leftarrow$ None
$p \leftarrow |\mathcal{X}|$

for $i$ from 0 to $p-1$ do
    $R \leftarrow get(h, \mathcal{A}, \mathcal{D}_i)$    (Algorithm 2)

    if $R$ is not None then
        for all $r$ in $R$ do
            $x = r(0)$
            $idx = r(1)$

            $\delta x = x_i - x$    (Eq. 5)
            $\Delta x = quantize(\delta x)$    (Eq. 6)

            $bin = hist[idx][\Delta x]$ (ensure it exists first)
            $bin$ += 1

            if $bin \geq score$ then
                $k = idx$
                $score = bin$

return $k$, $score$
```

erage, it takes less than 1 s to perform retrieval for any number of parallel associative arrays. By comparison with direct matching, our method is 10 to 1000 times faster for the range of n analyzed.

We also present the number of collisions with respect to the n, which exhibit a linear correspondence between γ and the computational time. Fig. 16 features the evolution of runtime against N_Q as well, and explicitly shows a linear tendency. The number of pairs N_Q is positively correlated to the complexity of the shape, as defined by the number of high curvature points and its length. Since we expect an average of $\overline{N_Q} = 374$ pairs, the average computational time is of around 100 ms for $n = 1400$.

8. Conclusion and discussion

We introduced the Low-complexity Arrays of Contour Signatures (LACS) system for shape retrieval. The scope was to deal with exact shape recognition in a constant time, regardless of the size of the database. Our method formulates a very robust representation even in the presence of rigid perturbations and noise. Retrieval is performed with the use of a system of associative arrays and simultaneous pairwise geometric alignment, which yields extremely short computational times on large databases compared to usual works on the subject. Our technique only tackles exact matching and so far cannot deal with occlusions, which comes from the design of the signature. The pipeline for retrieval is largely interchangeable, as one may use other shape representations in our system. Further improvements could lead to very satisfactory results in terms of recognition, classification as well as part-to-part matching in a versatile and very fast way.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] F. Lardeux, S. Marchand, P. Gomez-Krämer, Multi-light energy map, in: Eurographics Workshop on Graphics and Cultural Heritage (GCH), 2018, pp. 199–202.
[2] M. Cui, J. Femiani, J. Hu, P. Wonka, A. Razdan, Curve matching for open 2D curves, Pattern Recognit Lett 30 (1) (2009) 1–10.
[3] A. Wang, An industrial strength audio search algorithm, in: International Conference on Music Information Retrieval (ISMIR), 2003.
[4] C. Yang, H. Wei, Q. Yu, A novel method for 2D nonrigid partial shape matching, Neurocomputing 275 (2018) 1160–1176.
[5] H.J. Wolfson, I. Rigoutsos, Geometric hashing: an overview, IEEE Computational Science and Engineering 4 (4) (1997) 10–21.
[6] D. Zhang, G. Lu, Shape-based image retrieval using generic Fourier descriptor, Signal Process. Image Commun. (10) (2002) 825–848.
[7] S.X. Liao, M. Pawlak, On image analysis by moments, IEEE Trans Pattern Anal Mach Intell 18 (3) (1996).
[8] H. Riemenschneider, M. Donoser, H. Bischof, Using partial edge contour matches for efficient object category localization, in: European Conference on Computer Vision (ECCV), 2010, pp. 29–42.
[9] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans Pattern Anal Mach Intell 24 (4) (2002) 509–522.
[10] M.R. Daliri, V. Torre, Robust symbolic representation for shape recognition and retrieval, Pattern Recognit 41 (2008) 1782–1798.
[11] Q. Jia, X. Fan, Y. Liu, H. Li, Z. Luo, H. Guo, Hierarchical projective invariant contexts for shape recognition, Pattern Recognit 52 (2016) 358–374.
[12] X. Shu, X. Wu, A novel contour descriptor for 2D shape matching and its application to image retrieval, Image Vis Comput 29 (2011) 286–294.
[13] L. Chang, M. Arias-Estrada, J. Hernández-Palancar, L. Sucar, Partial shape matching and retrieval under occlusion and noise, Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications (CIARP), 2014.

[14] J. Yang, H. Wang, J. Yuan, Y. Li, J. Liu, Invariant multi-scale descriptor for shape representation, matching and retrieval, Comput. Vision Image Understanding 145 (2016) 43–58.
[15] Z. Kun, M. Xiao, L. Xinguo, Shape matching based on multi-scale invariant features, IEEE Access 7 (2019) 115637–115649.
[16] J. Wang, X. Bai, X. You, W. Liu, L.J. Latecki, Shape matching and classification using height functions, Pattern Recognit Lett 33 (2012) 134–143.
[17] M. Donoser, H. Riemenschneider, H. Bischof, Efficient partial shape matching of outer contours, in: Asian Conference on Computer Vision (ACCV), 2009.
[18] Y. Zheng, F. Meng, J. Liu, B. Guo, Y. Song, X. Zhang, L. Wang, Fourier transform to group feature on generated coarser contours for fast 2D shape matching, IEEE Access 8 (2020) 90141–90152.
[19] W. Shen, Y. Jiang, W. Gao, D. Zeng, X. Wang, Shape recognition by bag of skeleton-associated contour parts, Pattern Recognit Lett 83 (2016) 321–329.
[20] F. Attneave, Some informational aspects of visual perception, Psychol Rev 61 (3) (1954) 183–193.
[21] N. Kaothanthong, J. Chun, T. Tokuyama, Distance interior ratio: a new shape signature for 2D shape retrieval, Pattern Recognit Lett 78 (2016) 14–21.
[22] K. Grauman, T. Darrell, Fast contour matching using approximate earth movers distance, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2004, pp. 220–227.
[23] M.-J. Lim, C.-H. Han, S.-W. Lee, Y.-H. Ko, Fast shape matching using statistical features of shape contexts, IEICE Trans Inf Syst 94-D (2011) 2056–2058.
[24] Z. Zhang, D. Pati, A. Srivastava, Bayesian clustering of shapes of curves, J Stat Plan Inference 166 (2015) 171–186.
[25] X. Wang, B. Feng, X. Bai, W. Liu, L.J. Latecki, Bag of contour fragments for robust shape classification, Pattern Recognit 47 (2014) 2116–2125.
[26] J. Zeng, M. Liu, X. Fu, R. Gu, L. Leng, Curvature bag of words model for shape recognition, IEEE Access 7 (2019) 57163–57171.
[27] W. Shen, C. Du, Y. Jiang, D. Zeng, Z. Zhang, Bag of shape features with a learned pooling function for shape recognition, Pattern Recognit Lett 106 (2018).
[28] X. Bai, C. Rao, X. Wang, Shape vocabulary: a robust and efficient shape representation for shape matching, IEEE Trans. Image Process. 23 (2014).
[29] K.B. Sun, B.J. Super, Classification of contour shapes using class segment sets, IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2 (2005) 727–733.
[30] D. Zhang, G. Lu, A comparative study of Fourier descriptors for shape representation and retrieval, in: Asian Conference on Computer Vision (ACCV), 2002.
[31] T. Sebastian, P. Klein, B. Kimia, Recognition of shapes by editing shock graphs, Transactions on the IEEE Pattern Analysis and Machine Intelligence 26 (2004) 550–571.
[32] Y. Lamdan, H.J. Wolfson, Geometric hashing: A general and efficient model-based recognition scheme, in: IEEE International Conference on Computer Vision (ICCV), 1988.
[33] H. Kaplan, J. Tenenbaum, Locality sensitive hashing for efficient similar polygon retrieval, ArXiv (2021).
[34] L. Wei, E.J. Keogh, X. Xi, M. Yoder, Efficiently finding unusual shapes in large image databases, Data Min Knowl Discov 17 (3) (2008) 343–376.
[35] The Moving Picture Experts Group, MPEG-7 dataset (2004).
[36] C. Asian, S. Tari, An axis-based representation for recognition, in: Tenth IEEE International Conference on Computer Vision (ICCV'05), 2, 2005, pp. 1339–1346.

Florian Lardeux obtained in 2016 a Masters degree in engineering at Institut d'Optique Graduate School in France, as well as a Masters degree in image processing from the University of Bordeaux. He is currently pursuing a Ph.D. in computer vision and image processing, focusing on the analysis, recognition and classification of quasi-flat objects from images.

Sylvain Marchand received the M.Sc. degree in algorithmics and the Ph.D. degree, while carrying out research in computer music and sound modeling, from the University of Bordeaux 1, France, on 1996 and 2000, respectively. He was appointed Associate Professor at LaBRI (Computer Science Laboratory), University of Bordeaux, France, in 2001. In 2011, he was promoted full Professor at Lab-STICC, University of Brest, France. He was there at the head of the Image and Sound curriculum. Since 2015, Sylvain Marchand is Professor of Computer Science at the L3i laboratory, University of La Rochelle, France. He was the leader of the French ANR DReaM project, and is a member of the scientific committee of the international conference on Digital Audio Effects (DAFx). He was associate editor of the IEEE Transactions on Audio, Speech, and Language Processing, and Senior IEEE Member.

Petra Gomez-Krämer is Associate Professor in Computer Science at the L3i laboratory of the La Rochelle University in France. In 2007, she obtained the PhD degree in Computer Science from the University of Bordeaux 1 and then she was a researcher at the Vicomtech research center in Spain until 2009. Her main research interests are image, video and document processing.

Storage
Shape database
$\mathcal{S}(k),\ k \in \{0, 1, ..., n-1\}$
Feature pairs
store
Retrieval
Query shape $\mathcal{Q}$
Feature pairs
retrieve
h_1
h_2
...
h_p
$n_{\mathcal{H}}$
2
1
$\mathcal{H}$
Pairs alignment binning
$\mathcal{S}(k_{best})$