

Robust Normalization of Silhouettes for Recognition Applications

Javier Cortadellas^a, Josep Amat^b, Fernando de la Torre^c

^aElectronics Department, La Salle School of Engineering, Ramon Llull University
Pso. Bonanova 8, 08022 - Barcelona, Spain. Email: xavicort@salleURL.edu

^bDepartment of Automatic Control and Computer Engineering, Polytechnical University of Catalonia
C. Pau Gargallo 5, 08028 - Barcelona, Spain. Email: amat@esaii.upc.es

^cDepartment of Communications and Signal Theory, La Salle School of Engineering, Ramon Llull University. Pso. Bonanova 8, 08022 - Barcelona, Spain. Email: ftorre@salleURL.edu

In this work we introduce a new silhouette normalization method, which is robust with respect to deformations and enables the use of image domain based similarity measures for recognition applications. It is shown that a template matching operation can provide an accurate shortlist of candidates suitable for a more exact matching engine in spite of projective transformations and silhouette deformations. This retrieval of similar shapes has a low computational cost and has been tested with silhouettes of natural and man-made objects.

1. INTRODUCTION

Computer vision tries to provide artificial systems with a broader range of autonomy and action capabilities, but this can not be achieved without a key task in scene interpretation: object recognition.

The human visual system performs this task very efficiently processing several sources of information: visual object features (i.e. shape, color, texture, movement, 3D structure, etc...), context information (environment) and previous experience (learning). Among them, the silhouette, referred as the object profile and physical structure, is one of the most important features. Its perception is accomplished in the first steps of the recognition process [1] and enables the proposal of the first hypothesis about the object identity.

A computer vision system whose main goal is the recognition (or discrimination) of silhouettes has to solve first the way a silhouette will be represented. There are basically four ways: boundaries, regions, moments and structural representations. One of the simplest is to represent any arbitrary silhouette by the region it occupies. The benefits of this simple description are that it has a straight meaning for the human visual percep-

tion system, it avoids the computation of shape descriptors [2,3], which can be ambiguous and finally, there are specialized hardware architectures [4] that operate over the image domain at very high speeds.

The main drawback for this kind of representation is that it needs to be normalized with respect to projective transformations (translation, rotation and scaling) in order to be able to associate the silhouette to the correspondent model in recognition applications [5].

This normalization is difficult when one must deal with silhouettes that can vary. This change, that will be called deformation, can be due to multiple reasons: the object that projects de silhouette modifies its structure (in the case of articulated and deformable objects), the object is viewed from a different perspective, or two objects from a same class but slightly different.

In this work, a new robust normalization algorithm has been developed for estimating the localization, orientation and scale of deformable shapes [6]. It has enabled the use of image domain based similarity measures such as the normalized correlation factor for recognizing silhouettes that vary to a certain extent.

2. NORMALIZATION OF SILHOUETTES. TRADITIONAL APPROACH

One of the most popular approaches for normalizing silhouettes with respect to projective transformations involves the computation of the center of gravity, the orientation of the principal axis and finally, the size of the bounding box [2,7–9].

2.1. Principal Axis Based Orientation

Different methods have been used to determine the orientation of a shape like the Feret’s diameter (line between the two points on the periphery that are farthest apart), or the major axis of an ellipse fitted to the silhouette contour [3]. Among these methods, those that take into account all of the pixels of the shape have often worked better than those based on contour or boundary representations because they are less influenced by the presence or absence of a single pixel around the periphery.

In the principal axis method, one of the most widely used, orientation is defined as the angle of the axis of the least moment of inertia [7]. This axis corresponds to the line about which it takes the least amount of energy to spin an object. It is called principal axis, and can be regarded as the line that ‘best fits’ the silhouette. It is obtained by minimizing, with respect to the angle θ , the sum of the squared distances of each point $\vec{d}_i = (x_i, y_i)$ of the silhouette S to an axis that, passing through the coordinates (α, β) , has a slope $\tan \theta$:

$$\min_{\{\theta, \alpha, \beta\}} \sum_{(x,y) \in S} [(x - \alpha) \sin \theta - (y - \beta) \cos \theta]^2 \quad (1)$$

This minimization results in [7,8]:

$$\theta = \frac{1}{2} \tan^{-1} \left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \right) \quad (2)$$

$$\alpha = \bar{x} \quad \beta = \bar{y} \quad (3)$$

where μ_{jk} are the central moments of the silhouette and (\bar{x}, \bar{y}) are the coordinates of its center of gravity, computed as follows:

$$\bar{x} = \frac{\sum_{(x,y) \in S} x}{\sum_{(x,y) \in S} 1} \quad \bar{y} = \frac{\sum_{(x,y) \in S} y}{\sum_{(x,y) \in S} 1} \quad (4)$$

Being I a binary-valued picture, and $S = \{(x, y) \mid I(x, y) = 1\}$ the set of pixels representing a two-dimensional silhouette, for each pair of nonnegative integers (j, k) , the above central moments of S are given by [7]:

$$\mu_{jk} = \sum_{(x,y) \in S} (x - \bar{x})^j (y - \bar{y})^k \quad (5)$$

Moments can be given a physical interpretation by regarding S as composed of a set of point masses located at the points (x, y) , and thus providing useful information about the spatial arrangement of the points of S . A possible normalization of S with respect to rotations consists of rotating the silhouette in order to align its principal axis with a standard orientation, for example vertically [8].

The main problem of this principal axis approach is that, although it tolerates small variations on the boundary of the silhouette (i.e. due to the binarization process), it is quite sensitive to silhouette deformations, as it can be seen in figure (1).

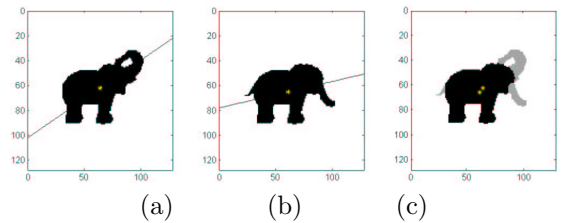


Figure 1. Centroid and Principal Axis orientation differences when a deformation occurs: (a) $\theta_1 = 30.47^\circ$, centroid = (64, 61). (b) $\theta_2 = 13.92^\circ$, centroid = (61, 65). (c) Overlapping of the two previous shapes, showing the differences.

This example shows two instances of the same class of object *elephant viewed laterally*. Despite of the fact that both silhouettes clearly represent the same object, there is a difference of more than 16 degrees between the principal axis of each figure. This behavior is very undesirable, specially

if, as it can be seen in figure (1.c), both silhouettes have such a similar structure (in black, the shared area).

The position of the center of gravity \vec{m} is also affected by the silhouette deformation. As it can be shown in figure (1.c), there is a difference of 7 pixels in two silhouettes that occupy less than 75×75 pixels. The distance metric that has been used to compute this shift is the city-block distance [8,10], defined as $\delta_{cb}(\vec{m}_1, \vec{m}_2) = |\bar{x}_1 - \bar{x}_2| + |\bar{y}_1 - \bar{y}_2|$.

This variation has a negative influence in the performance of any normalization method which make use of the center of gravity to achieve translation invariant representations. Furthermore, it is important to notice that the orientation normalization is accomplished rotating a silhouette with respect to its center of gravity. This fact gives even more importance to the role that this point plays in the normalization process.

2.2. Bounding Box

Size normalization is a necessary step when one must compare and recognize silhouettes. In an image domain framework, this is accomplished resizing the image (or a subimage) that contains the silhouette to a fixed predefined size. In this work, the normalized image size chosen has been 128×128 pixels: it has a good trade-off between the level of detail that a silhouette can show and the computational cost associated with its processing.

Once the final size has been chosen, the problem now is how to fit different silhouettes in an image with these dimensions. This problem is usually solved by means of the bounding box [9], defined as the bounding rectangle with minimum area. If the normalized images used are square, the bounding square must be computed. The ratio between the original image and the size of the bounding box is then used to resize the image.

The weakness of using this approach is that it is very sensitive to silhouette deformations. Even a narrow spike sticking out of the region can vastly modify the size of the bounding box, and thereby produce silhouettes with very different scales. An example of this problem is shown in figure (2). We can see two silhouettes from a same object

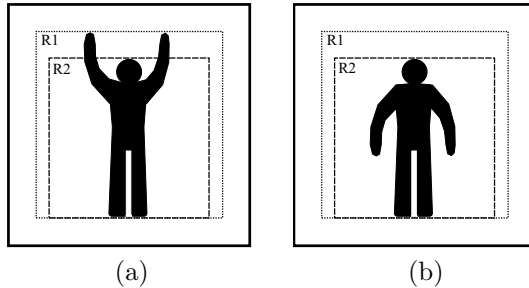


Figure 2. Two silhouettes of a same object which has changed its structure. Two regions are also shown ($R1$ and $R2$), which represents the subimages that could be used to resize the silhouette to a standard size.

which has changed its structure. The bounding rectangle in figure (2.a) is $R1$ and in figure (2.b) is $R2$. If we use these regions to normalize the silhouettes to the predefined size, similarity measures based for example in template matching operations could not be used for recognizing silhouettes.

On the other hand, if we choose the same region when processing each one of both silhouettes, then it would be possible to use image domain similarity measures (in case of using $R2$, the silhouette of the image (2.a) would be cropped, but even so, it would score a high similarity measure when it was compared with (2.b)).

The main problem is how to find, without having the reference of any other silhouette, which is the appropriate region for a particular silhouette. The final aim is to be able to recognize it successfully when it is compared with models of a database. Furthermore, the criterion used to take this decision must be robust with respect to admissible silhouette deformations.

It is important to remark the sentence ‘without having the reference of any other silhouette’: we are assuming that we have no access to the model database during the normalization process. The objective is not to compare for finding the appropriate size that optimizes a similarity measure, but to normalize first in order to reduce the com-

computational cost of the matching process when the model database is large. A registration algorithm between the original silhouette and each model would then not be a feasible solution.

3. ROBUST NORMALIZATION OF SILHOUETTES

The objective of this new method is to reduce the sensitivity of the localization, orientation and scale estimation with respect to silhouette deformations. It will then be possible to center and align in a similar manner different silhouettes that belong to a same object or class of objects.

3.1. Robust Localization and Orientation Estimation

The reason for the differences in the estimations shown in figure (1) is the change in a region on the periphery of the silhouette (in this case, the position of the elephant’s trunk). As the principal axis method is based on a least squares minimization, changes in regions far away from the axis have a great influence over the final estimation. A possible solution to reduce this sensitivity consists of decreasing the importance of the periphery when computing the principal axis, because this is the region that most varies when a deformation occurs. To achieve this goal and be able to control the influence of this region, robust statistics has been used.

Robust estimation has been successfully applied in many computer vision, signal processing and statistical problems [11–14]. There are many robust procedures, such as Least Median Squares, Ransac, M-estimator, etc, (see [15,16] for a review), which improve the estimation of parametric and non-parametric models when there are outliers in the data set. Among all the techniques, M-estimator is one of the best known. It has been used in this paper because its trade-off between computational cost and its percentage of outliers that is able to reject (breakdown point) [15].

Before applying an M-estimator to this problem, it is convenient to take into account the following fact: it can be shown that the orientation computed in equation (2) coincide with the direction of the eigenvector corresponding to the larger

eigenvalue of this matrix:

$$C = \begin{pmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{pmatrix} \quad (6)$$

which can also be seen as the covariance matrix of a distribution of points and be rewritten as:

$$C = \sum_{i=1}^n (\vec{d}_i - \vec{m})(\vec{d}_i - \vec{m})^T \quad (7)$$

where $\vec{d}_i = (x_i, y_i)^T$ are the coordinates of the pixels of the silhouette S , n the number of pixels of S and $\vec{m} = (\bar{x}, \bar{y})^T$ its center of gravity.

This fact allows us to use principal component analysis (PCA) to find the orientation of silhouettes. As it is only necessary to compute the first component, it is possible to use the approach followed in [17] and pose the computation of this direction as the following energy minimization problem:

$$\min_{\{\vec{v}, c_i\}} \sum_{i=1}^n \left\| \vec{d}_i - \vec{m} - \vec{v} \cdot c_i \right\|_2^2 \quad (8)$$

where \vec{v} is the vector that indicates the desired direction and c_i are the linear coefficients that minimize the projection error. The center of gravity \vec{m} of the data can be previously computed using equation (4).

The minimization of this function is not robust to outliers, because it is also based on a quadratic norm. M-estimators avoid the excessive influence of these points replacing the quadratic norm by a ρ -function, which limits the value of the residue of the outliers and reduce its contribution to the final solution. A ρ -function commonly used in computer vision is the German & McClure’s function [11]:

$$\rho_{GM}(x, \sigma) = \frac{x^2}{\sigma^2 + x^2} \quad (9)$$

where x is the residue of the data constraint and σ is the scale parameter, which controls what will be regarded as an outlier. Thus, the new energy function to be minimized is:

$$\min_{\{\vec{v}, \vec{m}, c_i\}} \sum_{i=1}^n \rho_{GM}(\|\vec{r}_i\|_2, \sigma) \quad (10)$$

where $\vec{r}_i = \vec{d}_i - \vec{m} - c_i \cdot \vec{v}$ is the residual vector and σ the scale parameter. There is no close form to solve this non-linear minimization problem[7], but the solution can be found by means of an iterative process. The algorithm commonly used is the IRLS (Iterative Reweighted Least-Squares). It is easy to show [12,18,19] that the minimum of equation (10) coincides with the minimum of:

$$\min_{\{\vec{v}, \vec{m}, c_i\}} \sum_{i=1}^n w_i \left(\vec{r}_i^T \cdot \vec{r}_i \right) \quad (11)$$

when the weights w_i are:

$$w_i = \frac{\sigma^2}{(\vec{r}_i^T \cdot \vec{r}_i + \sigma^2)^2} \quad (12)$$

The partial derivatives of the equation (11) with respect to the parameters are zero if we have reached its minimum. Taking into account this necessary condition and following [7,18], a possible iterative algorithm to solve the problem is:

1. Choose an initial approximation (for example, the solution based on the principal axis and the center of gravity) for $\vec{v}(0)$, $\vec{m}(0)$ and take $c_i(0) = \frac{\vec{v}(0)^T \cdot (\vec{d}_i - \vec{m}(0))}{\vec{v}(0)^T \cdot \vec{v}(0)}$ for $i = 1, \dots, n$.
2. Iterate. With $\vec{v}(k)$, $\vec{m}(k)$ and $c_i(k)$ in the iteration (k), find the solution for the iteration ($k+1$) as follows:
3. Find the residues $\vec{r}_i(k)$ and the value of the weights $w_i(k)$ as it is stated in equation (12).
4. Compute:

$$\vec{m}(k+1) = \frac{\sum_{i=1}^n w_i(k) (\vec{d}_i - c_i(k) \cdot \vec{v}(k))}{\sum_{i=1}^n w_i(k)} \quad (13)$$

$$c_i(k+1) = \frac{\vec{v}(k)^T \cdot (\vec{d}_i - \vec{m}(k))}{\vec{v}(k)^T \cdot \vec{v}(k)} \quad (14)$$

$$\vec{v}(k+1) = \frac{\sum_{i=1}^n w_i(k) \cdot c_i(k) \cdot (\vec{d}_i - \vec{m}(k))}{\sum_{i=1}^n w_i(k) \cdot c_i^2(k)} \quad (15)$$

5. Finish the iterative process if the difference in orientation between $\vec{v}(k+1)$ and $\vec{v}(k)$ is less than ε_d degrees and $\|\vec{m}(k+1) - \vec{m}(k)\|_2$ is less than ε_p pixels.

The scale parameter σ is estimated each iteration using the Median Absolute Deviation (MAD) [19,20] of the residues. The MAD can be regarded as a robust statistical estimation of the standard deviation, and it is computed as follows:

$$\sigma = 1.4826 \text{ med}_i \{ |(\|\vec{r}_i\|_2 - \text{med}_j(\|\vec{r}_j\|_2))| \} \quad (16)$$

where *med* is the median operator.

The computational cost of each update is $\mathcal{O}(n)$ and each one of the n points has only two coordinates. It is important to mention that the robust localization of the silhouette has been achieved due to the inclusion of the center of gravity \vec{m} as a parameter of the residue \vec{r}_i , and its estimation together with the other parameters. Once all the parameters are available, the silhouette is rotated with respect to the ‘robust’ center of gravity \vec{m} in order to align the vector \vec{v} vertically (this is the standard orientation chosen in this work).

3.2. Shape Dependent Weighting Function

The previous section presents a method for weighting the contribution of each silhouette pixel to its principal axis computation, in order to become more robust with respect to deformations. This weighting is dealt by means of w_i .

However, the computation of w_i only depends on the magnitude of the residue (see equation (12)). This means that the influence of a pixel over the orientation computation is only based on its distance to the axis. This is desirable, but it does not penalize those pixels that are more likely to vary due to a deformation, i.e. the periphery.

In order to cope with these situation, it is necessary to add more meaning to the weights, taking into account information about the shape of the silhouette: the original weights w_i are modified by multiplying them by a shape dependent weighting function $(P(\vec{d}_i))^\gamma$, whose value depends on the location of each pixel \vec{d}_i in the silhouette S . These new weights Ω_i are:

$$\Omega_i = w_i \cdot (P(\vec{d}_i))^\gamma \quad (17)$$

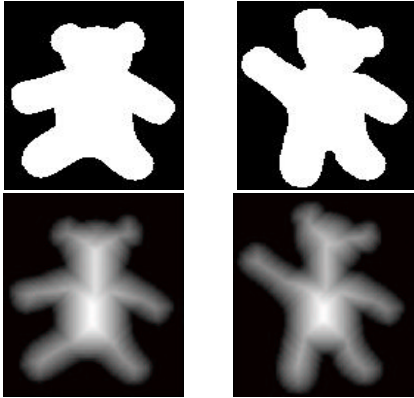


Figure 3. Above, two shapes of the same deformable object class "teddy bear". Below, their corresponding shape dependent weighting functions $(P(\vec{d}_i))^\gamma$ with $\gamma = 1$, showing that the parts more likely to vary (arms and legs) have smaller associated weights.

where γ is a constant factor that controls the influence of the periphery over the principal axis estimation.

A possible candidate for function $(P(\vec{d}_i))^\gamma$, assuming that major alterations occur on the periphery of silhouettes (as it can be seen in figure (7)), could be the minimum normalized distance of each pixel \vec{d}_i to the silhouette contour:

$$P(\vec{d}_i) = \frac{Q(\vec{d}_i)}{\max_j \{Q(\vec{d}_j)\}} \quad (18)$$

with:

$$Q(\vec{d}_i) = \min_k \left\{ \delta_{cb}(\vec{d}_i, \vec{d}_k) \mid \vec{d}_k \in S_B \right\} + 1 \quad (19)$$

where $\vec{d}_i = (x_i, y_i)^T \in S$ and $\vec{d}_k = (x_k, y_k)^T \in S_B$, being S_B the set of silhouette pixels that form its contour. The metric used for defining the distance between two pixels is the city-block distance [8,10], defined as:

$$\delta_{cb}(\vec{d}_i, \vec{d}_k) = |x_i - x_k| + |y_i - y_k| \quad (20)$$

There are efficient algorithms for computing distance maps that can be used for generating

the shape dependent weighting function (see figure (3) for an example). In this paper, instead of using an iterative method [8], the algorithm proposed in [10] has been applied, which lets equation (19) to be computed in just two scans of the binary image. The first pass requires a standard left-right top-bottom raster scan order, and the second pass requires a reverse right-left bottom-top scan order.

During the first scan, the partial result $Q_1(x, y)$ is computed using the silhouette contour image $I_c(x, y)$ of the original image $I(x, y)$, and the previously computed values of $Q_1(x, y)$, in the following manner:

$$Q_1(x, y) = \min\{Q_1(x, y-1), Q_1(x-1, y)\} + 1 \quad (21)$$

if $I_c(x, y) = 0$, and 0 otherwise, for $x = 1 \dots s$ and $y = 1 \dots t$. We consider that $Q_1(x, -1)$ and $Q_1(-1, y)$ have a value greater than $\max\{s, t\}$, where $s \times t$ is the size of the image. The second scan, now over the partial result $Q_1(x, y)$, is performed as follows:

$$Q_2(x, y) = \min\{Q_1(x, y), Q_1(x, y+1), Q_1(x+1, y)\} + 1 \quad (22)$$

considering that $Q_1(s+1, y)$ and $Q_1(x, t+1)$ have also a value greater than $\max\{s, t\}$. The final result $Q(x, y)$ is achieved multiplying each pixel of $Q_2(x, y)$ by its correspondent pixel in the original image $I(x, y)$.

The denominator of equation (18) lets normalize this function with respect to silhouette size. This normalization, together with equation (16), makes the computation of localization and orientation invariant with respect silhouette scale changes.

The algorithm shown in page 5 must be modified: in the third point, the computation of the Ω_i weights (following equation (17)) must be added. In all the of equations of the fourth point, weights w_i must also be substituted by the new weights Ω_i . The computational cost of the whole algorithm only increases in n multiplications in each iteration, because once $(P(\vec{d}_i))^\gamma$ has been computed, it does not vary during the whole process.

3.3. Direction ambiguity

The vector \vec{v} let us align a silhouette vertically. However, the direction of this vector depends on the initial solution of the robust localization and orientation algorithm. Hence, there is a 180 degrees ambiguity when it comes to rotate the silhouette.

This problem has been solved by means of a criterion: the vector \vec{v} must point towards the half plane that contains the region of the silhouette with larger weighted area. If this is not true at the end of the iterative algorithm, then vector \vec{v} is multiplied by -1 . Finally, the silhouette is rotated so that \vec{v} points upwards.

The line that splits the image plane in two parts passes through the center of gravity and it is perpendicular to the orientation defined by \vec{v} . It is easy to show that $\sum_{i=1}^n \frac{1}{2} (\text{sgn}(c_i) + 1)$ is the number of silhouette pixels which lies in the half plane pointed by \vec{v} , where $\text{sgn}(\cdot)$ indicates the sign function. The computation of the partial weighted area is achieved multiplying each pixel by its associated weight Ω_i .

The procedure consists of computing the value of z , defined as:

$$z = \frac{\sum_{i=1}^n (\text{sgn}(c_i) + 1) \cdot \Omega_i}{\sum_{i=1}^n (-\text{sgn}(c_i) + 1) \cdot \Omega_i} \quad (23)$$

If $z < 1$, it means that \vec{v} must be modified, as it has been previously described.

3.4. Size adjustment

The last step in the normalization process is size normalization, and its main goal is to resize silhouettes so that they can be compared in the image domain. Since the technique proposed in this work does not warrant the preservation of the whole silhouette, it has been called referred as adjustment instead of normalization. It consists of scaling a silhouette up or down in order to have a fixed predefined area. Due to the discrete nature of digital images, this premise will not be exactly fulfilled, but this fact does not affect significantly the performance of the method described below.

The area of a silhouette is defined by its number of pixels, which has been considered n . Hence, the

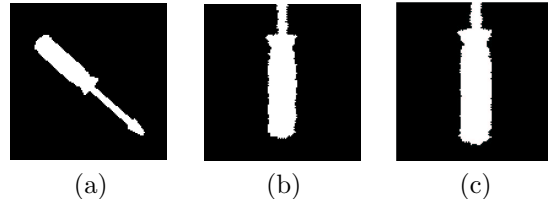


Figure 4. (a) Original silhouette (256×256 pixels). (b) and (c) correspond to the same silhouette once it has been normalized with respect to translation, rotation and scale. Image size is 128×128 pixels and the parameters used are $\gamma = 2$ and $A_R = 2500$ (b) and $A_R = 3000$ (c).

scale factor E_A can be computed as:

$$E_A = \sqrt{\frac{A_R}{n}} \quad (24)$$

where A_R is a constant which indicates the area that all the silhouettes should have at the end of the process. Resize must be performed so that the robust center of gravity of the silhouette lies on the central position of the normalized image.

An example is shown in figure (4). Depending on the value of A_R and the compactness of the shape, some silhouettes are cropped (the elongated ones), but they generally retain enough information to be recognized. If A_R is small, there would be no loss of information, but on the other hand, compact silhouettes would have a very small size, losing shape details. Thus, there is a trade-off when choosing the value of A_R . Typically A_R varies between 15% to 25% of the size of the final image.

4. SIMILARITY MEASURE

The similarity measure used to compare silhouettes is based on the chamfer matching algorithm [10,21] and the template matching operation. Silhouette contour convey a lot of visual information, enough in many cases to recognize the object from which it comes. One of the problems when two contours must be compared is the pixel correspondence. Chamfer matching algorithms avoids

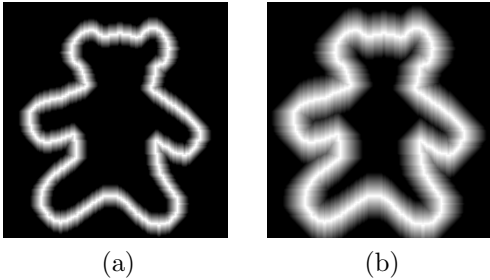


Figure 5. Example of $F(k, l)$ for the teddy bear of figure (3): (a) $\phi(d) = -255 \cdot d/6 + 255$ if $0 \leq d \leq 6$ and $\phi(d) = 0$ si $d > 6$. (b) $\phi(d) = -255 \cdot d/12 + 255$ if $0 \leq d \leq 12$ and $\phi(d) = 0$ if $d > 12$.

it assuming that it is unknown. They compute the minimum distance between each pixel of a contour and all the pixels of the model (usually with a distance map) and try to find the transformation that minimizes the sum of distances. In [21] for example, a stochastic search process is proposed to find such transformation.

This iterative process can be computational expensive. In this work, this cost is reduced due to the use of a template matching operation for measuring similarity between contours. This matching method can be used because of the excellent performance of the normalization algorithm previously described. However, this operation is drastically affected if both contours do not overlap exactly.

In order to increase the robustness of the measure in front of misalignments or quantization errors, the normalized silhouette contour image is transformed, propagating and weighting the contour. The contour propagated image, named $F(k, l)$, can be generated, taking $Q_2(k, l)$ from equation (22), as follows:

$$F(k, l) = E[\phi(Q_2(k, l) - 1)] \quad (25)$$

where $E[x]$ indicates the integer value of x , and $\phi(d)$ is a weighting distance function, positive defined and monotonically decreasing, with a maximum value of 255 for $d = 0$. Figure (5) shows two examples of $F(k, l)$ using a linear $\phi(d)$ function.

The comparison is carried out between a contour propagated image which represents the silhouette to be recognized, and each one of the models $M_i(k, l)$ stored in a database ($i = 1 \dots N$). These models consist of binary images that contain the contour of a normalized silhouette.

The similarity measure τ_i is based on a normalized correlation factor, but it has been simplified because a binary image is involved and also because it is not necessary to compute the exact correlation value: the goal is to find which model has the highest similarity measure. Therefore, it has been defined as:

$$\tau_i = \frac{\sum_{k=0}^{s-1} \sum_{l=0}^{t-1} M_i(k, l) F(k, l)}{\sum_{k=0}^{s-1} \sum_{l=0}^{t-1} M_i(k, l)} \quad (26)$$

with a maximum value of 255.

The performance of the similarity measure heavily depends on $\phi(d)$ and the velocity with which it goes towards zero, relatively to the size of the image. If $\phi(d)$ decreases abruptly, it does not improve the robustness of the method. On the other hand, if it decreases smoothly, many different models can achieve a high matching value.

5. RESULTS

Several experiments have been carried out to measure the effectiveness of the robust normalization method and its validity for silhouette recognition applications. The first one has tested the robustness of the proposed localization and orientation algorithm with respect to deformations. Figure (6) shows 8 pairs of silhouettes that simulate several deformations. The parameters used have been $\varepsilon_d = 0.02$, $\varepsilon_p = 0.05$ and $\gamma = 2$ for an image size of 128×128 pixels.

Two parameters have been computed: θ_{DIF} is the difference between the orientation estimation of two silhouettes in the traditional (T) and robust method (R), and C_{DIF} is the difference between the center of gravity estimation of two silhouettes, using the city-block distance. The results in Table 1 clearly suggest that robust normalization is much less sensitive to deformations.

The robust normalization method has been applied to the set of 48 silhouettes shown in figure (7), whose size is 256×256 pixels. The difference

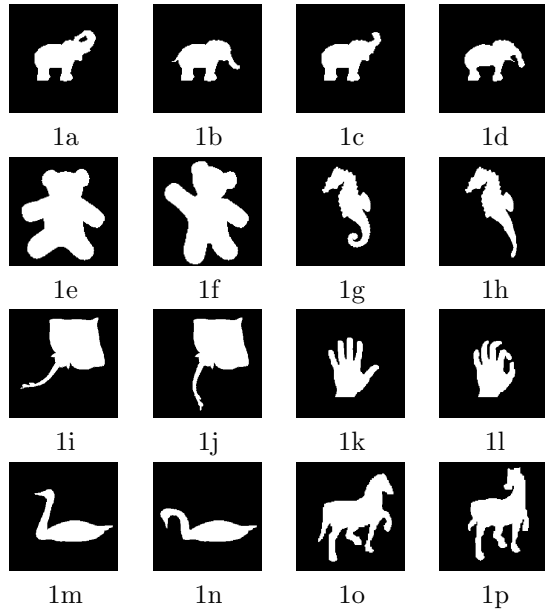


Figure 6. Silhouettes of 128×128 pixels used for testing the robustness of the localization and orientation estimation with respect to deformations.

between each pair of consecutive silhouettes is due to deformations, projective transformations or even to the fact that they belong to two different objects from a same class. Figure (8) shows the results, using $A_R = 2500$ and a normalized image size of 128×128 pixels. Instead of using individual images, it has been preferred to show the overlapping of each pair of consecutive normalized silhouettes, in order to visually evaluate the validity of the method.

The last experiment examines the performance of the similarity measure once the 48 silhouettes of figure (7) have been normalized with the robust algorithm described above. Each one of the silhouettes has been compared with the 47 remaining silhouettes (acting as models) using the similarity measure τ_i . In 34 of 47 possible cases, the highest value of τ_i corresponded to the expected model. This value increased to 46 if we counted how many times the expected model had achieved

Table 1

Results of applying the traditional (T) and robust (R) localization and orientation algorithm to the silhouettes of figure (6), with $\gamma = 2$.

	θ_{DIF}		C_{DIF}	
	T	R	T	R
1a/1b	16.64	1.33	12	3
1c/1d	12.54	0.28	3	2
1e/1f	13.55	3.71	10	3
1g/1h	6.21	0.07	6	2
1i/1j	29.10	0.00	3	0
1k/1l	2.29	0.11	2	0
1m/1n	10.17	0.00	12	2
1o/1p	18.32	4.53	7	5

one of the four highest similarity measures.

The weighted distance function $\phi(d)$ used in this experiment has been:

$$\phi(d) = \begin{cases} -255 \cdot d/\sigma + 255 & \text{if } 0 \leq d \leq \sigma \\ 0 & \text{if } d > \sigma \end{cases} \quad (27)$$

with $\sigma = 5$, for images of 128×128 pixels. It means that the width of the propagated contour is 9 pixels. Other functions can be used, but the linear is the one that has achieved better results.

6. CONCLUSION

This paper deals with the problem of normalizing and recognizing deformable silhouettes in the image domain. A traditional least squares framework is not capable of achieving acceptable normalization results when silhouettes change. Thus, it has been necessary to introduce robust statistics, as well as a shape dependent weighting function, to make the estimations much less sensitive to deformations. The excellent performance of the new normalization method has enabled the use of image domain based similarity measures, such as the normalized correlation. The approach is simple and easy to apply, with a low computational cost. The experiments have shown that this method is suited for a coarse silhouette recognition application. It can provide a shortlist of candidates in order to reduce the computational cost of other more expensive but accurate recognition methods.

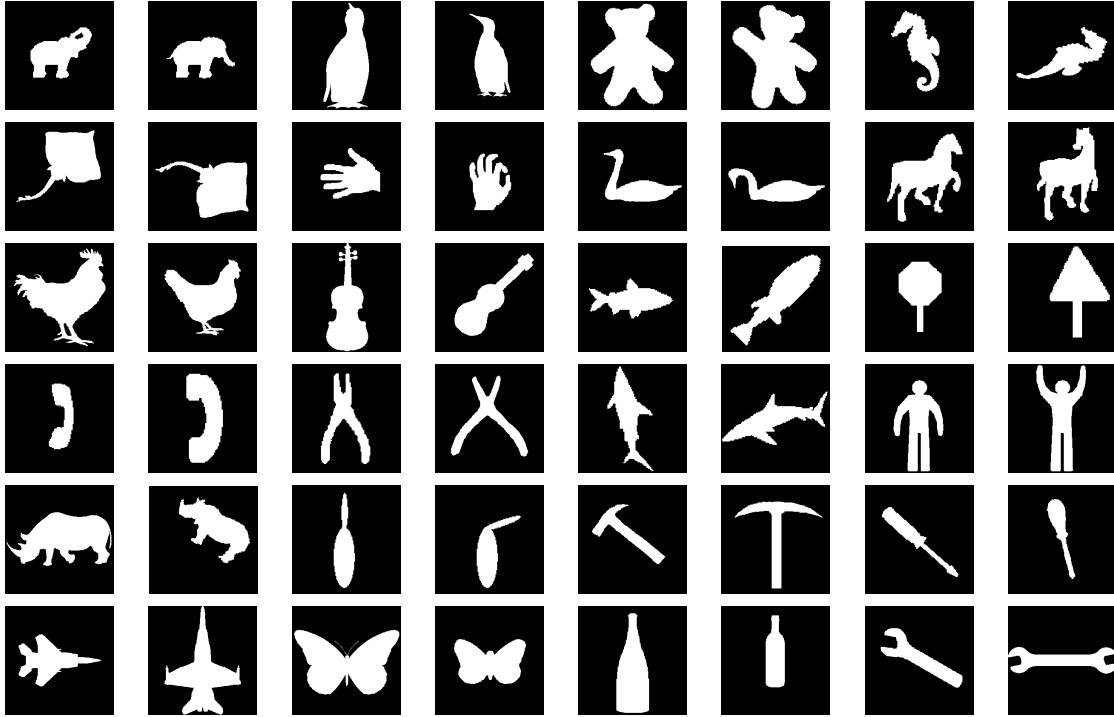


Figure 7. Silhouettes used in the experiments. Each pair of consecutive silhouettes represent the same object or class of objects.

The presented method is not robust to occlusions, specially if they affect the central part of the silhouettes. However, it can also be used for 3D data set registration applications. Algorithms such as ICP [22] heavily depend on the initial solution. The robust procedure here exposed can provide this first approximation.

REFERENCES

1. V. Bruce, P.R. Green, M.A. Georgeson. *Visual Perception. Physiology, Psychology, and Ecology*, 3rd Ed. Psychology Press, 1996.
2. A.K. Jain. *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.
3. J.C. Russ. *The Image Processing Handbook*, 2nd Ed., CRC Press, 1995.
4. J. Cortadellas, J. Amat. “Image Associative Memory”. 15th *Int. Conf. on Pattern Recognition*, Barcelona, 2000. Vol. 3, pp. 638–641.
5. R.C. Veltkamp, M. Hagedoorn. “State of the art in shape matching”. Tech. Report UU-CS-1999-27, Utrecht, 1999.
6. J. Cortadellas, J. Amat, M. Frigola. “Robust Normalization of Shapes”. 10th *Int. Conf. on Discrete Geometry and Computer Imagery*, Bordeaux, pp. 255– 266. 2002.
7. R. Haralick, L. Shapiro. *Computer and Robot Vision*, Vol.1 & 2. Addison Wesley, 1992.
8. A. Rosenfeld, A.C. Kak. *Digital picture Processing*, 2nd Ed., Academic Press, 1982.
9. P.L. Rosin. “Measuring Shape: Ellipticity, Rectangularity, and Triangularity”. *Int. Conf. Pattern Recognition*, (1) 952–955, 2000.
10. G. Borgefors. “Distance transformations in digital images”. *Computer Vision, Graphics, and Image Processing*, (34) 344–371, 1986.
11. M.J. Black, A.D. Jepson. “EigenTracking: ro-

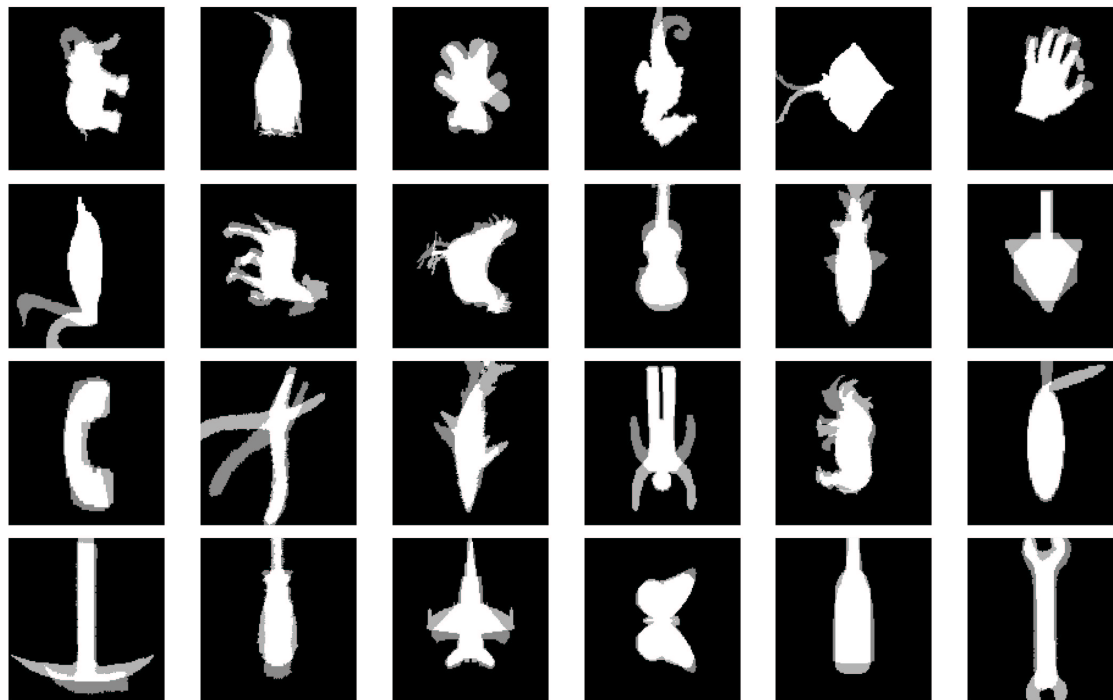


Figure 8. Overlapping of each pair of consecutive silhouettes of figure (7), once they have been normalized with the robust method.

- bust matching and tracking of articulated objects using a view-based representation”. *Int. J. Computer Vision*, (26):1, 63–84, 1998.
12. F. de la Torre, M. Black. “A Framework for Robust Subspace Learning”. Accepted for publication in *Int. J. Computer Vision*, 2002.
 13. R.L. Kashyap, J. Liaw. “Subset least squares method for robust speech and image processing”. *Int. J. Pattern Recognition and Artificial Intelligence*, (10): 5, 447–471, 1996.
 14. S.H. Lai. “Robust image alignment under partial occlusion and spatially varying illumination change”. *Computer Vision and Image Understanding*, (78),84–98, 2000.
 15. F. Hampel, E. Ronchetti, P. Rousseeuw, W. Stahel. *Robust Statistics: The Approach based on Influence Functions*, Wiley, 1986.
 16. D. Mintz, P. Meer. “Robust Estimators in Computer Vision: an Introduction to Least Median of Squares Regression”. *Artificial Intelligence & Computer vision*, Elsevier, 1991.
 17. L. Xu, A. Yuille. “Robust Principal Component Analysis by Self-Organizing Rules based on statistical physics approach”. *IEEE Trans. Neural Networks*, (6):1,131–143, 1995.
 18. G. Li. “Robust regression”. *Exploring Data, Tables, Trends and Shapes*, John Wiley, 1985.
 19. Z. Zhang. “Parameter estimation techniques: A tutorial with application to conic fitting”. *Image and Vision Computing*, 15(1), pp.59–76, 1996.
 20. P. Rousseeuw, A. Leroy. *Robust Regression and Outlier Detection*, John Wiley and Sons, 1987.
 21. M. Bengtsson. “Stochastic optimization algorithm - an application to pattern matching”. *Pattern Rec. Letters*, (11):11, 717-724, 1990.
 22. P. Besl, N. McKay. “A method for Registration of 3-D Shapes”. *IEEE Trans. Pat. Anal. Mach. Intell.*, Vol 14, n 2, pp. 239-356, 1992.