Shape Representation and Classification Using the Poisson Equation

Lena Gorelick, Meirav Galun, Eitan Sharon, *Member*, *IEEE Computer Society*, Ronen Basri, *Member*, *IEEE Computer Society*, and Achi Brandt

Abstract—We present a novel approach that allows us to reliably compute many useful properties of a silhouette. Our approach assigns, for every internal point of the silhouette, a value reflecting the mean time required for a random walk beginning at the point to hit the boundaries. This function can be computed by solving Poisson's equation, with the silhouette contours providing boundary conditions. We show how this function can be used to reliably extract various shape properties including part structure and rough skeleton, local orientation and aspect ratio of different parts, and convex and concave sections of the boundaries. In addition to this, we discuss properties of the solution and show how to efficiently compute this solution using multigrid algorithms. We demonstrate the utility of the extracted properties by using them for shape classification and retrieval.

Index Terms—Computer vision, shape, Poisson equation, silhouette classification.

1 Introduction

Silhouette contours contain detailed information about the shape of objects. In many cases, it is possible, given a silhouette, to determine the parts that compose a shape, identify their local orientation and rough aspect ratio, and detect convex and concave sections of the boundaries. When a silhouette is sufficiently detailed, people can readily identify the object or judge its similarity to other shapes (see examples in Fig. 1).

Computer vision systems may use similar information to classify objects. Silhouettes may be available to these systems as a result of segmentation. Simple thresholding can be applied in applications involving fairly isolated objects, such as objects placed on a conveyer belt or characters in a document. More sophisticated algorithms may be used, with a certain degree of success, to segment objects in cluttered scenes. In either case, properties of silhouettes extracted automatically and reliably provide (possibly in conjunction with additional properties such as color and texture) a powerful cue for recognition.

In this paper, we present a novel approach that allows us to reliably compute many useful properties of a silhouette. We consider a silhouette surrounded by a simple, closed contour. Based on the notion of random walks, we compute a function that assigns, for every internal point in the silhouette, a value reflecting the mean time required for a random walk beginning at the point to hit the boundaries. This function can be formalized as a partial differential equation, called the *Poisson equation*, with the silhouette contours providing boundary conditions. We then show how we can use the solution to the Poisson equation to reliably extract various properties of a shape including its part structure and rough skeleton, local orientation and aspect ratio of different parts, and convex and concave sections of the boundaries. In addition to this, we discuss properties of the solution and show how to efficiently compute this solution using multigrid algorithms. We demonstrate the utility of the extracted properties by using them for shape classification and retrieval. A preliminary version of this paper appeared in [26].

The paper is divided as follows: We discuss related work in Section 2. Section 3 introduces the Poisson equation and its properties. Section 4 describes how the solution can be used to extract various properties of a silhouette. Efficient multigrid solutions to the Poisson equation are discussed in Section 5. Section 6 introduces the classification algorithm we used in our experiments. Finally, Section 7 demonstrates the utility of the properties extracted with the Poisson equation through some applications.

2 Related Work

The computer vision literature contains numerous examples for the use of properties extracted from silhouettes.

Many early theories [6], [39] alongside more recent developments in digital imaging and computer vision, e.g., [44], [49], rely on part-based representations for object detection and recognition. Such methods first define a set of basic parameterized primitives (e.g., generalized cylinders [39], superquadrics [44], geons [6]). An object is then described by a collection of such primitives along with their relative spatial arrangement. Instances of these objects are identified in an image by fitting part models to regions in the image.

Perhaps the best known shape descriptor is the Medial Axis Transform, defined by Blum in the 1960s [8] as the loci of centers of bitangent circles that fit entirely within the

- *L. Gorelick, M. Galun, R. Basri, and A. Brandt are with the Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, PO Box 26, Rehovot 76100, Israel. E-mail: {lena.gorelick, meirav.galun, ronen.basri, achi.brandt}@weizmann.ac.il.*
- *E. Sharon is with the Division of Applied Mathematics, Brown University, 182 George Street, Providence, RI 02912. E-mail: eitans@dam.brown.edu.*

Manuscript received 9 Aug. 2005; revised 3 Apr. 2006; accepted 11 Apr. 2006; published online 12 Oct. 2006.
Recommended for acceptance by B.S. Manjunath.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0429-0805.

Fig. 1. A collection of silhouettes.

silhouette (forming a skeleton structure). The Medial Axis Transform can be computed by first solving the Eikonal equation $\|\nabla u\|^2 = 1$ (resulting in a function u that assigns to each internal pixel of a silhouette a value reflecting its minimum distance to the boundary contour) and then finding the ridges of the solution u. The Medial Axis Transform opened the way to the advent of skeleton-based representations. For example, studies such as [46], [50] utilize shock-graphs structures (medial axis, endowed with geometric and dynamic information at the points of discontinuities) to determine shape category. A similarity between two shapes is then produced by comparing their shock graph topology and attributes.

Another approach is to represent shape using a finite set of "features" or by embedding the shapes in an abstract space of shapes. A measure of dissimilarity (distance) can then be defined as the Euclidian distance between their features or more generally as the length of the shortest path (geodesics) connecting the shapes in their space. Common features include global properties, e.g., Fourier descriptors [37], geometric shape moments [20], Zernike moments [56], Angular Radial Transform [30], [19], [9], and boundary curve descriptors via Helmholtz PDE [57]. Local features may include local tangents, curvature, shape contexts [5], "Curvature Scale Space" [40], and other, "qualitative" descriptions of shape boundaries such as "Order Structure" [14]. In this case, computing the distance between shapes involves finding point-wise correspondences between the shapes' contours. These correspondences are often found by applying optimization techniques, particularly dynamic programming (e.g., [4], [23], [47]) or the fast marching method [22]. Various abstract shape spaces are proposed based, e.g., on conformal maps [48], harmonic embedding [17], or suitable parameterization of closed curves [31].

Another popular approach is to represent a shape as a geometric constellation of appearance templates (e.g., image fragments in [10], [54], "tags" in [2], and small patches in [1], [21], [35]).

Random walks are used in various vision applications including perceptual grouping and segmentation [3], [24], [38], [42], [55]. For example, in [42], [55], couplings between two edge elements are determined using a random walk process where the probability that a particle leaving an edge element reaches a certain location and orientation attenuates with the distance and curvature along its path. These coupling values are then used to extract smooth curves in images.

Finally, the Poisson equation is used in various computer vision applications. In [52], [36], [32], low-level vision problems such as optical flow, surface reconstruction, and shape from shading are formulated using variational principles whose Euler-Lagrange solutions take the form of a Poisson equation providing a necessary condition for a minimum. In [45], a generic interpolation machinery based on Poisson equations is introduced as a result of a similar variational formulation. This machinery is used in a variety of seamless image editing tools such as importation of source image regions into a destination regions, modifying the appearance of the selected regions, and many others.

In this paper, we present a new shape descriptor that is based on the notion of random walks and can be formulated as the solution to the Poisson equation. We use the solution to extract many useful properties of a shape, which are then integrated using shape moments to represent a shape as a set of global features.

3 The Poisson Equation

Consider a silhouette S embedded in a grid with mesh size h surrounded by a simple, closed contour ∂S. A sensible approach to inferring properties of the silhouette is to assign to every internal point a value that depends on the relative position of that point within the silhouette. One popular example is the distance transform, which assigns to every point within the silhouette a value reflecting its minimal distance to the boundary contour. An alternative approach is to place a set of particles at the point and let them move in a random walk until they hit the contour. Then, we can measure various statistics of this random walk, such as the mean time required for a particle to hit the boundaries. Let $U(x, y)$ denote this particular measure. Then, $U(x, y)$ can be computed recursively as follows: At the boundary of S, i.e., $(x, y) \in \partial S$, $U(x, y) = 0$. At every point (x, y) inside S, $U(x, y)$ is equal to the average value of its immediate four neighbors plus a constant (representing the amount of time required to get to an immediate neighbor), i.e.,

$$U(x,y) = 1 + \frac{1}{4}\Big(U(x+h,y) + U(x-h,y) + U(x,y+h) + U(x,y-h)\Big). \tag{1}$$

We set this constant to one time unit. Note that (1) is a discrete form approximation of the Poisson equation

$$\Delta U(x,y) = -\frac{4}{h^2}, \tag{2}$$

with $\Delta U = U_{xx} + U_{yy}$ denoting the Laplacian of U and $4/h^2$ denoting the overall scaling. For convenience, we set $4/h^2 = 1$ (intuitively, meaning one spatial unit per one time unit, where one spatial unit measures the distance to an immediate neighbor) and, therefore, solve

$$\Delta U(x,y) = -1, \tag{3}$$

with $(x, y) \in S$, subject to Dirichlet boundary conditions $U(x, y) = 0$ at the bounding contour ∂S.

Fig. 2 shows the solution to the Poisson equation obtained for the silhouettes in Fig. 1. High values of U are attained in the central part of the shape, whereas the external protrusions (the limbs, head, and tail) disappear at relatively low values of U. The level sets of U represent smoother versions of the bounding contour. This is different from the distance transform, which smoothes the shape near concavities while introducing discontinuities near convex sections of the contour (see Fig. 3). Also, unlike the distance transform in which every value is determined by a single contour point (the nearest), the values assigned by the Poisson equation take into account many points on the boundaries and, so, they reflect more global properties of the silhouette. Below, we

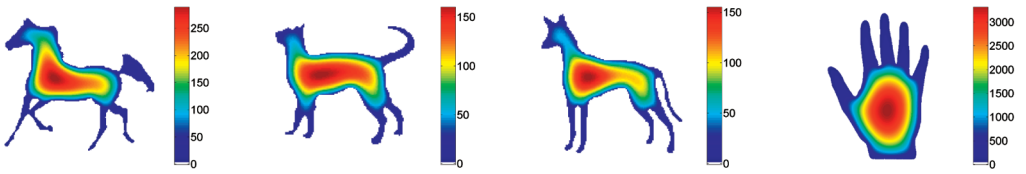


Fig. 2. Solutions to the Poisson equation for the silhouettes in Fig. 1.

exploit these properties of the Poisson solution to characterize a silhouette using measures constructed with derivatives of the solution. In addition, we will explain (in Section 5) that the solution to the Poisson equation can be calculated much faster than the distance transform.

Poisson's equation arises in gravitation and electrostatics. Here, we only mention a few of its relevant properties. The solution to the Poisson equation exists and is unique for any closed region with boundary conditions given by any integrable function. Uniqueness is shown by noticing that the solution to the related homogeneous equation $\Delta U = 0$ (called the *Laplace equation*) with zero boundary conditions is identically zero. More generally, the values of U along any closed curve within S determine the values of U inside the region bounded by this curve, but they are insufficient to uniquely determine the values of U outside the curve.

For silhouettes described by conics, the Poisson equation takes a particularly simple form. Consider a silhouette composed of the points (x, y) satisfying

$$P(x,y) = ax^2 + by^2 + cxy + dx + ey + f \leq 0. \tag{4}$$

Then, the solution is given by

$$U(x,y) = -\frac{P(x,y)}{2(a+b)}. \tag{5}$$

Note that the solution is unique and can be readily verified:

$$U_{xx} + U_{yy} = -\frac{P_{xx} + P_{yy}}{2(a+b)} = -\frac{2a+2b}{2(a+b)} = -1.$$

In this case, the level sets of U simply contain a nested collection of scaled versions of the boundaries, where the value of U increases quadratically as we approach the center (see Figs. 4 and 5).

For example, for a canonical ellipse with axes $(\tilde{a}, \tilde{b})$ given by

$$\frac{x^2}{\tilde{a}^2} + \frac{y^2}{\tilde{b}^2} < 1, \tag{6}$$

it follows from (5) that

$$\begin{aligned} U &= \frac{\tilde{a}^2\tilde{b}^2}{2(\tilde{a}^2+\tilde{b}^2)}\left(\frac{x^2}{\tilde{a}^2} + \frac{y^2}{\tilde{b}^2} - 1\right), \\ U_x &= \frac{-x\tilde{b}^2}{(\tilde{a}^2+\tilde{b}^2)}, \quad U_y = \frac{-y\tilde{a}^2}{(\tilde{a}^2+\tilde{b}^2)}, \\ U_{xx} &= \frac{-\tilde{b}^2}{(\tilde{a}^2+\tilde{b}^2)}, \quad U_{yy} = \frac{-\tilde{a}^2}{(\tilde{a}^2+\tilde{b}^2)}. \end{aligned} \tag{7}$$

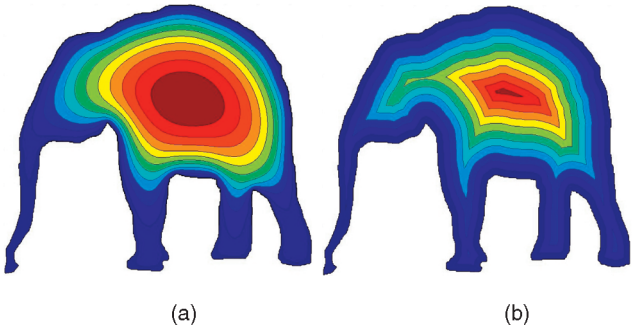


Fig. 3. (a) The level sets of the solution to the Poisson equation for the silhouette of an elephant. (b) The level sets of the distance transform for the same silhouette.

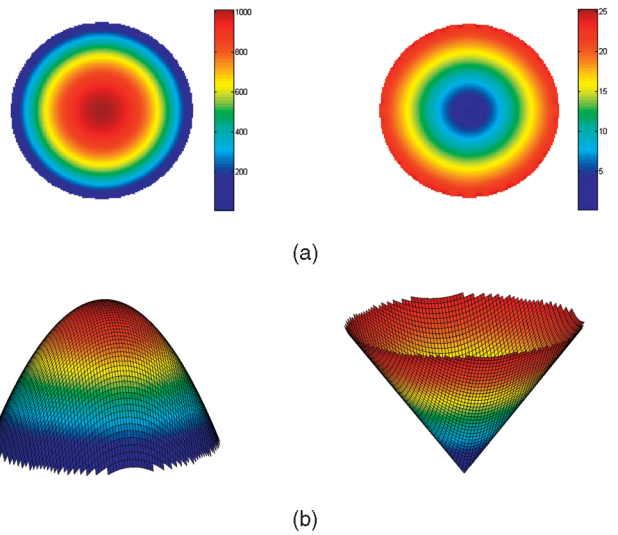


Fig. 4. (a) U computed for a circle and (b) the gradient of U.

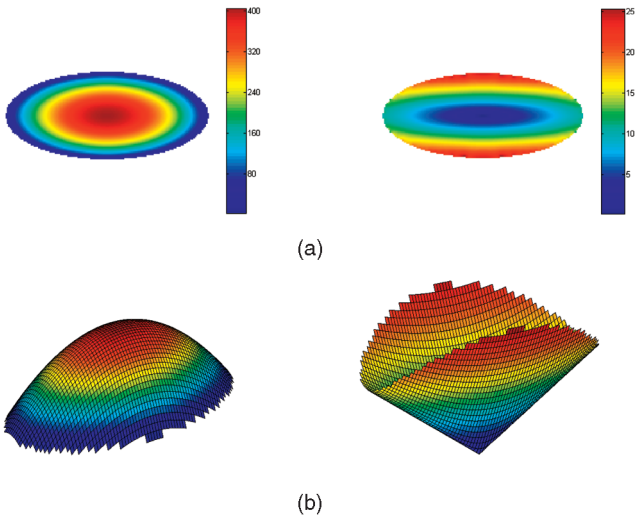


Fig. 5. (a) U computed for an ellipse's and (b) the gradient of U.

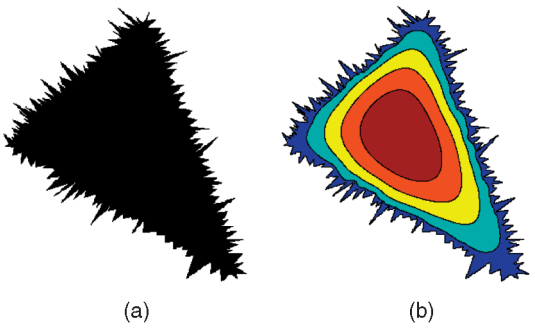


Fig. 6. (a) Noisy boundary and (b) a few level sets of low values.

The gradient of U increases from the origin towards the boundaries of the ellipse, reaching higher values at the upper and lower boundaries than at the left and right boundaries (assuming $\tilde{a} > \tilde{b}$ see Fig. 5). The values of the second derivatives of U are constant inside the ellipse, as can be seen from the (7). In general, for elongated objects, the function U has low second derivative in the direction of the length and a high second derivative in the direction of the width. In the case of a circle, the gradient increases quadratically from the origin toward the boundaries in all the directions (see Fig. 5). The second derivatives of U have the constant value of $-1/2$ at every internal point.

For silhouettes described by more complicated equations, the level sets of U represent smoother versions of the bounding curve (as is shown in Fig. 6). To see this, consider a line on which U oscillates (e.g., as a result of a ragged bounding contour). As we proceed toward the inside of the region, the oscillation will attenuate at an exponential rate. We can illustrate this "interior regularity" property (common to all smooth elliptic equations [11], [16]) by solving (3) within the infinite band $0 \leq y \leq L$ with boundary conditions $U(x,0) = e^{ix/\lambda}$ and $U(x,L) = 0$. The solution is given by $U = e^{-y/\lambda}e^{ix/\lambda} + 0.5y(L-y)$. Consider the value of U along a line parallel to the x-axis (constant y). The left term oscillates with the same wavelength as on the lower boundary, but with a decaying amplitude ($e^{-y/\lambda}$), whereas the right term is constant. The faster the oscillation (smaller λ), the faster the decay of its amplitude. Note that, due to the linearity of the Poisson equation, this analysis also applies to oscillations expressed as superpositions of frequencies. The interior regularity of the discrete version to Poisson equation (3) has also been proven [11].

4 Extracting Silhouette Properties

We can use the Poisson equation to extract a wide variety of useful properties of a silhouette. In this section, we provide a few examples. We begin by showing how the Poisson equation can be used to segment a silhouette into parts. Next, we show how we can identify corners at various resolution scales and derive a skeleton structure. Finally, we show how we can locally judge the orientation and rough aspect ratio of portions of the silhouette.

4.1 Hierarchical Shape Representation

Complex shapes can often be naturally described as a collection of parts. In particular, many common objects can undergo articulated motion in which every part moves rigidly while stretching or rotating with respect to the other parts of the object. For these reasons, a number of recognition schemes were proposed that rely on part representations (e.g., [6], [39], [46]). In this section, we show how we can use the Poisson equation to divide a shape into parts.

We will focus on shapes composed of a central part (e.g., the torso of an animal) to which exterior parts are connected (e.g., the head, limbs, and tail). In such a case, the highest values of U are obtained within the central part. Using an appropriate threshold, we can identify the central part (see Fig. 7). However, in this case, we obtain a smooth version of the central part and lose the portions of the central part that are near the boundaries since those portions yield low values of U.

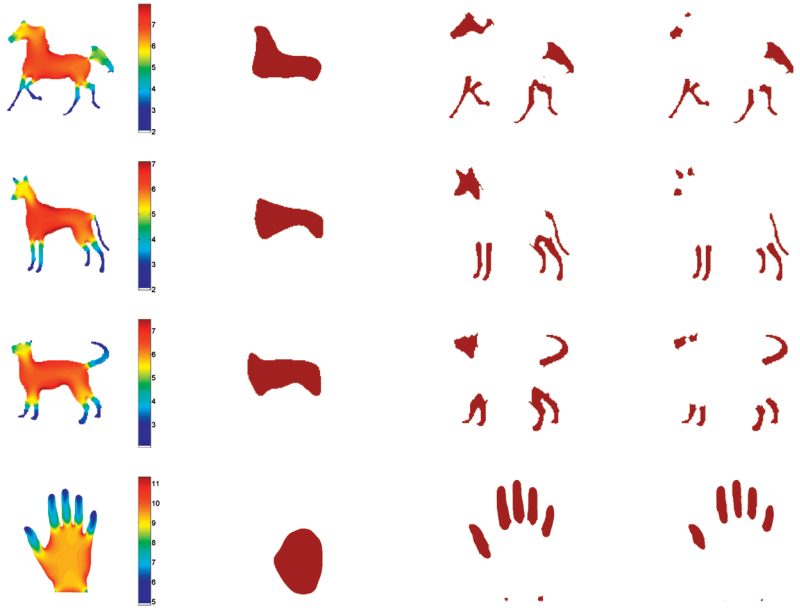


Fig. 7. From left to right: $\log \Phi$ computed for several silhouettes, central blobs extracted using 55 highest percentiles of U, and exterior parts extracted using uniform thresholds (78 and 66 low percentiles of $\log \Phi$).

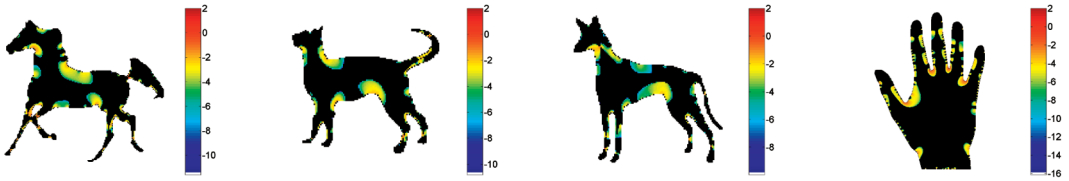


Fig. 8. Detecting concavities in different shapes using Ψ. The figure highlights $\log|\Psi|$ for all silhouette points with negative Ψ.

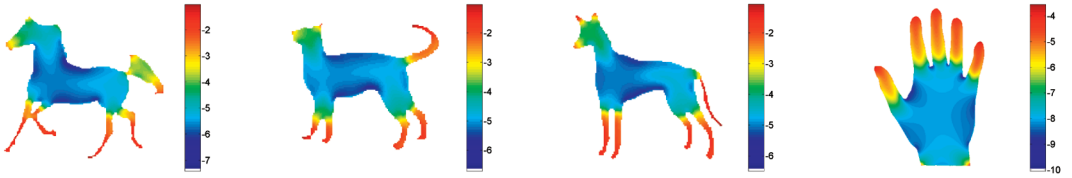


Fig. 9. Detecting convex sections in different shapes using low values of Φ. The figure shows $\log(1/\Phi)$ for all silhouette points. Notice that convex points are highlighted.

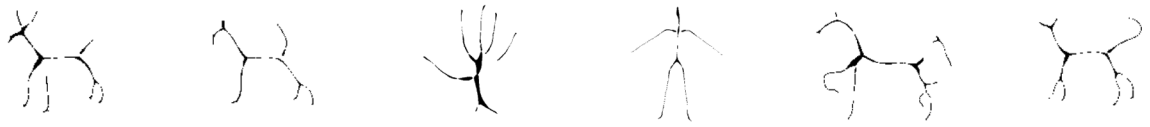


Fig. 10. Rough skeletons computed with $\tilde{\Psi}$. To obtain narrow skeletons we used a threshold, and repeated the computation of $\tilde{\Psi}$ on the obtained region.

We can include the portions of the central part that fall near the boundaries by noticing that these portions must have a high gradient. In order to identify the central part and the exterior parts of a given shape, we therefore define

$$\Phi(x,y) = U(x,y) + \|\nabla U(x,y)\|^2. \tag{8}$$

Φ has several interesting properties. First, it is constant on a disc ($\Phi = r^2/4$ on a disc of radius r). On other shapes, Φ generally has a saddle point exactly where U is maximum. This can be easily shown by first rotating the coordinate system such that $U_{xy} = 0$ at the point of maximum (causing Φ_{xy} to vanish as well) and then using the second derivative test discriminant as follows: Let p be the local maximum of U. Let U_θ be the directional derivative of U in the direction θ and denote the direction perpendicular to θ by $\theta^\perp$. Then, to satisfy the requirement $U_{\theta\theta^\perp}(p) = 0$, θ must satisfy the following relation:

$$\tan 2\theta = \frac{U_{xy}(p)}{1 + 2U_{xx}(p)}. \tag{9}$$

The solution exists unless $U_{xx} = U_{yy} = -1/2$ (in which case the local domain has a shape of a disc where Φ is a constant).

For example, on an ellipse, we obtain a saddle point at the center with low values along the major axis and high values along the minor axis. In more complex shapes, there may be additional saddle points at minor parts, indicating the hierarchy of the parts. The highest values of Φ are obtained near concavities. This follows the fact that U grows faster with distance from concavities than from convexities and, so, the gradient magnitude is higher.

By simply thresholding Φ, we can divide a shape into parts. Fig. 7 shows a few examples. Note that, by changing the threshold, we can also extract very small parts, such as the ear or a horseshoe.

4.2 Identifying Corners

For every point inside the silhouette, we can evaluate the curvature of the level set passing through this point using [43]

$$\Psi = -\nabla \cdot \left(\frac{\nabla U}{\|\nabla U\|}\right). \tag{10}$$

High level values of Ψ mark locations where level sets are significantly curved. Since the level sets of U are smooth versions of the bounding contour, this measure can be used, for example, to detect corners at different scales.

Negative values of Ψ identify all shape concavities, as is shown in Fig. 8. High negative values are obtained near local, sharp concavities (e.g., the horse's neck), and somewhat lower values in an extended area are obtained for large-scale concavities (e.g., the horse back). High positive values of Ψ will reveal all convexities, along with ridge (skeleton-like) locations. It is therefore preferable to detect convex corners using low values of Φ, see Fig. 9.

4.3 Skeleton

To further emphasize the skeleton of a shape, we may use a scale-invariant version of Ψ defined as:

$$\tilde{\Psi} = -\frac{U \cdot \Psi}{\|\nabla U\|}. \tag{11}$$

This measure emphasizes the skeletal structure, while attenuating the response for corners near the boundaries since, near the boundaries, U is small, while, near ridge location, $\|\nabla U\|$ is small. The skeleton obtained using the Poisson equation is similar, but not identical to the skeleton commonly obtained with the brushfire algorithm [8]. Fig. 10 shows rough skeletons computed with $\tilde{\Psi}$.

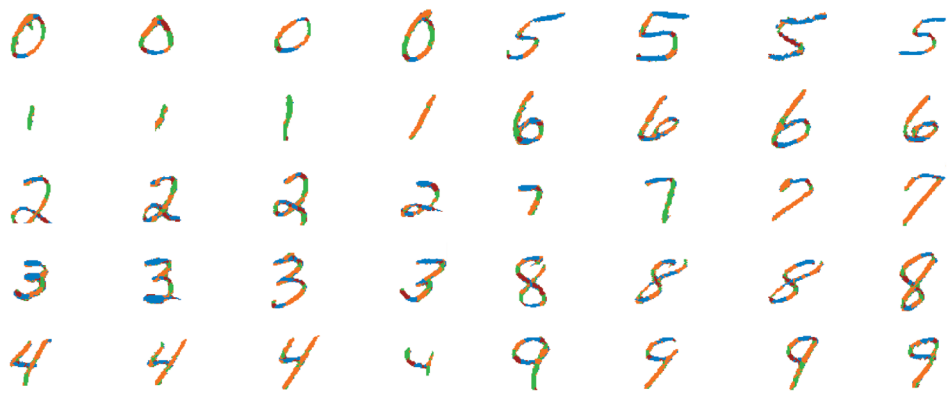


Fig. 11. A schematic representation of the second derivatives for handwritten numerals: Each pixel is assigned a color according to the direction in which the derivative at this pixel is minimal (indicating that the pixel lies in a part that is elongated in this direction), i.e., green for the vertical direction, blue for horizontal, brown and orange for the two diagonals.

4.4 Local Orientation and Aspect Ratio

The Poisson equation can also be used to estimate the local orientation and dimension of a shape. One example in which this may be useful is character recognition (OCR). Characters often can be described as a collection of elongated, narrow strokes, most often with no noticeable central part. The lack of a central part makes the method discussed in Section 4.1 above inappropriate for characters. We thus need a different procedure that can identify the parts that compose a character. Furthermore, it is important that this method will be applicable for very thin (pixel-wide) as well as thick characters.

A natural way to divide a character into parts is to identify the local orientation of the part and divide the character into sections of different orientation. This can be done simply by looking at the second derivatives of U. In a narrow elongated section of a shape, the level sets of U are largely parallel to the orientation of that section. The second derivative of U along a level set is generally very small, while the second derivative at the perpendicular direction must complement it to -1 and, so, it must be much larger (in absolute value). We may therefore identify the local orientation at each pixel by detecting the orientation θ in which $U_{\theta\theta}$ is small.

Figs. 11 and 12 show a schematic representation of local orientation for sets of thick and thin handwritten numerals. In the case of pixel-wide characters, we may use the smoothing postprocessing described in Section 5, although the detection of orientation also seems to work well without this smoothing. Note that the detection of orientation can be applied everywhere within the shape and, so, it is fairly insensitive to small perturbations of the contour. In Section 7 below, we show an example of using such detection in a numeral recognition task.

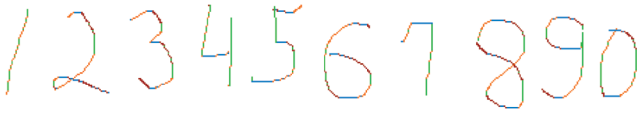


Fig. 12. A schematic representation of the second derivatives (similar to the previous figure) for pixel-wide numerals.

We can further generalize the analysis above to arbitrary shapes by constructing measures that locally estimate the second order moments of a shape near any given point. Consider a shape defined by a second order polynomial $P(x,y) < 0$, as in (4). As we mentioned in Section 3, every level set of U is simply a scaled version of $P(x,y)$ (5). If we now consider the Hessian matrix of U, we obtain at any given point exactly the same matrix, namely,

$$H(x,y) = -\frac{1}{a+b}\begin{pmatrix} a & c/2 \\ c/2 & b \end{pmatrix}. \tag{12}$$

This matrix is in fact the second moment matrix of the entire shape, scaled by a constant. The eigenvectors and eigenvalues of H, then will reveal the orientation of the shape, its aspect ratio, and will indicate whether the shape is elliptic or hyperbolic.

For more general shapes, the Hessian will vary continuously from one point to the next, but we can treat it as providing a measure of moments of shape felt locally by the point. Figs. 13 and 14 show a few examples. In these figures, $\lambda_{\min}(x,y)$ and $\lambda_{\max}(x,y)$ denote the eigenvalues of the Hessian matrix s.t. $|\lambda_{\min}(x,y)| \leq |\lambda_{\max}(x,y)|$ and $\alpha(x,y)$ denotes the orientation of the leading eigenvector.

5 Multigrid Solutions

The Poisson equation with Dirichlet boundary conditions can be discretized on a grid with mesh size h as a set of linear equations:

$$L^h u^h = f^h, \tag{13}$$

where L^h is an appropriate discretization of the negative Laplace operator $(-\Delta)$,

$$L^h u(i,j) \equiv \frac{1}{h^2}(4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}). \tag{14}$$

Numerical solutions to the Poisson equation can be obtained using successive Jacobi or Gauss-Seidel relaxations, but their convergence rate is slow. It requires $O(n^2)$ computer operations to reach an adequate solution for a silhouette region with n pixels. This slowness is a result of the fact that, after several relaxation sweeps, the obtained approximation

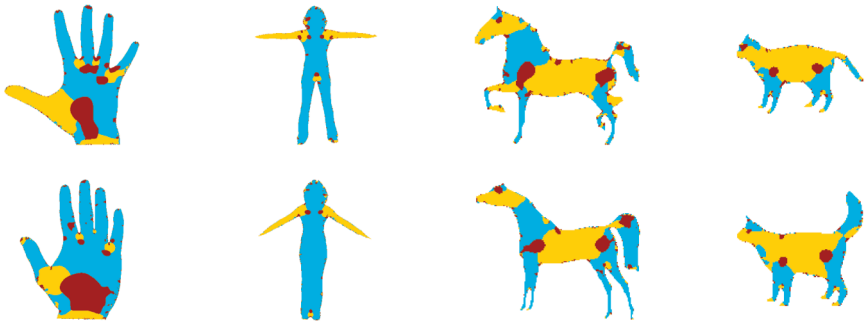


Fig. 13. A schematic representation of orientations computed via the Hessian matrix for several shapes. The turquoise color represents vertical regions ($|\alpha| \geq \frac{\pi}{4}$), yellow for horizontal ($|\alpha| < \frac{\pi}{4}$), and brown for isotropic sections $\sqrt{|\lambda_{\min}/\lambda_{\max}|} \geq \frac{2}{3}$). It can be seen that pairs of similar shapes give rise to similar orientation measures, but notice the thumb (left figure), which changes its orientation from near horizontal (top) to near vertical (bottom).

Fig. 14. Elliptic (brown) and hyperbolic (yellow) sections computed with the Hessian matrix for several shapes (sign of $\lambda_{\min} \cdot \lambda_{\max}$).

has a smooth error, which can only be slightly reduced by each relaxation sweep. Such a smooth error can, however, be approximated on a coarser grid. This in fact is what multigrid algorithms do.

A multigrid solver consists of several relaxation passes, followed by averaging the residual equations (the equations for the current, smooth error function) to represent them on a coarser grid with mesh size twice the fine-grid mesh size. The approximate solution to the resulting coarse grid equations yields a good approximation for the smooth error. The coarse-grid approximation is then interpolated back to the fine-grid and used to correct the previous, erroneous solution. This combination of several relaxation sweeps followed by a correction from a coarser grid is called *a multigrid cycle*. One such cycle typically reduces the error by an order of magnitude. The coarse equations in this cycle are themselves approximately solved by a similar cycle, i.e., several relaxation sweeps (on the coarse-grid equations) followed by a correction from *a still coarser* grid (with mesh size four times the finest-grid mesh size) and so on recursively until, at a very coarse grid, the equations are solved directly. The overall complexity of this multigrid algorithm is $O(n)$. The formal algorithm is described in the Appendix. (See analysis and more details in, e.g., [12]. For recent multigrid textbooks, see [13], [53].)

In fact, for all the applications described below, which largely require qualitative features rather than precise numerical values, a crude accuracy suffices. This can be achieved by applying just one, simplified cycle, which employs very naive boundary conditions at the coarse levels (placing the boundary at the nearest coarse grid points instead of modifying the nearby coarse equations to account for the fine, pixel-level location of the boundary). We observed no qualitative difference in the measured features between this one-cycle solver and a fully accurate, several-cycle solver. This plain solver costs less than two dozen operations per silhouette pixel.

Moreover, the interior regularity property described in Section 3 implies that at a distance r from the boundary a grid with mesh size $h = O(r)$ suffices to preserve the discretization error. This property can be utilized to further improve the complexity of the multigrid algorithm above. At each level with mesh size h_k, the modified version will conduct each step of the algorithm only at a neighborhood of the boundary whose thickness is $O(h_k)$. The complexity of such a solver is therefore linear in the number of contour points, while its accuracy is unchanged.

Finally, the solution obtained may be noisy near the boundaries due to discretization. To reduce this noise, we may apply as a postprocessing stage a few relaxation sweeps enforcing $U_{xx} + U_{yy} = -1$ inside the silhouette and $U_{xx} + U_{yy} = 0$ outside the silhouette without explicitly enforcing the Dirichlet boundary conditions on the boundary. This will smooth U near the boundaries ($U(x, y)$ will not necessarily be zero on $(x, y) \in \partial S$) and hardly affect more inner points.

6 Classification with Decision Trees

To demonstrate the utility of the shape properties extracted with the Poisson equation, we used them in shape classification experiments. In this section, we describe the classification algorithm used. Experimental results with these methods are presented later, in Section 7.

Our algorithm is based on a decision trees framework. The classifier is trained on a set of labeled shapes, each represented by a high-dimensional vector of features (as described in Section 7). It is natural to assume that some features are more prominent in some classes of shapes than the others and, thus, use different combinations of features at different steps of the classification algorithm. Therefore, decision trees were our first choice.

6.1 Training the Classifier

Given the training set, we recursively build a binary decision tree. At every node of the tree, the training data is split into two subsets (left and right sons). Since the data is high-dimensional, we split the data in a lower dimension by projecting all the data onto a line. Projection onto an arbitrary line will usually produce a confused mixture of samples from all of the classes. Thus, we will seek an orientation (linear combination of features) for which the projected samples are well separated [18], where by "well separated" we mean that each class is concentrated on some compact range of the line (ideally, different ranges for different classes), and it is possible to divide the set of classes into two subsets of classes such that the mixture between the classes in each subset is reduced drastically after the separation. Each of the two subsets of classes can, in turn, be split by repeating the same procedure.

To find the optimal discriminant direction $\mathbf{w}$, we define a score $S(\mathbf{w})$ as the ratio between the average width of the projected classes' clouds and the total projected width:

$$S(\mathbf{w}) = \frac{\tilde{\sigma}^2_{avg}}{\tilde{\sigma}^2} = \frac{\mathbf{w}^T A \mathbf{w}}{\mathbf{w}^T B \mathbf{w}}, \tag{15}$$

where A denotes the within-class scatter and B denotes the total scatter of the data points. We then look for a direction that minimizes this score. This definition is equivalent to the definition of the multiclass Fisher linear discriminant [18]. It can be shown that the vector $\mathbf{w}$ that minimizes S must satisfy the generalized eigenvalue problem

$$A\mathbf{w} = \lambda B\mathbf{w}. \tag{16}$$

We choose the eigenvector corresponding to the minimal eigenvalue (i.e., minimal ratio between the average within-class scatter and a total scatter) to be the discriminant. This is in contrast to the standard multiclass Fisher approach where the first k eigenvectors (with k denoting the number of classes) are used to reduce the dimensionality of the data.

Once the optimal direction $\mathbf{w}$ is found, we seek a threshold t along $\mathbf{w}$ such that, after splitting the data using this threshold, the mixture is maximally reduced in both sons of the current node.

We evaluate the data mixture at a node of the tree using the entropy function as a measure of impurity. Suppose we are given k classes $c_0, \ldots c_{k-1}$ and their respective observed frequencies $\{p_0, p_1, \ldots p_{k-1}\}$, where $p_q = P(x \in c_q),\ q \in 0, 1, \ldots, k-1$, and $\sum_{q=0}^{k-1} p_q = 1$. Then, the impurity of the data at a certain node v is defined as

$$i(v) = -\sum_{q=0}^{k-1} P(x \in c_q|v) \log P(x \in c_q|v). \tag{17}$$

Given a splitting threshold t on $\mathbf{w}$ and denote the fraction of data points at v that goes to v_l and v_r by $P_l(t)$ and $P_r(t)$, respectively ($P_l(t) + P_r(t) = 1$), then the quality of the split is defined as the decrease in impurity

$$\triangle i(v,t) = i(v) - P_l(t) i(v_l) - P_r(t) i(v_r), \tag{18}$$

where $i(v_l)$ and $i(v_r)$ are calculated with respect to t. We are looking for the threshold t^* such that $\triangle i(v, t^*)$ is maximized, i.e., $t^* = \arg\max_t \triangle i(v,t)$.

While the eigenvector corresponding to the smallest eigenvalue minimizes (15), the difference between the impurity decrease obtained for directions corresponding to the few smallest eigenvalues might be negligible. Moreover, the impurity decrease depends on both the direction $\mathbf{w}$ and the threshold t. Thus, at every node, we compute the maximal impurity decrease for the best three eigenvectors (i.e., corresponding to the three smallest eigenvalues) and choose the best pair of direction $\mathbf{w}^*$ and threshold t^* along it. The remaining two eigenvectors are stored along the branches of the tree for a possible future use in nodes where only few data points remain and overfitting might occur.

To avoid overfitting at the leaves level of a decision tree, we stop splitting whenever the sum of elements in the two biggest classes at the node is smaller than the number of features used. Then, we try to separate the data using one of the stored directions that were calculated when there were still enough statistics. Among these directions, we again choose the pair of direction and threshold to maximize impurity decrease.

6.1.1 Complexity Analysis

In the remainder of this section, we analyze the time complexity of the construction of a single decision tree. We denote the number of data points by n and the number of features d.

Consider first the number of operations needed at a node with n_v data points. To formulate the generalized eigenvalue problem in (16), we need to construct the within class scatter A and the total class scatter B matrices, which takes $O(n_v \cdot d^2)$ operations. Then, naive Cholesky factorization approach for solving the generalized eigenvalue problem takes $O(d^3)$ operations.

Given a candidate orientation (eigenvector) $\mathbf{w}$, finding the threshold t that maximizes $\triangle i(v,t)$ in (18) can be done by an exhaustive search over the n_v thresholds taken at n_v locations of the data points projected onto $\mathbf{w}$. First, we sort the data points along $\mathbf{w}$ (which takes $O(n_v \log n_v)$ operations) and then go over the sorted points. It takes $O(n_v)$ to compute the impurity decrease for the first threshold and $O(1)$ to compute the impurity decrease for each of the next thresholds (assigning each data point below or above the threshold, given the previous assignment).

There are three eigenvectors that are considered for each eigenvalue problem. Unused eigenvectors are accumulated from level to level of a tree for a possible use at lower levels when there are not enough statistics. Thus, the number of operations that are needed to find the best pair of orientation $\mathbf{w}$ and threshold t at the current node is at most

$$3i \cdot n_v \log n_v, \tag{19}$$

where i is the depth of the current node.

TABLE 1
An Overview of the Databases of Handwritten Numerals

DB name	img size	# training	# test	# in tree	error rate
USPS [28]	16×16	6975(75%)	2323(25%)	2794(30%)	$2.1\% \pm 0.3\%$
NIST19 [27]	64×64	20000	1000	3000	$1.8\% \pm 0.5\%$
MNIST [33]	28×28	20000	1000	3000	$2\% \pm 0.3\%$

For each data set, we provide image size, number of training samples used in our experiments, number of held out test samples, number of samples used to construct each tree, and classification error rate (mean and standard deviation).

Since we are using weighted impurity decrease criterion for splitting data at each level, our decision trees are built in an almost balanced way. (By "balanced" we mean that the number of the data points at a parent node is roughly double the amount of the data points at any of its sons.) Hence, for the simplicity of the analysis, we compute the complexity as if it was a binary balanced tree. In this case, the number of data points at every node is $n_v = n/2^i$ and the final complexity is then given by:

$$\begin{aligned} T(n) &= \sum_{i=0}^{\log n} 2^i \left[d^2 \frac{n}{2^i} + d^3 + 3i \frac{n}{2^i} \log \frac{n}{2^i} \right] \\ &= O(d^2 n \log n) + O(d^3 n) + O(n \log^2 n). \end{aligned} \tag{20}$$

6.2 Classification

To classify an unseen shape by a decision tree, the shape is "rolled" down the tree until it reaches a leaf. The information found at the leaf is used to assign a set of probabilities to the shape being classified. For any class label c_q present at the leaf with observed relative frequency p_q, the probability of the shape in question getting this label is exactly p_q. The complexity of rolling a shape down a single decision tree is $O(d \log n)$.

In addition, we also use an approach similar to [2] to improve the classification error of a single decision tree by resampling the training set several times and building a *Decision Forest*. To classify an unseen shape by a decision forest, the sets of label probabilities resulting from each decision tree are summed up, examined, and the label with the highest probability is chosen.

7 Experiments and Results

Below we describe preliminary classification and retrieval experiments which demonstrate the utility of properties extracted with the Poisson equation. In these experiments, we represent a shape by a collection of features derived for the shape using the Poisson solution (3). The features we use are weighted moments of the form

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} w(x,y) x^p y^q dx dy, \tag{21}$$

in which, for every feature, we substitute for $w(x,y)$ a different Poisson-based measure. Following is a description of the features used and the results obtained in each of the experiments.

7.1 Handwritten Numerals

In the first set of experiments, we learn to classify handwritten numerals. All numerals were first centered about their centroid and brought to a uniform scale using $\sqrt{\langle x^2 \rangle + \langle y^2 \rangle}$ as the normalization factor. We used two types of measures as weights in (21). The first measure, L_θ, identifies local orientation by detecting the orientation in which the second derivative of the solution is near zero. This measure is defined by

$$L_\theta(x,y) = e^{-\beta \left| \tilde{U}_{\theta\theta}(x,y) \right|}, \tag{22}$$

where $\tilde{U}_{\theta\theta}(x,y)$ denotes a three-pixel average of $U_{\theta\theta}(x,y)$, the second derivative of U in one of the four directions $\theta \in \{0, \pi/2, \pi/4, -\pi/4\}$ (we used $\beta = 3$). The second measure, $J_{\theta_1\theta_2}(x,y)$, identifies junctions by taking the geometric mean of two detectors, $L_{\theta_1}(x,y)$ and $L_{\theta_2}(x,y)$, in the six pairs of orientations (θ_1, θ_2) s.t. $\theta_1, \theta_2 \in \{0, \pi/2, \pi/4, -\pi/4\}$ and $\theta_1 < \theta_2$. We then computed moments of up to order $p + q = 5$ with each of the 10 measures above (four L_θ and six $J_{\theta_1\theta_2}$) as weights in (21) resulting in 21 moments per measure. In addition, we used two global properties, the aspect ratio of the numeral and the ratio between the product of the length and width of the shape determined by PCA and the area of the numeral. All together, this yields a vector of 212 features for each numeral.

We used these features to train a classifier with multiple decision trees (see Section 6), each trained on a random subset of the training set. We then test the algorithm by applying it to an unseen randomly chosen test set. We compare our results with those reported in the literature and with the results obtained using the same classifier when the features used include ordinary shape moments (computed as in (21) with $w(x,y) = \chi(x,y)$, the characteristic function of the shape).

An overview of the databases used in this set of experiments is shown in Table 1. For each of the databases, we report the obtained error rate along with the sizes of training and test sets and the sizes of the images. The error rate reported is an average over 20 random experiments, each performed on a random split of the database into training and test sets. In each experiment, the algorithm constructed a forest of 50 decision trees, each trained on a random subset of the training set (the size of the subset is reported as well). In the following paragraphs, we shortly summarize and compare the obtained results.

7.1.1 USPS Database

The US Postal Service database (USPS) [28] consists of 9,298 handwritten numerals of size 16×16 pixels with intensity values varying between 0 and 2. In order to extract a silhouette for each digit, we considered only the pixels with intensity values higher than 0.8. The average error rate obtained in 20 random experiments was $2.1\% \pm 0.3\%$, which is comparable to the state-of-the-art results (1.9 percent) reported on this database in [29].

7.1.2 NIST Special Database 19

The NIST special database 19 is the most comprehensive and apparently most difficult database available by NIST [27]. We randomly chose 20,000 numerals from the database and downsampled each numeral to fit in a 64×64 pixels box. The average error rate obtained in 20 random experiments was $1.8\% \pm 0.5\%$. (At least one of our errors was due to mislabeling in the database.) For comparison, with ordinary shape moments of up to order 8 (including the two global features, yielding 44 features), we obtained an error rate of $4.2\% \pm 0.6\%$. Further experiments with moments up to all orders between 4 and 20 (resulting in 27-230 features) yielded even inferior performance. In addition, our results favorably compare to the $3.58\% \pm 0.13\%$ reported in [25].

7.1.3 MNIST Database

We then applied the same procedure to numerals from the MNIST database [33]. The average error rate obtained in 20 random experiments was $2\% \pm 0.3\%$. Our results are superior to many of the algorithms tested in the benchmark [34], including RBF, Neural nets, PCA, and linear classifiers, but are inferior to the best results published for boosted LeNet-4 [34] (0.7 percent), shape context [5] (0.63 percent), virtual SVM [15] (0.56 percent), local context and nonlinear deformation models [29] (0.43 percent), and convolutional neural networks [51] (0.4 percent). However, our algorithm is fast. For example, our nonoptimized Matlab implementation requires less than 15 minutes of training with 20 K numerals.

Moreover, our method is general and can be applied to a wide variety of shape types. However, we believe that a task-driven feature selection procedure and enrichment of the database with distorted versions of the numerals as in [15], [51] might help improve our classification results in the future.

7.2 Natural Silhouettes

The next set of experiments, experiment demonstrates the utility of the Poisson-based features in classification and retrieval of general shapes. We collected a database of silhouettes of natural objects and expanded it with the most variable classes from [46]. The database contained 490 silhouettes of 12 classes (see Fig. 15). We centered the silhouettes about their centroid and brought them to a uniform scale (using $\sqrt{< x^2 > + < y^2 >}$ as the normalization factor) and then solved the Poisson equation (3). We characterized every silhouette using two measures. The first measure is analogous to L_θ (22). It identifies vertical and horizontal regions of a shape by detecting points for which the orientation computed with the Hessian matrix (Section 4.4) is close to either zero or $\pi/2$. This measure is defined by

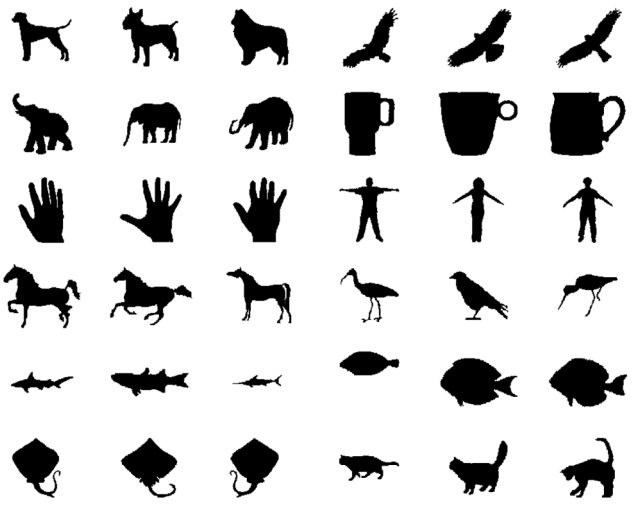


Fig. 15.A collection of silhouettes from the database.

$$H_\theta(x,y) = e^{-\gamma|\theta - |\alpha(x,y)||}, \tag{23}$$

where $\alpha(x,y)$ is the principal direction felt locally at (x,y) ($\frac{-\pi}{2} \leq \alpha(x,y) \leq \frac{\pi}{2}$) and γ is a constant (we used $\gamma = 3$). We evaluated H_θ at every point with ratio $\sqrt{|\lambda_{\min}/\lambda_{\max}|} \leq \frac{1}{2}$ for the two orientations $\theta \in \{0, \frac{\pi}{2}\}$. The second measure $K_\Phi(x,y)$ identifies concave regions as well as elongated convex sections by emphasizing points with high values of Φ (Section 4.1). Denote by $\hat{\Phi}(x,y)$, the function $\Phi(x,y)$ centered about its saddle point value (the value at the point where U is maximal) and normalized so that its maximal absolute value is 1. We define

$$K_\Phi(x,y) = \frac{1}{1 + e^{-\delta\hat{\Phi}(x,y)}} \tag{24}$$

(we used $\delta = 4$).

We then computed moments up to order 3 as in (21), substituting, for $w(x,y)$, the two vertical and horizontal measures H_θ and the measure K_Φ resulting in 10 moments per measure and 30 moments in total.

With this database of natural silhouettes, we performed two types of experiments: shape classification using the decision trees classifier described in Section 6 and shape retrieval experiments using the nearest neighbors technique. We compare our results with results obtained using the same classification/retrieval methods when the features used include geometric shape moments (GM) (computed as in (21) with $w(x,y) = \chi(x,y)$, the characteristic function of the shape) and Zernike moments (ZM) (computed as described in [56]). In the retreival test, we further compare the performance of the Poisson-based shape descriptor with that of the contour-based and region-based shape descriptors adopted by the MPEG-7 International Standards, namely, the curvature scale space descriptor (CSSD) [40], [19] and the Angular Radial Transform descriptor (ARTD) [30], [19], [9].

7.2.1 Shape Classification

In the classification experiments, we used a random selection of 396 of the silhouettes for training and the remaining 94 silhouettes for testing. The error rate was averaged over 100 random experiments, each performed on a random split of the database into training and test sets. Using the same classification algorithm as with the numerals (constructing 30 trees, each trained with a random selection of 322 silhouettes), we obtained an average error rate of $4.3\% \pm 2\%$. With geometric shape moments (GM) of orders up to 6 resulting in 25 features (not including the first three trivial moments), we obtained $6.2\% \pm 2.5\%$. Experiments with moments of different orders (we tried all orders between 4 and 9, resulting in 12-30 features) yielded even inferior performance.

We also compared the performance of our Poisson-based shape descriptors with that of Zernike moments descriptors (ZMD). Since Zernike moments assume integration over the range of the unit circle, we applied a slightly different shape normalization that allows the majority of the shape pixels to fall within the unit circle. We centered each silhouette about its centroid and brought it to a uniform scale using $2\sqrt{< x^2 > + < y^2 >}$ as the normalization factor. Ignoring the portions of the shape that fell outside the unit circle, we then computed, for each shape, 36 Zernike moments up to order 12 (using only their magnitudes and normalizing by the mass of the shape, as described in [56]). Using the same classification algorithm with Zernike moments of orders up to 8, resulting in 25 features, we obtained an average error rate of $8.9\% \pm 2.7\%$. Experiments with Zernike moments of different orders (we tried all orders between 7 and 12, resulting in 20-49 features) yielded even inferior performance.

7.2.2 Shape Retrieval

To further test our Poisson-based descriptors, we used them in a retrieval experiment using the database of natural silhouettes. Again, we compare the performance of the Poisson-based shape descriptor with that of geometric moment descriptors (GMD) and Zernike moment descriptors (ZMD). In addition, we compare our results to the contour and region-based shape descriptors adopted by the MPEG-7 International Standards, namely, the curvature scale space descriptor (CSSD) [40], [19] and Angular Radial Transform descriptor (ARTD) [30], [19], [9]. In these experiments, the code for CSSD and ARTD extraction and retrieval (XMWin6_1) was downloaded from the MPEG7-XM software repository [41].

We used the same Poisson-based features (PF) as described in Section 7.2. Each feature was centered about its centroid over the entire database of shapes and brought to a uniform scale. We then gave each feature a weight inversely proportional to the ratio between its average within class variation and its between class variation and computed weighted euclidian distances between every two shapes in the database. For each shape, the 15 closest shapes were retrieved and average recall and precision rates were computed. Precision measures the accuracy of retrieval and is defined as the ratio of the number of relevant shapes retrieved to the total number of retrieved shapes. Recall measures the robustness of the retrieval performance and is defined as the ratio of the number of relevant shapes retrieved to the total number of relevant shapes in the database. Fig. 16 shows the examples of the retrieved shapes for randomly picked query shapes from each class. Erroneous shapes are shown in gray. Note that the retrieval is mostly accurate with some confusion mainly between the four legged animals. The same procedure was also performed with unweighted geometric moments (GMD) and Zernike moments (ZMD) descriptors. With the ZMD, we then used both, the weighted euclidian distance (WED) and the "city block distance" (CBD) (as described in [56]) to run the same shape retrieval experiment. The results are summarized in Fig. 17.

For further comparison, we then turned to MPEG-7 shape descriptors. We used the software provided by MPEG-7 to extract CSSD and ARTD shape descriptors and to perform shape retrieval within the same database of natural silhouettes. All the results are summarized in Fig. 17. It can be seen from the graph that the retrieval performance of the Poisson-based shape descriptor (PF) is superior to that of the GMD, ZMD, MPEG-7 CSSD, and ARTD shape descriptors. The relatively weak performance of MPEG-7 shape descriptors could possibly be explained by the fact that the source code of the XM software version is optimized in terms of representation (4 bit representation) and speed by sacrificing the accuracy.

8 Conclusion

Solutions to the Poisson equation provide rich descriptive information that can be used to compute useful properties of shape silhouettes. In this paper, we derived several such properties and described how to compute them efficiently using multigrid solvers. Many of the properties we derived are obtained from the Poisson solution simply by differentiation. Also, we have shown how we can use the solution to construct a scalar field whose thresholding decomposes an object into parts.

We demonstrated the utility of the extracted properties in shape classification and retrieval tasks applied to several databases of handwritten numerals as well as shapes of natural objects. Generally, our results favorably compare to those reported in the literature and to those obtained with other shapes descriptors (such as MPEG-7 standard shape descriptors) using the same classification/retrieval methods.

Finally, additional useful processes and recognition tools can be built based on the properties derived with the Poisson equation. In particular, based on the local orientation measure presented in Section 4.4, we may segment each numeral to its constituent strokes by identifying sections in which orientation changes smoothly. Each stroke

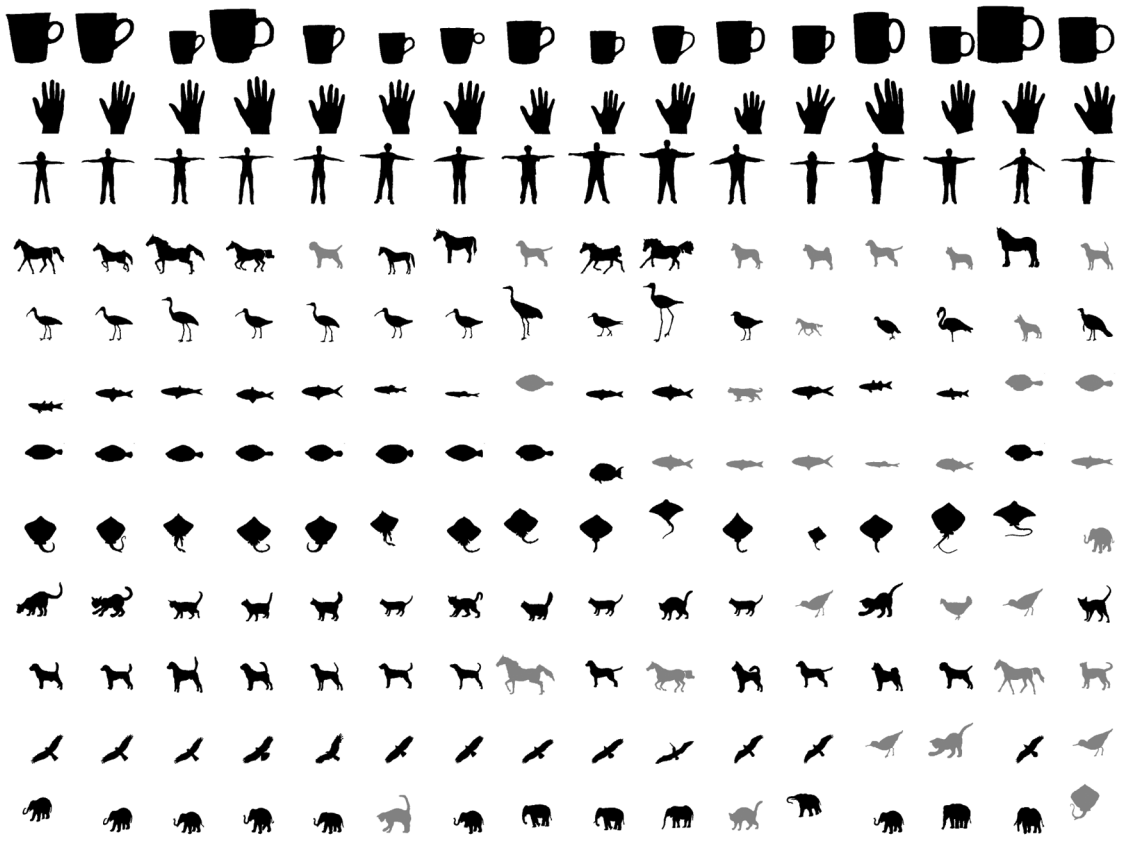


Fig. 16. Examples of the retrieved shapes for randomly picked query shapes from each class, sorted by ascending distance using the Poisson-based shape descriptors. The first column is the query shape and the retrieved shapes occupy the rest of the row.

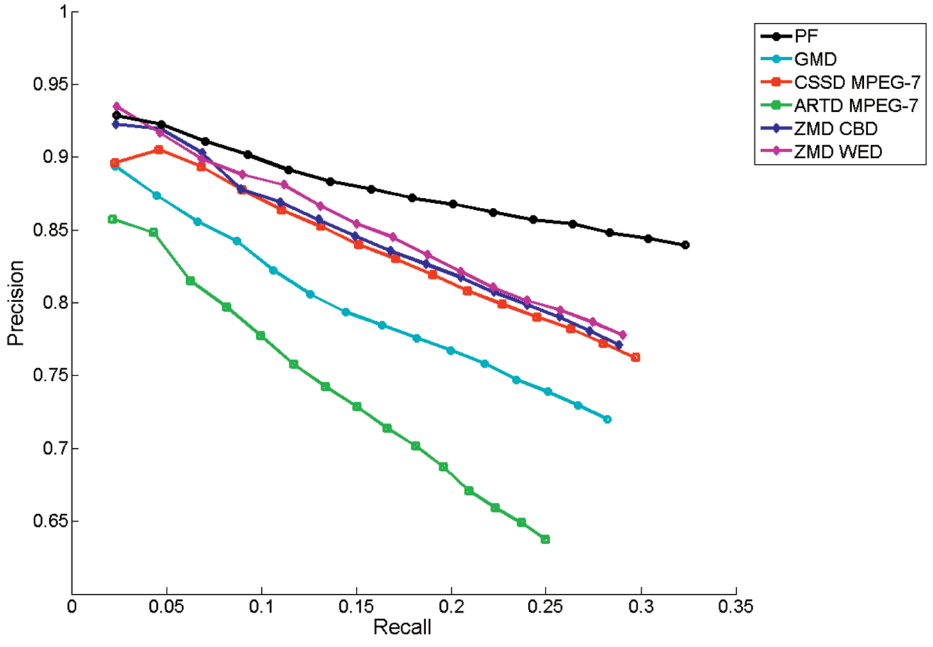


Fig. 17. Average Precision-Recall in shape retrieval experiments on the database of natural silhouettes. Several methods are compared, including Poisson-based features ($\mathbf{PF}$), geometric moments descriptor ($\mathbf{GMD}$), curvature scale space descriptor implemented by MPEG-7 ($\mathbf{MPEG-7CSSD}$), angular radial transform descriptor implemented by MPEG-7 ($\mathbf{MPEG-7ARTD}$), Zernike moments descriptor and using city block distance ($\mathbf{ZMDCBD}$), Zernike moments descriptor and using weighted euclidian distance ($\mathbf{ZMDWED}$).

can then be characterized by various properties, such as its location and several moments of its orientation as a function of arclength. These ideas can also be applied to identifying directions of skeleton parts. Finally, it is also worth noting that solutions to the Poisson equation and the properties extracted from its solution can also be applied with very

$$\uparrow_k^{k-1} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_k^{k-1} \qquad \uparrow_{k-1}^{k} = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_{k-1}^{k}$$

Fig. 18. The restriction (I_k^{k-1}, top) and bilinear interpolation (I_{k-1}^k, down) operators in stencil notation. (The stencil entries of the interpolation operator correspond to weights in a distribution process, therefore, the brackets are reversed.)

little change to objects in higher dimensions. A recent extension to 3D space-time analysis has been applied in the context of action recognition [7].

APPENDIX

MULTIGRID CYCLE

For a formal description of a multigrid cycle, we use a sequence of coarser and coarser girds Ω_{h_k}, characterized by a sequence of mesh sizes h_k: $\Omega_{h_M}, \Omega_{h_{M-1}}, \ldots \Omega_{h_0}$. The coarsest grid is characterized by the mesh size h_0, whereas the finest grid is characterized by $h = h_M$ and $\forall k,\ h_{k-1} = 2h_k$. To simplify notation, we replace the index h_k by k (for grids, grid functions, and grid operators) in the following.

As explained above, performing relaxation sweeps on a grid Ω_k very quickly reduces all high-frequency components of the error. So, it can be approximated on a coarser grid Ω_{k-1} with mesh size $h_{k-1} = 2h_k$. Generally, for any linear fine-grid equation $L^k u^k = f^k$ and any approximate solution $\tilde{u}^k$, the error $v^k = u^k - \tilde{u}^k$ satisfies the *residual equation* $L^k v^k = r^k$, where $r^k = f^k - L^k \tilde{u}^k$. It can therefore be approximated by the coarse-grid function v^{k-1} which satisfies

$$L^{k-1} v^{k-1} = \uparrow_k^{k-1} r^k, \tag{25}$$

where L^{k-1} is some coarse-grid approximation to L^k and $\uparrow_k^{k-1}$ is a fine-to-coarse restriction operator (defined in Fig. 18).

Having obtained an approximate solution $\tilde{v}^{k-1}$, it is used as a correction to the fine-grid solution. Namely, we replace $\tilde{u}^k$ by $\tilde{u}^k + \uparrow_{k-1}^k \tilde{v}^{k-1}$, where $\uparrow_{k-1}^k$ is a coarse-to-fine interpolation operator (Fig. 18).

To efficiently get an approximate solution to the coarse-grid equation (25), we employ the above solution process recursively at each scale k. This multiscale algorithm employs uniform grids at all scales. Given a fine grid, the coarse grid is obtained by simply taking a subset of every other point in both directions.

The complete recursive process, known as the multigrid cycle, is summarized below: The equation on a grid Ω_k is generally written as

$$L^k u^k = f^k, \tag{26}$$

with L^k denoting an appropriate discretization of the negative Laplace operator ($-\Delta$),

$$L^k u(i,j) \equiv \frac{1}{h_k^2}(4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}). \tag{27}$$

Unless k is the finest level ($k = M$), u^{k-1} is always designed to be the coarse correction to $\tilde{u}^k$ (the current approximation on the next finer grid) and, hence, $f^{k-1} = \uparrow_k^{k-1}(f^k - L^k \tilde{u}^k)$, starting with $f^M \equiv 1$. The *multigrid cycle* algorithm for improving a given approximate solution $\tilde{u}^k$ to (26) is defined recursively as follows:

$\tilde{u}^k \leftarrow \mathrm{MGC}(k, \tilde{u}^k, f^k)$

1. If the given grid consists of a small number of points—solve the equations directly, e.g., by Gauss elimination. Otherwise:
2. Perform ν_1 relaxation sweeps on (26), resulting in a new approximation $\bar{u}^k$.
3. Starting with $\tilde{u}^{k-1} = 0$, perform γ successive cycles of the type
$$\tilde{u}^{k-1} \leftarrow \mathrm{MGC}\big(k-1, \tilde{u}^{k-1}, \uparrow_k^{k-1}(f^k - L^k \bar{u}^k)\big).$$
4. Calculate $\hat{u}^k = \bar{u}^k + \uparrow_{k-1}^k \tilde{u}^{k-1}$.
5. Finally, perform ν_2 relaxation sweeps on (26), starting with $\hat{u}^k$ and yielding the final $\tilde{u}^k$. (We used $\nu_1 = \nu_2 = 2$ and $\gamma = 1$.)

Since the computational work at the increasingly coarser levels diminishes geometrically, the overall cost of a multigrid cycle is only a fraction more than the cost of the several fine-grid relaxation sweeps. Several such cycles are enough to produce a very accurate solution, so the total cost of the multigrid solver is just $O(n)$.

ACKNOWLEDGMENTS

This research was supported in part by the European Commission Project IST-2002-506766 Aim Shape and by the Binational Science foundation, Grant No. 2002/254. Ronen Basri was supported in part by the European Commission Project IST-2000-26001 VIBES. Achi Brandt was supported by the Israel Science Foundation grant No. 295/01, GIF Contract I-718-135.6/2001, and by the Israel Institute of Technology. The research was conducted at the Moross Laboratory for Vision and Motor Control at the Weizmann Institute of Science. The authors thank Yaara Goldschmidt for providing software for early experiments and Shachar Weis for his technical support.

REFERENCES

[1] S. Agarwal and D. Roth, "Learning a Sparse Representation for Object Detection," *Proc. European Conf. Computer Vision,* vol. 2, pp. 113-130, 2002.

[2] Y. Amit and D. Geman, "Shape Quantization and Recognition with Randomized Trees," *Neural Computation,* vol. 9, no. 7, pp. 1545-1588, 1997.

[3] J. August and S.W. Zucker, "Sketches with Curvature: The Curve Indicator Random Field and Markov Processes," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 4, pp. 387-400, Apr. 2003.

[4] R. Basri, L. Costa, D. Geiger, and D. Jacobs, "Determining the Similarity of Deformable Shapes," *Vision Research,* vol. 38, pp. 2365-2385, 1998.

[5] S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 4, pp. 509-522, Apr. 2002.

[6] I. Biederman, "Human Image Understanding: Recent Research and a Theory," *CVGIP*, vol. 32, pp. 29-73, 1985.

[7] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *Proc. IEEE Int'l Conf. Computer Vision*, 2005.

[8] H. Blum, "A Transformation for Extracting New Descriptors of Shape," *Proc. Symp. Models for the Perception of Speech and Visual Form*, pp. 362-380, 1967.

[9] M. Bober, "MPEG-7 Visual Shape Descriptors," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 1, no. 6, 2001.

[10] E. Borenstein, E. Sharon, and S. Ullman, "Combining Top-Down and Bottom-Up Segmentation," *Proc. IEEE Workshop Perceptual Organization in Computer Vision*, 2004.

[11] A. Brandt, "Interior Estimates for Second Order Elliptic Differential (or Finite-Difference) Equations via the Maximum Principle," *Israel J. Math.*, vol. 7, pp. 95-121, 1969.

[12] A. Brandt, "Multi-Level Adaptive Solutions to Boundary Value Problems," *Math. Computation*, vol. 31, pp. 333-390, 1977.

[13] W.L. Briggs, V.E. Henson, and S.F. McCormick, *A Multigrid Tutorial*, second ed. SIAM, 2000.

[14] S. Carlsson, "Order Structure, Correspondence and Shape Based Categories," *Proc. Int'l Workshop Shape, Contour and Grouping*, p. 1681, 1999.

[15] D. Decoste and B. Schölkopf, "Training Invariant Support Vector Machines," *Machine Learning*, vol. 46, no. 1-3, pp. 161-190, 2002.

[16] A. Douglis and L. Nirenberg, "Interior Estimates for Elliptic System of P.D.E," *Comm. Pure Applied Math.*, vol. 8, pp. 503-538, 1955.

[17] A. Duci, A.J. Yezzi, S.K. Mitter, and S. Soatto, "Shape Representation via Harmonic Embedding," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 656-662, 2003.

[18] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, first ed. Wiley, 1973,

[19] *Introduction to MPEG-7: Multimedia Content Description Interface*, B.S. Manjunath, P. Salembier, and T. Sikora, eds. Wiley, 2002.

[20] M. Elad, A. Tal, and S. Ar, "Content Based Retrieval of VRML Objects—An Iterative and Interactive Approach," *Proc. Sixth Eurographics Workshop Multimedia*, pp. 97-108, 2001.

[21] R. Fergus, P. Perona, and A. Zisserman, "Object Class Recognition by Unsupervise Scale Invariant Learning," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 264-274, 2003.

[22] M. Frenkel and R. Basri, "Curve Matching Using the Fast Matching Method," *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 35-51, 2003.

[23] Y. Gdalyahu and D. Weinshall, "Flexible Syntactic Matching of Curves and Its Application to Automatic Hierarchical Classification of Silhouettes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 21, no. 12, p. 1999, Dec. 1999.

[24] D. Geiger, K. Kumaran, and L. Parida, "A Computational View of Visual Organization for Figure/Ground Separation," *Proc. EEE Conf. Computer Vision and Pattern Recognition*, pp. 155-160, 1996.

[25] M. Giudici, F. Queirolo, and M. Valle, "Evaluation of Gradient Descent Learning Algorithms with Adaptive and Local Rate for Hand-Written Numerals," *Proc. European Symp. Artificial Neural Networks*, pp. 289-294, 2002.

[26] L. Gorelick, M. Galun, E. Sharon, A. Brandt, and R. Basri, "Shape Representation and Classification Using the Poisson Equation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 61-67, 2004.

[27] P. Grother, NIST Special Database 19, Handprinted Forms and Characters Database, http://www.nist.gov/srd/nistsd19.htm, 1995.

[28] D. Keysers, The USPS Database of Handwritten Digits. http://www_i6.informatik.rwth_aachen.de/ keysers/usps.html. YEAR?

[29] D. Keysers, C. Gollan, and H. Ney, "Local Context in Non-Linear Deformation Models for Handwritten Character Recognition," *Proc. Int'l Conf. Pattern Recognition*, vol. 4, pp. 511-514, 2004.

[30] W.-Y. Kim and Y.-S. Kim, "A New Region-Based Shape Descriptor," ISO/IEC MPEG99/M5472, 1999.

[31] E. Klassen, A. Srivastava, W. Mio, and S. Joshi, "Analysis of Planar Shapes Using Geodesic Paths on Shape Spaces," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 3, Mar. 2004.

[32] S.H. Lai and B.C. Vemuri, "An O(N) Iterative Solution to the Poisson Equation in Low-Level Vision Problems," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 9-14, 1994.

[33] Y. LeCun, "The MNIST Database of Handwritten Digits," http://yann.lecun.com/exdb/mnist/index.html. 2006.

[34] Y. LeCun, L. Botou, L. Jackel, H. Drucker, C. Cortes, J. Denker, I. Guyon, U. Muller, E. Sackinger, P. Simard, and V. Vapnik"Learning Algorithms for Classification: A Comparison on Handwritten Digit Recognition," *Neural Networks*, pp. 261-276, 1995.

[35] B. Leibe, A. Leonardis, and B. Schiele, "Combined Object Categorization and Segmentation with an Implicit Shape Model," *Proc. European Conf. Computer Vision '04 Workshop Statistical Learning in Computer Vision*, pp. 17-32, May 2004.

[36] J. Li and A. Hero, "A Spectral Method for Solving Elliptic Equations for Surface Reconstruction and 3D Active Contours," *Proc. IEEE Int'l Conf. Image Processing*, vol. III, pp. 1067-1070, 2001.

[37] C. Lin and R. Chellappa, "Classification of Partial 2-D Shapes Using Fourier Descriptors," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, no. 5, pp. 686-690, 1987.

[38] M. Maila and J. Shi, "A Random Walks View of Spectral Segmentation," *AI and STATISTICS*, 2001.

[39] D. Marr and H. K. Nishihara, "Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes," *Proc. Royal Soc., London*, vol. B200, pp. 269-294, 1978.

[40] K. Mokhtarian and M. Bober, *Curvature Scale Space Representation: Theory, Applications and MPEG-7 Standardisation*. Kluwer Academic (now Springer), 2003.

[41] "The MPEG-7 XM software repository," http://www.lis.ei.tum.de/research/bv/topics/mmdb/e_mpeg7.html, 2005.

[42] D. Mumford, *Elastica and Computer Vision, Algebraic Geometry and Its Applications*, Chandrajit Bajaj, ed., pp. 491-506. Springer Verlag, 1994.

[43] *Geometric Level Set Methods in Imaging, Vision, and Graphics*, S. Osher and N. Paragios, eds. Springer, 2003.

[44] A.P. Pentland, "Recognition by Parts," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 612-620, 1987.

[45] P. Perez, M. Gangnet, and A. Blake, "Poisson Image Editing," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 313-318, 2003.

[46] T.B. Sebastian, P.N. Klein, and B.B. Kimia, "Shock-Based Indexing into Large Shape Databases," *Proc. European Conf. Computer Vision*, vol. 3, pp. 731-746, 2002.

[47] T.B. Sebastian, P. Klien, and B.B. Kimia, "On Aligning Curves," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 1, pp. 116-125, Jan. 2003.

[48] E. Sharon and D. Mumford, "2D-Shape Analysis Using Conformal Mapping," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 350-357, 2004.

[49] K. Siddiqi and B. Kimia, "Parts of Visual from: Computational Aspects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 3, pp. 239-251, Mar. 1995.

[50] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, and S.W. Zucker, "Shock Graphs and Shape Matching," *Proc. IEEE Int'l Conf. Computer Vision*, p. 222, 1998.

[51] P.Y. Simard, D. Steinkraus, and J. Platt, "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis," *Proc. Int'l Conf. Document Analysis and Recognition*, pp. 958-962, 2003.

[52] T. Simchony, R. Chellappa, and M. Shao, "Direct Analytical Methods for Solving Poisson Equations in Computer Vision Problems," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 5, pp. 435-446, May 1990.

[53] U. Trottenberg, C. Oosterlee, and A. Schuller, *Multigrid*. Academic Press, 2001.

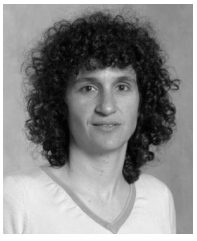
[54] S. Ullman, E. Sali, and M. Vidal-Naquet, "A Fragment-Based Approach to Object Representation and Classification," *Proc. Fourth Int'l Workshop Visual Form (IWVF4)*, 2001.

[55] L.R. Williams and D.W. Jacobs, "Stochastic Completion Fields: A Neural Model of Illusory Contour Shape and Salience," *Neural Computation*, vol. 9, no. 4, pp. 837-858, 1997.

[56] D. Zhang and G. Lu, "Evaluation of MPEG-7 Shape Descriptors against Other Shape Descriptors," *Multimedia Systems*, vol. 9, no. 1, pp. 15-30, 2003.

[57] M. Zuliani, C.S. Kenney, S. Bhagavathy, and B.S. Manjunath, "Drums and Curve Descriptors, " *Proc. British Machine Vision Conf.*, Sept. 2004.

Lena Gorelick received the BSc degree cum laude in computer science from Bar-Ilan University in 2001 and the MSc degree summa cum laude in computer science and applied mathematics from the Weizmann Institute of Science in 2004. She received the Weizmann Institute Excellence Award for her MSc thesis and is currently a PhD candidate in the Department of Computer Science and Applied Mathematics at the Weizmann Institute of Science. Her current research interests lie in computer vision, specifically in the area of object recognition and image segmentation.



Meirav Galun received the BSc degree in mathematics and computer science from Tel Aviv University in 1985, where she graduated summa cum laude. She received the MSc degree in applied mathematics from the Weizmann Institute of Science in 1992. She received the PhD degree in applied mathematics from the Weizmann Institute of Science in 1998, where she accepted an Excellency Award. She is currently a staff scientist associated with Professors Achi Brandt and Ronen Basri in the Applied Mathematics and Computer Science Department at the Weizmann Institute of Science. Her research interests are scientific computation, multiscale methods, computer vision, medical imaging, and data analysis.

Eitan Sharon received the BSc degree in mathematics from Tel-Aviv University in 1995, where he graduated cum laude. He received the PhD degree in computer science and applied mathematics from the Weizmann Institute of Science in 2002. From 2003 to 2004, he was a visiting research associate at Brown University in the Division of Applied Mathematics. He then held a postdoctoral fellowship at the Mathematical Sciences Research Institute in Berkeley in 2005 and consecutive postdoctoral fellowships at both the University of California, Berkeley and the University of California, Los Angeles during 2005-2006. Starting in October 2005, he held a faculty position in the Department of Electrical Engineering at the Technion-Israel Institute of Technology. His research interests include applied mathematics, scientific computation, and, in particular, computer vision. He is a member of the IEEE Computer Society.

Ronen Basri received the BSc degree in mathematics and computer science from Tel Aviv University in 1985, where he graduated summa cum laude. He received the PhD degree in computer science from the Weizmann Institute of Science in 1990. From 1990 to 1992, he was a postdoctoral fellow at the Massachusetts Institute of Technology in the Department of Brain and Cognitive Science and the Artificial Intelligence Laboratory under the McDonnell-Pew and the Rothchild programs. Since then, he has been affiliated with the Weizmann Institute of Science in the Department of Computer Science and Applied Mathematics, where he currently holds the position of associate professor. Between 1999 and 2000, he spent a sabbatical at the NEC Research Institute in Princeton, New Jersey. His research has focused on computer vision, especially in the areas of image segmentation, shape reconstruction, and object recognition. He is a member of the IEEE Computer Society.

Achi Brandt is a full professor at the Weizmann Institute of Science, where he served in the past as head of the Department of Pure Mathematics (1973-1975), head of the Department of Applied Mathematics (1978-1982), and director of the Gauss Center for Scientific Computation (1992-2003). He is a recipient of the Landau Prize in Mathematics (1978) and the Rothschild Prize in Mathematics (1990). Professor Brandt is a pioneer in research on multiscale computation, introducing fundamental computational approaches to diverse fields in mathematics, science, and engineering, including algebraic and partial differential equations, fluid dynamics, integro-differential equations, integral transforms, quantum mechanics, statistical mechanics, elementary particles molecular and macromolecular dynamic, bioinformatics, global optimizations, graph problems, medical imaging, and multiscale algorithms for various vision tasks, such as surface reconstruction, edge and fiber detection and completion, image segmentation, and recognition processes. He has published more than 150 scientific papers and organized many international workshops. In 2005, he received the highest international prize in scientific computation, the SIAM/ACM Prize in Computational Science and Engineering, cited for "pioneering modern multilevel methods, from multigrid solvers for partial differential equations to multiscale techniques for statistical physics" and "for influencing almost every aspect of contemporary computational science and engineering."