Augmenting the Generalized Hough Transform to Enable the Mining of Petroglyphs

Qiang Zhu Xiaoyue Wang Eamonn Keogh [1]Sang-Hee Lee
Dept. of Computer Science & Engineering, [1]Dept. of Anthropology
University of California, Riverside
Riverside, CA 92521
{qzhu, xwang, eamonn}@cs.ucr.edu, sang-hee.lee@ucr.edu

ABSTRACT

Rock art is an archaeological term for human-made markings on stone, including carved markings, known as petroglyphs, and painted markings, known as pictographs. It is believed that there are millions of petroglyphs in North America alone, and the study of this valued cultural resource has implications even beyond anthropology and history. Surprisingly, although image processing, information retrieval and data mining have had large impacts on many human endeavors, they have had essentially zero impact on the study of rock art. In this work we identify the reasons for this, and introduce a novel distance measure and algorithms which allow efficient and effective data mining of large collections of rock art.

Keywords

Image Processing, Similarity Search, Cultural Artifacts.

1. INTRODUCTION

Rock art is an archaeological term for human-made markings on stone, including petroglyphs, *carvings* into stone surfaces and pictographs, *paintings* on stone. Figure 1 illustrates some examples of each, which hint at the extraordinary variability of rock art in terms of complexity.

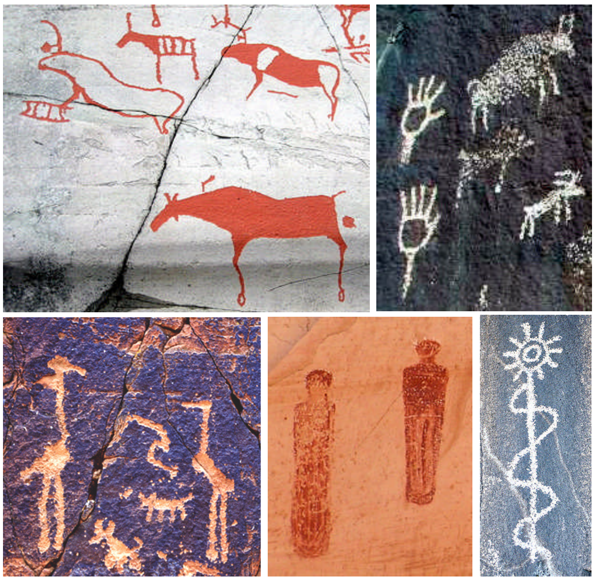


Figure 1: A random selection of petroglyphs and pictographs, hinting at their incredible variability, complexity and beauty

Petroglyphs and pictographs are one of the earliest expressions of abstract thinking, and a true hallmark of humanity. They provide a rich body of information on several different dimensions, beyond their value as an aesthetic expression. Studies of rock art have implications beyond anthropology and history. For example, a recent study postulates the existence of a now-extinct Australian bat species based on extraordinarily detailed pictographs known to be at least 17,500 years old [19]; petroglyphs have been used in studies of climate change; the changing inventories of species in the Dampier Archipelago from the Pleistocene to the early Holocene period have been reconstructed partly by petroglyph evidence [3]. However, in spite of these successes, progress in petroglyph research has been frustratingly slow.

A decade ago, Walt et al. summed up the state of petroglyph research by noting, "*Complete-site and cross-site research thus remains impossible, incomplete, or impressionistic*" [24]. Surprisingly, there has been little change in the intervening decade, yet in the same time frame we have seen significant advances in image processing and data mining. These advances have resulted in fielded applications in domains as diverse as medicine, entertainment, wildlife management, e-commerce, biometrics, zoology [18], etc. Nevertheless, these advances have had essentially zero impact on the analysis of petroglyphs and pictographs.

We believe that this is because the extraordinarily diverse and complex structure of rock art images defies most existing image matching algorithms. Most approaches are simply not suitable to capture the similarity of petroglyphs, and those that are, even in limited cases, do not scale to large collections we need to examine. In this work we introduce a novel distance measure for rock art, and show that it can correctly capture the *subjective* (and where available, *objective*) similarity between petroglyphs. We show how we can use this distance measure as a basis of several higher-level "data-mining" algorithms, for example finding repeated motifs, clustering, or simply enabling query-by-content. The rest of the paper is organized as follows. Section 2 contains background information and a discussion of related work. In Section 3 we review the Generalized Hough Transform, and show how we can adapt it to produce a fast and robust distance measure for petroglyphs. We test our ideas with a comprehensive set of experiments in Section 4, before offering conclusions and directions for future work in Section 5.

2. BACKGROUND AND RELATED WORK

The earliest petroglyphs have traditionally been associated with the appearance of modern humans in Europe such as the famous example from the Lascaux Cave, France, and an early one from the Chauvet Cave, France which dates back to as early as 30,000 years ago [22]. Recent work has shown that the idea of expressing abstract motif appears much earlier, 77,000 years ago in South Africa [10]. Given this long history, it is one of the most valuable sources of humanity that has persisted to the present time.

Beyond their value as an aesthetic expression, petroglyphs provide a rich source of information for researchers. Repeated motifs can be identified and traced through time and space, which in turn may shed light on the dynamic histories of human populations, patterns of their migrations and interactions, and even continuities to the present indigenous societies. However, the nature of petroglyphs poses an extremely difficult challenge. As in the case for any other artifacts of history, damages to petroglyphs are permanent and irreversible. However, unlike other artifacts that can be preserved and protected within the confines of a controlled environment in a museum, petroglyphs are mostly left in their natural settings, exposed to elements of nature that will erode them inevitably with time. There is an urgent need to identify petroglyphs and to archive them for humanity.

2.1 Background on Rock Art

As we shall show in Section 3, our algorithm assumes the input images are (relatively) low-resolution bitmaps with a 1-bit color depth, one petroglyph per image. However, as Figure 1 illustrates, obtaining such images may be non-trivial. With rare exceptions, petroglyphs do not lend themselves to automatic extraction with segmentation algorithms. For example, in the two images on the left of Figure 1, segmentation algorithms find the "edges" due to cracks in the rock to be more significant that the actual edges of the petroglyphs. Moreover, these images were chosen for this example for their high contrast and clarity; most petroglyphs would be even more challenging. In spite of this, in the next two sections we show how we easily obtained tens of thousands of petroglyphs for this study, and how we plan to have at least one million examples in the very near future.

2.1.1 Human Computation to Process Petroglyphs

The last five years has seen a flurry of research on *Human Computation*, much of it leveraging of the pioneering work of Luis von Ahn at CMU [1]. The essence of human computation is to have computers do as much work as possible to solve a given problem, but to outsource certain critical steps to humans. These steps are ones which are difficult for computers, but simple for humans. One of the most famous examples is the *Google Image Labeler*, which is a program that allows the user to label random images to help improve the quality of Google's image search results. Like many such efforts, human time is donated for free, because the task is embedded in a fun game, hence the recently coined term, Games with a Purpose, or GWAP [2].

In a parallel ongoing research effort, we have created a tool called *PetroAnnotator* which allows human volunteers to "help" computer algorithms segment and annotate petroglyphs. While the domain of interest does not have the broad appeal of *Google Image Labeler*, and is difficult to frame as a game, this does not matter. We tentatively estimate that if every grad student in anthropology in the US were to donate just one hour a month to the project, all the worlds' rock art could be processed in just a few years. We leave a detailed discuss of *PetroAnnotator* to a future publication; however the interested reader can find more details and working code at [27].

2.1.2 Existing Archives of Petroglyphs

Beyond the examples captured by our human computation system, there are several other rich sources of rock art data to be mined. For example, anthropologists have been sketching petroglyphs for hundreds of years, and recent efforts to digitize historical manuscripts have made at least hundreds of books, each with at least a few thousand petroglyph images, freely available on the web. In Figure 2 we show an example from the 1888 edition of a series of government reports [20].

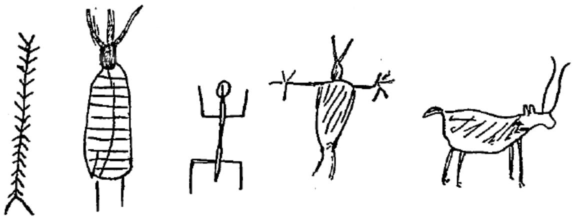


Figure 2: An excerpt from an 1888 government report [20]. The original caption is "*Petroglyph in Arizona*"

Images of this type can be of particular interest because they may refer to petroglyphs which have long since been destroyed. Furthermore, although the petroglyphs in Figure 2 predate photography, it is important to note that because petroglyphs often do not reproduce well in photographs, the practice of hand drawing or tracing petroglyphs is still used in modern anthropological texts.

2.2 Background on Image Processing

An understanding of *similarity* must be at the heart of any effort to analyze petroglyphs and other cultural artifacts. For example, an image of a horseman incised on a fossilized ostrich eggshell fragment was recently found among eolian deposits in the Gobi Desert, Mongolia [14]. An obvious thing to do with such an image in order to place it in a cultural context is to ask if a *similar* image exists in the many petroglyphs in the region. Thus, we began this project with careful consideration of shape similarity.

In soliciting feedback and advice for early previews of this work from various researchers in the data mining and image processing community, the feedback obtained was almost always of the form "*Very nice, but have you considered using X*", where *X* was Geometric Hashing, Hausdorff Distance, Chamfer Matching, Shape Contexts, Fréchet Distance, Skeleton Graphs, Zernike moments, Earth Movers, etc. While we have considered (and in some cases experimented with, see [27]) these distance measures, space limitations prohibit a detailed review and discussion of the pros and cons of each of them. Indeed, the preceding list is only a small subset of the hundreds of shape similarity measures in existence. See [23][26] and the references therein for an overview. However, we argue that some of the unique properties of petroglyphs render most of them unsuitable for the task at hand. Consider the following difficulties illustrated by Figure 3.

- A single atomic petroglyph may contain several disconnected parts. Thus, boundary based methods [12] and graph based methods [4] cannot be applied, at least not directly (c.f. Figure 12, which shows an example of a problem which would defeat boundary and graph based methods).
- Geometric hashing is a very useful technique for indexing large collections of shapes [25]. However, it is only well defined for machine parts and architectural drawings with

many clearly defined right angles/intersections/circle centers, etc. It has not been shown to have utility for more general unconstrained shapes.

- There are many specialized distance measures which have been introduced for indexing music notation, Japanese kanji, mathematical symbols, pen-based computing, etc. At least some subsets of these look like at least some subsets of petroglyphs. However, it must be remembered that in these domains there are only a finite (and relatively small) number of possible classes, and we can at least imagine an idealized prototype for each class (i.e. a perfectly drawn square root sign). However, this is not the case for petroglyphs which do not generally fall into discrete classes, and cannot generally be seen as corrupted versions of an idealized template.

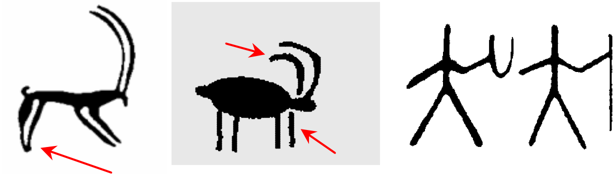
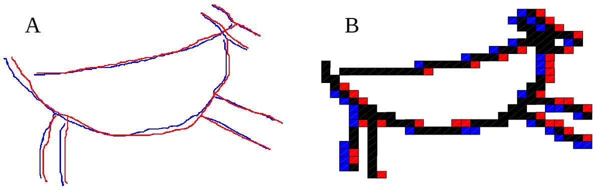


Figure 3: (*left*) An Ibex petroglyph taken from [21] has its two rear hoofs fused. It is not clear if this is an artifact of scanning or the artist's intent, and it *does* make a critical difference to graph based methods. (*center*) This bighorn sheep from a classic work [9] has a disconnected leg and horn, which will greatly affect its representation for graph based methods. (*right*) Two petroglyphs from Easter Island are clearly distinct, yet identical in graph based representations

Instead of attempting an exhaustive discussion of why we have discounted existing shape distance measures, we will briefly review the positive reasons for why we choose the GHT measure.

- As we shall show, on real, but unlabeled anthropological datasets, the GHT produced subjectively correct answers (cf. Section 4.1). Furthermore, on *labeled* datasets which are very similar to petroglyphs, GHT produces results which are competitive with state-of-the-art approaches.
- As we will demonstrate in this work, we are able to tightly lower bound the GHT, allowing for very efficient searches in large datasets. Moreover, we show that we can make a slight variant of the GHT obey the triangular inequality, thus allowing us to use off-the-shelf data mining algorithms, for example to find motifs.
- The GHT makes essentially no assumptions about the data, and thus is defined for open/closed boundaries, for connected/disconnected shapes, etc. This is important because, as hinted at in Figures 1, 2 and 3, petroglyphs are extraordinarily diverse.

We are now in a position to give some intuition as to why we intend to do data mining on a relatively low resolution of the petroglyph images. Using our *PetroAnnotator*, we asked two individuals to trace a petroglyph of a bighorn sheep petroglyph found in Arizona; the resulting two skeletons are shown in Figure 4.A. The skeletons are on a bitmap of 340 by 250. Although the two images are very similar, less than 3.5% of the pixels from each image overlap. We can contrast this with the situation after converting the images to a down sampled representation as shown in shown in Figure 4.B. Here the images are transformed to a mere 30 by 23 grid representation. However, of the 130 pixels that form each image, 75.6% of the pixels are common to both.



Key: Black pixels are common to both. Red or blue pixels are in one image, but not the other

Figure 4: A) Two overlaid skeleton traces of the same image of a Bighorn sheep, B) The same two images after downsampling

In essence, the original image representation has spurious precision. This precision is unwarranted because there is some uncertainty introduced by the human element of the algorithm[1]. The quantizing produced in the downsampling step also introduced some uncertainty, but this is completely dwarfed by original uncertainty. Furthermore, as we shall see, the lower resolution representation has several unique advantages which we can leverage off. In Section 5, we provide forceful empirical evidence that appropriate amounts of downsampling significantly improve accuracy in objective tests.

3. GENERALIZED HOUGH TRANSFORM

We begin by reviewing the classic generalized Hough Transform algorithm and then introduce our modifications and extensions.

3.1 Classic Generalized Hough Transform

The Hough transform [11][8] is a useful method for two-dimensional shape detection, but it is limited to analytic curves. It was generalized to *detect* arbitrary shapes in [5][15]; however, these works did not explicitly encode a similarity measure.

We note that there are many variants of the Hough transform, and the notation in the literature is inconsistent. The particular variant of the algorithm we consider, and the notation we will describe it, is most similar to Merlin and Farber's [15], in which shapes are constituted of *edge points*. Edge points are simply the dark pixels in our one-bit representation of shapes. Suppose we have a candidate shape *C* defined as:

$$C_{[x,y]} = \begin{cases} 0 & if\ [x,y]\ is\ an\ edge\ point \\ 1 & otherwise \end{cases}$$

and we want to find the *best fit* of a query shape *Q* defined in the same way as *C*. That is, given a reference point *R* in *Q*, to find the best point *R'* in *C*, if we put *C* onto *Q* (with only translation in the plane is allowed) and points *R* and *R'* coincide, then the number of matched edge points would be the maximal.

For clarity, we use a very simple example to illustrate the algorithm. Figure 5 shows a query shape *Q* and a candidate shape *C*. Note that the shapes can be disconnected, as in *Q*.

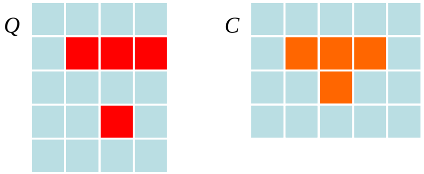


Figure 5: Toy examples of a query *Q* and a candidate match *C*. Each cell is a pixel, and the dark colors denote edge points of shapes

[1] For those rare petroglyphs that can be processed without human intervention, there is uncertainty introduced by camera angle, focal length, etc.

As shown in Figure 6, the first step is to mark a reference point *R* in *Q* (usually the center of mass of all edge points) and rotate edge points of *Q* around *R* by 180° (left and center of Figure 6). We then draw vectors from *R* to each edge point (as shown in the right of Figure 6). These vectors form a "star-like" pattern which we will use to determine the best fit of *Q* in *C*.

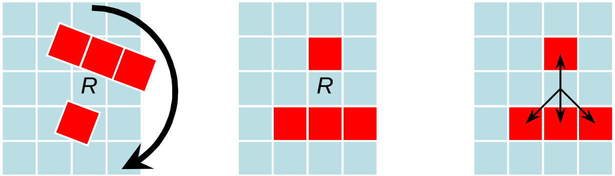
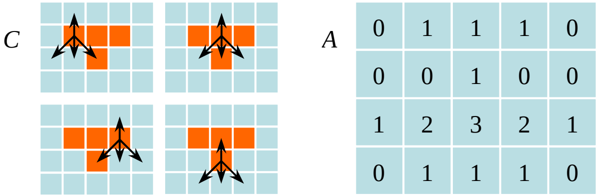


Figure 6: (*left and center*) The shape *Q* is rotated 180° around center of mass *R*. (*right*) four vectors of *Q* form a "star pattern"

To find both the best alignment of *Q* to *C*, together with a numeric evaluation of their similarity, we do the following. The "star" vectors are superimposed on each edge point of *C* (as shown in Figure 7.*left*). An accumulator matrix *A* of the same dimensions as *C* is used to record the number of vector-ends (i.e. the arrowheads) that fall into each cell (Figure 7.*right* shows the final accumulator).



0	1	1	1	0
0	0	1	0	0
1	2	3	2	1
0	1	1	1	0

Figure 7: Placement of vectors on each edge point of *C* (*left*) and the final accumulator *A* (*right*)

The cell in *A* with the maximal value is the best point *R'* we want to find, and its value equals the maximal number of edge points can be matched between *Q* and *C*. This is 3 in our example. Note that while *R* is the center of mass of *Q* by definition, point *R'* is not necessarily the center of mass of *C*.

Based on this maximal value, we can further obtain the *minimal unmatched edge points* (*MUE*) of *Q*. This is simply the number of edge points in *Q* minus maximal matched points. This *MUE* can be used as a distance measure. In our toy example, with similar shapes, its value is 1. If *Q* were exactly the same as *C*, the *MUE* would be 0, meaning D(*Q*,*C*) = 0. As we shall later see, it can be useful to normalize and adjust this number before using it as a distance measure.

For concreteness we show the algorithm to compute the *minimal unmatched edge points* in Table 1.

Table 1: The *minimal unmatched edge points* (*MUE*) from *Q* to *C*

```
Procedure [MUE] = Classic_GHT (Q, C)
1    $(Rx,Ry)$ ← center of mass of $Q$;
2    foreach edge points $(x,y)$ in $Q$
3        $x$ ← $2 \times Rx - x$; $Vx$ ← $x - Rx$;
4        $y$ ← $2 \times Ry - y$; $Vy$ ← $y - Ry$;
5        add $(Vx,Vy)$ to the set Vectors;
6    endfor
7    Initialize a matrix $A$ with the same size of $C$ to 0;
8    foreach edge points $(x,y)$ in $C$
9        foreach vector $(Vx,Vy)$ in Vectors
10           $A(x + Vx, y + Vy)$++;
11       endfor
12   endfor
13   MUE ← number of edge points of $Q$ – max($A$);
```

If *Q* and *C* have *S*×*S* pixels, and we denote the number of edge points in *Q* and *C* by N_Q and N_C respectively, then the time complexity of this algorithm is $O(N_Q \times N_C + S^2 \times \log S^2)$.

3.2 A New Cell Incrementation Strategy

The classic GHT algorithm can be seen as a cell value incrementation process of the accumulator (as reflected line 8-12 in Table 1), and we need to wait for all of the incrementation to finish before we can obtain the value for any particular cell. Here we propose a new cell value incrementation strategy which allows obtaining the cell values one by one. This will allow us, for the first time, to use a *lower bounding* strategy for the GHT.

Instead of superimposing vectors on edge points and increasing the value of the corresponding cell, we reverse this process by checking all positions that are possible to increase the value of one particular cell. To achieve this, we need to reverse the direction of vectors.

Figure 8 shows this simple idea (using the same example as in the last section): first we draw vectors from *R* to each edge point of *Q*, but without rotating *Q* (on the left); if we want to calculate the value of a particular cell, say, the one at the third row and second column, we superimpose all vectors on that cell (on the right). Then we check every cell with a vector falling into it: if this is also an edge point, we increase the cell value by 1 (because it is guaranteed, when using classic GHT, one vector superimposed on this edge point would fall into the target cell). Finally, after checking four cells, we obtain the value 2 for this cell.

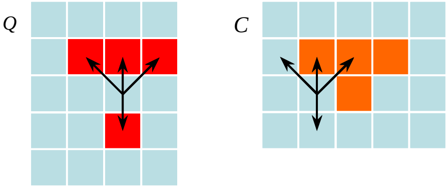


Figure 8: Four vectors of *Q* (*left*) and placement of vectors on one cell of *C* (*right*)

It is obvious that our new cell value incrementation strategy is equivalent to the classic one. However, this strategy has one advantage in that it allows for the implementation of the cell incrementation process in parallel, which avoids nesting for-loops in the classic GHT (line 8-12 in Table 1). In this paper, we are not going to discuss this. We will utilize the nice property "obtaining cell value one by one" as a basis to explore a lower bound of minimal unmatched edge points in the next two sections.

3.3 The Intuition behind Lower Bounding

As noted above, the time complexity of the GHT is quite high, and this limits its applicability for larger datasets. The classic data mining solution to the problem of time consuming distance measures is to find an efficiently computable tight *lower bound* to the distance measure, and to use this bound to cheaply prune off unpromising candidates [12].

We are now in a position to show the first known lower bound of the GHT-based distance. Our idea is based on extracting one-dimensional "signatures" from the two-dimensional query and candidate images. While we extract signatures from both the rows and columns, for ease of exposition we begin by showing just the column signature, which we denote as *SigCx*.

For a candidate shape *C* with *m* rows and *n* columns, we have:

$$SigCx = \{\sum_{i=1}^{m} C_{[1,i]}, \sum_{i=1}^{m} C_{[2,i]}, \dots, \sum_{i=1}^{m} C_{[n,i]}\}$$

In other words, we are simply counting all of the edge points in each column of *C*. For example, the truncated-corner square shape shown on the Figure 9.*right* has *SigCx* = {0,0,0,3,2,2,2,3,0,0,0}

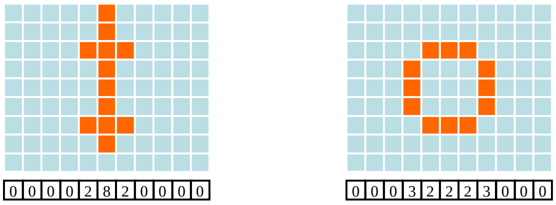


Figure 9: We can extract "signatures" from shapes by summing up the number of edge points in each column

We can extract these signatures as part of the preprocessing of the images, and store them in an index. At query time, we can use an identical technique to extract a signature, *SigQx*, from the query image *Q*. As shown in the Figure 10.*left* the only difference is that we truncate any leading or trailing 0's from the *SigQx* signature.

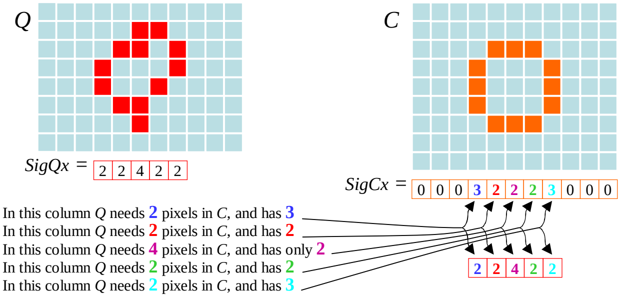

Q
C
SigQx = 2 2 4 2 2
SigCx = 0 0 0 3 2 2 2 3 0 0 0
In this column Q needs 2 pixels in C, and has 3
In this column Q needs 2 pixels in C, and has 2
In this column Q needs 4 pixels in C, and has only 2
In this column Q needs 2 pixels in C, and has 2
In this column Q needs 2 pixels in C, and has 3
2 2 4 2 2

Figure 10: (*left*) A query image *Q* has its signature *SigQx* extracted. (*right*) By noting how many edge points it *needs* *C* to have at each column, and how many edge points the column as *C actually* has, we can derive a lower bound of D(*Q*,*C*)

As it happens, the *MUE* distance in this case is 4, a number we can compute using the algorithm in the previous section. However, we can compute a lower bound to this value by looking at *just* the respective signatures.

We can obtain the intuition behind the lower bound by imagining that *Q* "wants" to match perfectly to *C*, with no missing edge points. As we place "star" vectors to one cell on the center column of *C*, if *Q* "wants" all vectors to fall into edge points of *C*, a necessary, but not sufficient, condition for this to happen is that the number of vectors falling into each column is less than or equal to the number of edge points in that column. This is equivalent to checking whether each value in a *SigQx* cell is less than or equal to the corresponding cell in *SigCx* (as shown in Figure 10).

Referring to Figure 10, we can see that in the slot $SigQx_1$ we need two edge points, and the corresponding slot in $SigCx_i$ actually has three. There is no penalty for *SigCx* having a surfeit of edge points. In the next slot $SigQx_2$ we need two edge points, and the corresponding slot in $SigCx_{i+1}$ has the two required edge points.

However, in the slot $SigQx_3$ we need four pixels, but the corresponding slot in $SigCx_{i+1}$ has only two pixels. Thus, we are guaranteed that no matter how the pixels are arranged, this column will contribute at least two to the number of missed edge points in the accumulator. As we continue, we find that neither of the two remaining slots contributes to the lower bound, because in each case there are at least enough pixels in *SigCx* to satisfy *SigQx*. Thus, we can say that in this alignment, the lower bound LB($SigQx,SigCx_{[4:8]}$) = 2.

Note that this lower bound is only for the particular alignment shown in Figure 10; if we had shifted *SigQx* one to the left, the lower bound would be 12, and if we had shifted *SigQx* one to the right, the lower bound would also be 12. If we test all alignments, we must choose the *smallest* value discovered as the true lower bound for the columns, which we denote as LB(*SigQx*,*SigCx*) = 2.

Finally, as hinted at above, we can do the same thing for the rows, using *SigQy* and *SigCy*. The final global lower bound to D(*Q*,*C*) is then simply the *larger* of the two individual lower bounds

3.4 A Formal Description of the Lower Bound

We expand the intuition presented in the last section to introduce a formal description of the lower bound. We again begin by considering the lower bound for just the columns. The algorithm is formalized in Table 2, which takes in a query shape *Q* and the column signature of candidate shape *C*. As described in the previous section, to obtain LB(*SigQx*,*SigCx*), we need to shift *SigQx* from left to right of *SigCx* by aligning the center of mass of *SigQx* to each cell of *SigCx* (lines 5,7 and 8 of Table 2). In each alignment, we calculate the lower bound for each column of *C*. Note that when some cells of *SigQx* shift out of *SigCx*, the edge points in these cells cannot find points in *C* to match them and then all contribute to the number of missed points (line 9-10 of Table 2). Finally, LB(*SigQx*,*SigCx*) is the minimal value of all these lower bounds (reflected in line 21-23 of Table 2).

One important optimization we use here is *early abandoning*. When calculating the lower bound for a column, if the number of missed points exceeds the current best (*smallest*) lower bound, we can stop calculations and shift to the next position (line 17-19 of Table 2). For a better pruning, we can align *SigQx* and *SigCx* by their centers of mass first, and then shift stepwise to two sides (omitted in Table 2 for brevity).

Table 2: Algorithm to calculate the column lower bound of GHT by giving the query shape *Q* and column signature of candidate shape *C*

```
Procedure [LBx] = LB_GHT(Q,SigCx)
1    SigQx ← column signature of Q;
2    LBx ← number of edge points in Q;
3    Rx ← center of mass of SigQx;
4    left ← Rx – 1;
5    for i ← 1: length(SigCx)
6        missed ← 0;
7        for j ← 1: length(SigQx)
8            k ← (i – left) + (j – 1);
9            if k < 1 || k > length(SigCx)
10               missed ← missed + SigQx[j];
11           else
12               delta ← SigQx[j] – SigCx[k];
13               if delta > 0
14                   missed ← missed + delta;
15               endif
16           endif
17           if missed > LBx
18               break;
19           endif
20       endfor
21       if missed < LBx
22           LBx ← missed;
23       endfor
24   endfor
```

In summary, we have:

$$LB(SigQx, SigCx) = \min_{i=1}^{length(SigCx)} LB(SigQx, SigCx[i-left : i-left+length(SigQx)-1])$$

To get the final lower bound, we simply run the algorithm in Table 2 again, this time with *SigCy* instead of *SigCx*, and with all column operators changed to row operations. After then calculating LB(*SigQy*,*SigCy*), the final lower bound LB(*Q*,*C*), is simply *max*[LB(*SigQx*,*SigCx*), LB(*SigQy*,*SigCy*)].

The time complexity of our lower bound algorithm is O(S^2). Note that it is independent of the number of edge points in images. As we shall show in Section 4.3, similarity search using the lower bound achieves a one to two order of magnitude speed-up.

3.5 Variants on the Basic Distance Measure

While the *MUE* is in itself a useful distance measure, it is helpful to consider slight variations of it to enable higher-level data mining algorithms. Note that in every case, we can still use the lower bound technique to speed up the high-level data mining algorithms. Below we consider three useful variants, and in the next section we empirically evaluate them.

Query-by-Content: In the simple examples we have considered thus far, we have implicitly assumed that the number of edge points in *Q* and *C* was the same. While *MUE* is surprisingly robust to small deviations from this assumption (say, less than a factor of two differences) it is clear that it has a bias. In particular, images that have relatively numerous edge points simply tend to be somewhat similar to everything. Since any large collection of images will invariably contain a few of these "rich" images, they can distort the results of any nearest neighbor searches. To mitigate this problem we define the nearest neighbor distance from *Q* to *C* as:

$$D_{nn}(Q,C) = \begin{cases} \dfrac{1}{N_Q - MUE(Q,C)} \sqrt{N_C / N_Q} & if \;\; N_C > N_Q \\[2ex] \dfrac{1}{N_Q - MUE(Q,C)} & otherwise \end{cases}$$

Note that we do not use *MUE* directly, but the inverse of "$N_Q - MUE$" (i.e. *maximal matched edge points*). The term $\sqrt{N_C / N_Q}$ is an explicit penalty for the problem $N_C >> N_Q$. Note that we can still use the lower bound of *MUE* to lower bound $D_{nn}(Q,C)$.

Clustering: The D_{nn} measure is perfect for similarity searching, which requires *one-to-all* matching. However, clustering requires *all-to-all* matching. In this case, with all things being equal, the D_{nn} measure would be biased into claiming that two images with many edge points are more similar than two images with few edge points. We can use $D_{clustering}(Q,C)$ to compensate for this:

$$D_{clustering}(Q,C) = \sqrt{N_Q \times N_C} \times [D_{nn}(Q,C) + D_{nn}(C,Q)]$$

Finding Motifs: Many data mining algorithms explicitly require a distance measure that obeys the triangular inequality. As a concrete example, we recently introduce an efficient and exact algorithm for finding motifs (approximately repeated patterns) [16], which makes no assumptions about the data or distance measure, other than the triangular inequality. We can modify *MUE* to obtain such a distance with:

$$D_{motifs}(Q,C) = (N_Q + N_C)/2 - (N_Q - MUE(Q,C))$$

The proof of triangular inequality can be found at [27].

4. EXPERIMENTAL RESULTS

We have designed all experiments such that they are not only reproducible, but *easily* reproducible. To this end, we have built a webpage [27] which contains all datasets and code used in this work, together with spreadsheets which contain the raw numbers displayed in all the figures. The webpage also contains many additional experiments which we did not include for brevity; however, we note that this paper is completely self-contained. All of the experiments are performed on a computer with an Intel i7-920 processor and 6.0GB of DDR3 memory.

4.1 Evaluation of Utility

We begin with simple sanity checks. We took a collection of petroglyphs from the Southwest USA and extracted fourteen images that would reasonably be grouped into seven pairs. Figure 11 shows the clustering obtained by our distance measure.

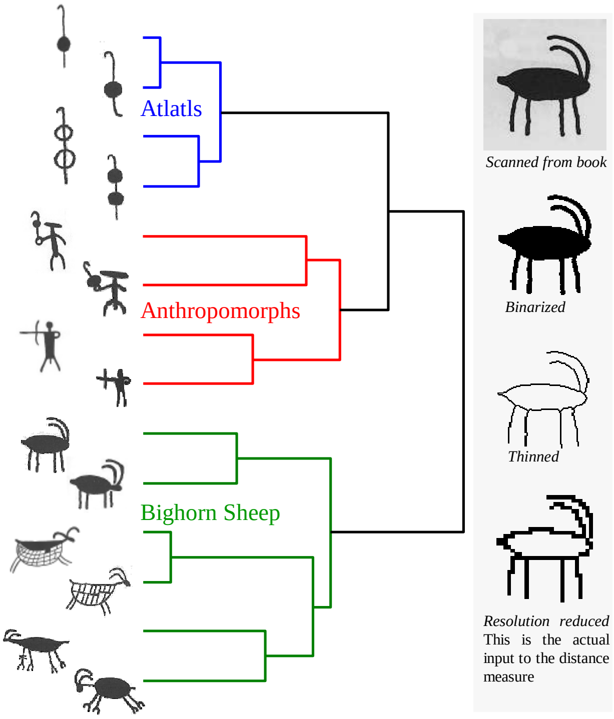

Atlatls
Anthropomorphs
Bighorn Sheep
Scanned from book
Binarized
Thinned
Resolution reduced This is the actual input to the distance measure

Figure 11: (*left*) A group-average linkage hierarchical clustering of typical Southwestern USA petroglyphs, with the $D_{clustering}$ measure. (*right*) While the dendrogram to the left shows the full resolution images for clarity, the images input to the distance measure have binarized, thinned and scaled to fit in a 30 by 30 bounding rectangle

Not only does the measure correctly group the seven pairs, but the higher level structure of the dendrogram correctly groups the images into Bighorn Sheep/Anthropomorphs/Atlatls[2]. Note that due to the thinning preprocessing step, the measure seems invariant to the hollow/solid nature of the Atlatls.

[2] An Atlatl is a spear-throwing device.

In the 1920's Dr. Stephen Chauvet noticed that many of the petroglyphs discovered on Easter Island showed humans in poses very similar to petroglyphs created by the Harappa culture (in what is now modern-day Pakistan). He noted these similarities in his 1935 text [7], which inspired a flurry of speculation about the origin of the Easter Island peoples[3]. It is natural to ask if our proposed distance measure could have "noticed" this similarity. This is a *very* difficult challenge for a distance measure, because the Harappa culture used stick-figures, whereas the Easter Island petroglyphs used highly stylized outlines. Nevertheless, as we can see in Figure 12, our method can capture the intuitive similarities.

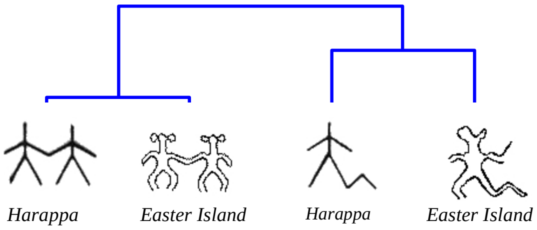


Figure 12: The GHT distance is able to find the intuitive similarity between pairs of anthropomorphic figures, in spite of the different styles of representations

4.2 Evaluation of Accuracy

Because there currently no large collections of objectively labeled petroglyphs, in this section we will test two publicly available datasets that are very similar to (some kinds of) petroglyphs. With these experiments we intend to show:

- Competitive or superior accuracy for query-by-content compared to some state-of-the-art algorithms.
- Relative insensitivity to amount of downsampling, which would mean our method is essentially parameter-free.
- As claimed in Figure 4, very high resolution imagery hinders rather than helps accuracy.

The first dataset is the NicIcon dataset [17], which contains 24,441 images from the 14 categories shown in Figure 13. Thirty-three participants were asked to sketch these icons in different sizes (small, medium and large) and a digital tablet was used to record the data (spatial, time and pressure coordinates). Note that counter to the original intention for the data and subsequent algorithms, our algorithm *only* considers the shape, and completely ignores pen speed and pressure information.

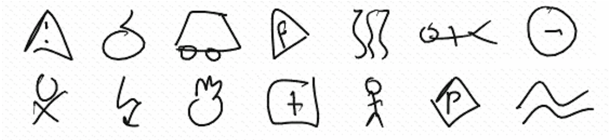


Figure 13: Examples of 14 categories from NicIcon dataset

We did both writer dependent (WD) and writer independent (WI) tests, in both cases, randomly choosing 60% of data as the train set and the rest as the test set, the same division as used the original paper [17].

The original data is 234×234 pixels. To explore the sensitivity of our algorithm to the amount of downsampling (its only user-specified parameter), we tested on six resolutions from 5×5 to 50×50 for both WD and WI tests, using the simple one-nearest-neighbor classifier. Figure 14 shows the results.

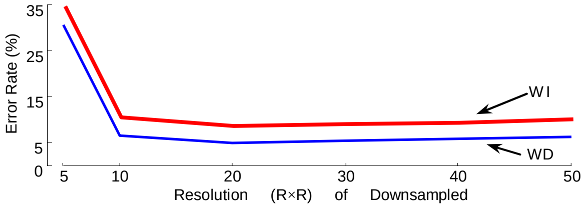


Figure 14: Error rate vs. Resolution. WD and WI tests on NicIcon dataset in 6 resolutions. Error rate makes little difference once the resolution is larger than 10×10

This plot suggests the sampling rate is not critical. The error rate only increased significantly when resolution was reduced to 5×5, which is clearly highly undersampled for any non-trivial dataset.

We obtained the best error rate 4.78% for WD and 8.46% for WI with the size of 20×20 pixels. The dataset creators tested on the online data using three classifiers [17]: the multilayered perceptron, the linear multi-class SVM classifier and a Dynamic Time Warping Based (DTWB) algorithm. The reported error rate for WD is from 1.94% to 15.61% and 5.3% to 20.01% for WI. Only the DTWB is better than our method, and recall that the DTWB had access to information about the pen speed, pen pressure, and the direction in which the lines were drawn, all of which is unknown to our algorithm. While the original authors do not measure time for classification, each comparison with the DTWB measure requires DTW calculations to be performed a number of times which are quadratic in the number of line strokes (i.e, the number of pen-ups) in each image, which is clearly very expensive.

We also tested without any downsampling, and the error rate increased dramatically: 31.75% for WD and 35.75% for WI, even worse than the ultra-low resolution 5×5. This verifies our analysis in Section 2.2.

Another petroglyph-like dataset is introduced by Khosravi and Kabir [13]. It is a very large dataset of handwritten Farsi digits extracted from about 11,942 registration forms. They obtained 102,352 binary images of Farsi digits, and chose 60,000 for training and 20,000 for testing (see samples in Figure 15).

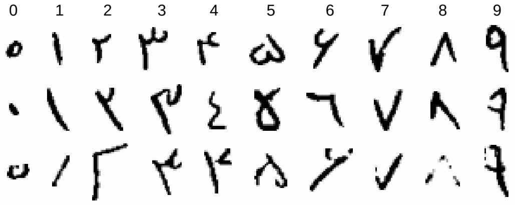


Figure 15: Sample digits from Farsi dataset. Note: number 2, 3 and 4 are very similar (3 and 4 in the third row are even impossible for human to distinguish); some digits have different styles (4 and 6); some digits are in bad quality (7, 8 and 9 in the third row)

The size of images in the Farsi dataset is smaller than in the NicIcon dataset: the minimum bounding rectangle (MBR) of the largest digits is 54×64 pixels. We tested on four downsampling resolutions from 5×5 to 30×30, using a one-nearest-neighbor classification using the same train and test data splits. The results are shown in Figure 16.

[3] DNA analyses now shows that this speculation was wrong; the Easter Island people are descended from Polynesians.

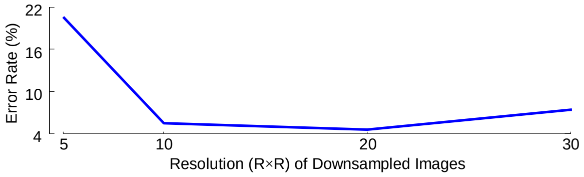


Figure 16: Error rate vs. Resolution. One-nearest-neighbor classification on Farsi dataset with four resolutions. Note that the error rate varies little when the resolution is greater than 10×10

We obtained the best error rate of 4.54% in the resolution of 20×20 (the same as the best resolution for the NicIcon dataset). Borji et al. [6] performed extensive empirical tests on this dataset, testing multiple algorithms, 3-NN, ANN, SVM$_{polynomial}$, SVM$_{linear}$ and SVM$_{RBF}$, each with four parameter choices (two choices of filters *times* two numbers of orientations). Of the twenty reported error rates, the mean was 8.69% and only four combinations beat our approach with a best performance of 2.36%. However, it is important to note that in addition to the two explicit parameter choices, there are at least four other parameters set "in the background" here.

Having shown that low resolution images can produce high accuracy in our domain, we have fixed the resolution to 30×30 pixels in all remaining experiments in this paper.

4.3 Evaluation of Speed and Scalability

As noted in Section 2, while we currently have only thousands of petroglyphs, we expect to shortly have on the order of a million. Therefore, we will test our algorithm dataset containing more than one million objects. To make this possible, we made our own synthetic petroglyphs dataset. We obtained the twenty-two petroglyphs (samples are shown in the top row of Figure 17). Then ten volunteers were asked to duplicate the petroglyphs by drawing them with an HP pavilion tx2510us tablet PC. A total of 250 petroglyphs were created in this way as our basic dataset (samples are shown in the second row of Figure 17). We then applied a random second-order *Polynomial Transformation* to each image in the basic dataset to make [39 79 159 319 639 1,279 2,559 5,119] distorted copies of each (as shown in the third row of Figure 17). With this basic dataset, we finally created eight datasets from size 10,000 to 1,280,000.

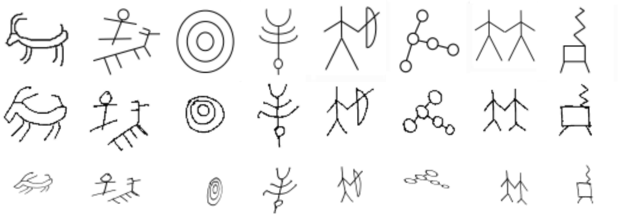


Figure 17: The Synthetic Petroglyphs Dataset. *first* row: samples of petroglyphs templates; *second* row: sample petroglyphs of the basic "human-copied" dataset; *third* row: samples of distorted petroglyphs. Note for each template, we have copies in different scales, translations, orientations and non-linear distortions

We first did a leave-one-out one-nearest-neighbor test. For each dataset, we randomly picked an image as the testing sample, removed it from the dataset and found its nearest neighbor using our lower bound based algorithm. We repeated this process ten times; Figure 18 shows the result.

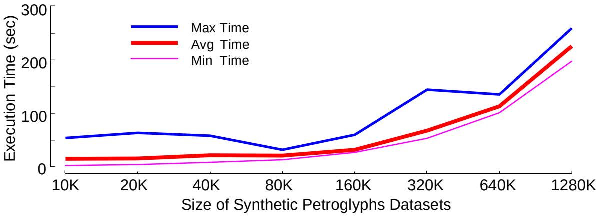


Figure 18: Time taken for the 1-NN query on eight synthetic petroglyphs datasets. For each dataset, maximal, average and minimal time of 10 runs are reported. Note log scale is used in x axis

We can see that the range between the maximal and minimal time is relatively small. When viewed on a normal scale plot (see [27]), we can see that the average running time is linear to the size of the dataset. While this is a test of *scalability*, we note in passing that the *accuracy* of this 22-class problem is 100% for all experiments.

It is natural to ask how much of the effectiveness of the search can be attributed to our lower bound. We measured the pruning rate:

$$\textit{pruning rate} = 1 - \frac{\textit{number of GHT calculations, lower bound search}}{\textit{number of GHT calculations, brute force search}}$$

for each of the 10 runs; the result is shown in Figure 19.

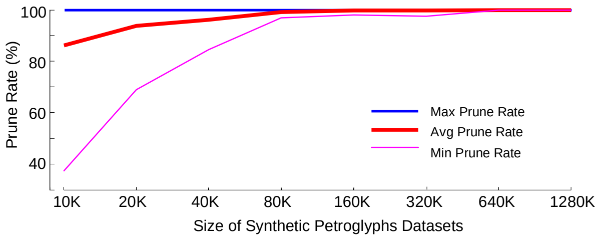


Figure 19: Pruning rate of our lower bound algorithm on eight synthetic petroglyphs datasets. For each dataset, maximal, average and minimal rates are reported. Note log scale is used in x axis

The results show that the pruning is extremely effective, particularly for larger datasets. The average prune rate exceeds 99.0% when examining 80,000 objects, and even the *minimal* prune rate is more than 96.9% at that point.

We also did a similar experiment with the brute force algorithm. Figure 20 compares the percentage of execution time for our lower bound algorithm relative to the brute force algorithm. Notice that for the largest dataset, our lower bound time is only 2% of the brute force one.

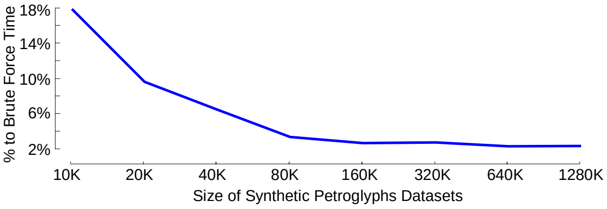


Figure 20: Percentage of execution time for our lower bound algorithm relative to the brute force algorithm. Note log scale is used in x axes

In addition to query-by-content, we also tested our ability to find motifs in these datasets. We can use the D_{motifs} distance measure combined with the algorithm recently published in [16] to efficiently find a pair of images whose distance is the smallest in a given dataset. Figure 21 shows the running time of finding motifs in our synthetic petroglyphs datasets.

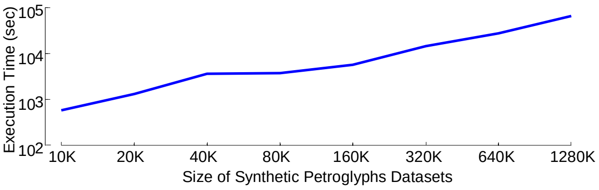


Figure 21: Time of finding motifs in eight synthetic petroglyphs datasets. Note log-log scale

A brute force algorithm to find motifs requires time quadratic in the size of dataset. But from a normal scale plot (see [27]), we find that our algorithm scales linearly. This is because we only need to calculate a tiny fraction of the exact distance between two images: even for the smallest dataset with 10,000 objects, we can prune 99.84% of the calculations, and by the time we are considering 1,280,000 images we are pruning more than 99.99% of the calculations. In Figure 22 we show the explicit speed-up over the brute force search. Even for the smallest dataset, our algorithm is 712 times faster and by the time we see the largest dataset, our algorithm is more than 100,000 times faster.

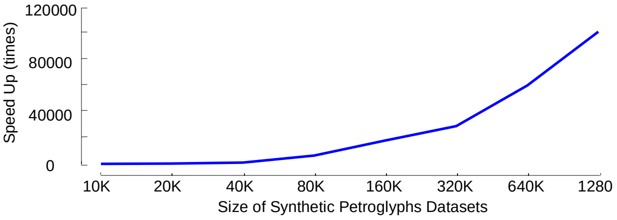


Figure 22: Speed-up of our lower bound algorithm against brute force algorithm of finding motifs in increasingly large petroglyphs datasets. For the brute force algorithm, we only ran it on the 10,000 datasets and extrapolated other values. Note log scale is used in x axis

While these results show that we can make the otherwise intractable task of finding motifs in large datasets tenable, it does not consider the *effectiveness*. Normally motif discovery cannot be evaluated directly in terms of accuracy, since we assume unlabeled data. However, since we actually know the labels in this case, we can measure the accuracy. For example when testing the dataset with 80,000 petroglyphs images (from 22 classes) over 100 runs on random sets of 80,000 objects (taken from a pool of 1280K) , we found that on 99 occasions the labels agreed.

5. CONCLUSIONS AND FUTURE WORK

In this work we consider, for the first time, the problem of mining large collections of rock art. We introduced an explicit framing of the GHT algorithm as a similarity measure, and showed that by lower bounding the measure we can effectively mine large data archives. Future work includes achieving rotation invariance and supporting partial shape matching.

Acknowledgements: We would like to thank the many donors of datasets, particularly Dr. Robert Mark and Evelyn Billo of *www.rupestrian.com*, and Taryn T. Rampley of UCR. This work was funded by NSF 0803410 and NSF 0808770.

6. REFERENCES

[1] von Ahn, L., Maurer, B., McMillen, C., Abraham, D., and Blum, M. 2008. reCAPTCHA: Human-Based Character Recognition via Web Security Measures. Science, September 12, 2008. Pages 1465-1468.

[2] von Ahn, L.2006. Games with a purpose. Computer, 39(6):92–94.

[3] Aseyev, I. V. 2008. Horseman image on an ostrich eggshell fragment. Archaeology Ethnology & Anthropology of Eurasia 34/2 (2008) 96–99

[4] Bai, X., and Latecki, L. J. 2008. Path similarity skeleton graph matching. IEEE PAMI, 30(7).

[5] Ballard, D. H. 1981. Generalizing the Hough transform to detect arbitrary shapes, Pattern Recognition 13, 111-122.

[6] Borji, A., Hamidi, M., Mahmoudi, F. 2008. Robust handwritten character recognition with features inspired by visual ventral stream, Neural Processing Letters, v.28 n.2, p.97-111, October 2008

[7] Chauvet, Stéphen-Charles. 1935. L'île de Pâques et ses Mystères (Easter Island and its Mysteries). Paris: Éditions Tel.

[8] Duda, R. O. and Hart, P. E. 1972. Use of the Hough transform to detect lines and curves in pictures, Commun. ACM 15(1) pp.11-15.

[9] Grant, C., Baird, J. & Pringle, J. K. 1968. Rock drawings of the coso range. Maturango Museum, China Lake, California.

[10] Henshilwood, CS., d'Errico, F., Yates, R., Jacobs, Z., Tribolo, C., Duller, GAT., Mercier, N., Sealy, JC., Valladas, H., Watts, I., and Wintle, AG. 2002. Emergence of modern human behavior: middle Stone Age engravings from South Africa. Science 295:1278-1280.

[11] Hough, P.V.C. 1966. Method and mean for recognizing complex pattern, USA patent 3,069,654.

[12] Keogh, E., Wei, L., Xi, X., Lee, S. H. and Vlachos, M. 2006. LB_Keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures. VLDB 2006.

[13] Khosravi, H., Kabir, E. 2007. Introducing a very large dataset of handwritten Farsi digits and a study on their varieties, Pattern Recognition Letters, v.28 n.10, p.1133-1141, July, 2007.

[14] McDonald, J. J., Veth, P. M. 2007. Pilbara and Western desert rock art: style graphics in arid landscapes. In: Rock art in the frame of Cultural Heritage of Humankind. Proceedings of the XXII Valcamonica Symposium 2007. pp. 327–334.

[15] Merlin, P. M., and Farber, D. J. 1975. A parallel mechanism for detecting curves in pictures, IEEE Trans. Comput. C24, 96-98.

[16] Mueen, A., Keogh, E., Zhu. Q., Cash, S., Westover, B. 2009. Exact Discovery of Time Series Motifs. SDM 2009.

[17] Niels, Ralph., Willems, Don. & Vuurpijl, Louis. 2008. The NicIcon database of handwritten icons. ICFHR 2008.

[18] Pan, J., Balan, A., Xing, Eric P., Traina, Agma J. M., Faloutsos, C. 2006. Automatic mining of fruit fly embryo images. KDD 2006.

[19] Pettigrew, J., Nugent, M., McPhee, A.,Wallman, J. 2008. An unexpected, stripe-faced flying fox in ice age tock art of Australia's Kimberley. Journal of Antiquity.

[20] Powell, J. W. (Editor) 1888. Annual report of the Bureau of American ethnology to the Secretary of the Smithsonian Institution. Bureau of American ethnology, Washington, D.C.

[21] Takaki, R., Toriwaki, J., Mizuno, S., Izuhara, R., Khudjanazarov, M. and Reutova, M. 2006. Shape analysis of petroglyphs in central Asia. Forma, Vol. 21 (No. 3), pp. 243-258.

[22] Valladas, H., Clottes, J., Geneste, J-M., Garcia, MA., Arnold, M., Cachier, H., and Tisnérat-Laborde, N. 2001. Palaeolithic paintings: evolution of prehistoric cave art. Nature 413:479.

[23] Veltkamp, R.C. 2001.Shape matching: similarity measures and algorithms. Int. Conf. on Shape Modeling and Applications.

[24] Walt, H., David, B., Brayer, J. & Musello, C. 2006. The International Rock Art Database Project. URL: *www.cs.unm.edu/~brayer/rock/waltet.html*

[25] Wolfson, H. J. & Rigoutsos, I. 1997. Geometric hashing: an overview. IEEE Computational Science and Engineering. 10-21.

[26] Zhang, D. and Lu, G. 2004. Review of shape representation and description techniques. Pattern Recognition, 37(1): 1-19.

[27] Zhu, Q. 2009. The Petroglyphs Webpage: *http://www.cs.ucr.edu/~qzhu/petro.html*