Published in final edited form as:
Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit. 2006 ; : 977–984.

Shape-Based Approach to Robust Image Segmentation using Kernel PCA

Samuel Dambreville, **Yogesh Rathi**, and **Allen Tannenbaum**
Georgia Institute of Technology, School of Electrical and Computer Engineering, Atlanta, GA, USA 30332

Samuel Dambreville: samuel.dambreville@gatech.edu; Yogesh Rathi: yogesh.rathi@gatech.edu; Allen Tannenbaum: tannenba@ece.gatech.edu

Abstract

Segmentation involves separating an object from the background. In this work, we propose a novel segmentation method combining image information with prior shape knowledge, within the level-set framework. Following the work of Leventon et al., we revisit the use of principal component analysis (PCA) to introduce prior knowledge about shapes in a more robust manner. To this end, we utilize Kernel PCA and show that this method of learning shapes outperforms linear PCA, by allowing only shapes that are close enough to the training data. In the proposed segmentation algorithm, shape knowledge and image information are encoded into two energy functionals entirely described in terms of shapes. This consistent description allows to fully take advantage of the Kernel PCA methodology and leads to promising segmentation results. In particular, our shape-driven segmentation technique allows for the simultaneous encoding of multiple types of shapes, and offers a convincing level of robustness with respect to noise, clutter, partial occlusions, or smearing.

1. Introduction

Segmentation consists of extracting an object from an image, an ubiquitous task in computer vision applications. It is quite useful in applications ranging from finding special features in medical images to tracking deformable objects; see [7, 14, 16, 17] and the references therein. The active contour methodology has proven to be quite valuable for performing this task. However, the use of image information alone often leads to poor segmentation results in the presence of noise, clutter or occlusion. The introduction of shape priors in the contour evolution process has been shown to be an effective way to address this issue, leading to more robust segmentation performances.

Many different methods which use a parameterized or an explicit representation for contours have been proposed [2, 15, 3]. In [4], the authors use the B-spline parametrization to build shape models in the kernel space [8]. These models were then used in the segmentation process to provide shape prior. The geometric active contour framework (GAC) (see [12] and the references therein) involves a parameter free representation of contours, i.e., a contour is represented implicitly by the zero level set of a higher dimensional function, typically a signed distance function [9]. In [7], the authors obtain the shape statistics by performing linear principal component analysis (PCA) on a training set of signed distance functions (SDFs). This approach was shown to be able to convincingly capture small

variations in the shape of an object. It inspired other schemes to obtain shape prior described in [14, 11], notably, where SDFs were used to learn the shape variations.

However, when the object considered for learning may undergo complex or non-linear deformations, linear PCA can lead to unrealistic shape priors, by allowing linear combinations of the learnt shapes that are unfaithful to the true shape of the object. Cremers *et al.*, successfully pioneered the use of kernel methods to address this issue, within the GAC framework [5]. In the present work, we propose to use Kernel PCA to introduce shape priors for GACs. Kernel PCA was presented by Scholkopf [8] and allows to combine the precision of kernel methods with the reduction of dimension in the training set. This is the first time, to our knowledge, that Kernel PCA is explicitly used to introduce shape priors in the GAC framework. In this paper, we also propose a novel intensity segmentation method, specifically tailored to allow for the inclusion of shape prior.

In the next section, we compare linear PCA to Kernel PCA, using SDFs and binary maps as representations of shapes. In Section 3, we propose an intensity-based energy functional in terms of binary shapes for separating an object from the background, in an image. These energies are qualitatively similar to the ones proposed by [1, 10] but quantitatively different. In Section 4, we present a robust segmentation framework, combining image cues and shape knowledge in a consistent fashion. The robustness of the proposed algorithm is demonstrated on various challenging examples, in Section 5.

2. Kernel PCA for shape prior

Kernel PCA can be considered to be a generalization of linear principal components analysis. This technique was introduced by Scholkopf [8], and has proven to be a powerful method to extract nonlinear structures from a data set. The idea behind Kernel PCA consists of mapping a data set from an input space $\mathcal{I}$ into a feature space F via a nonlinear function ϕ. Then, PCA is performed in F to find the orthogonal directions (principal components) corresponding to the largest variation in the mapped data set. The first l principal components account for as much of the variance in the data as possible by using l directions. In addition, the error in representing any of the elements of the training set by its projection onto the first l principal components is minimal in the least square sense.

The nonlinear map ϕ typically does not need to be known, through the use of Mercer kernels. A *Mercer kernel* is a function $k(.,.)$ such that for all data points χ_i, the kernel matrix $\mathbf{K}(i,j) = k(\chi_i, \chi_j)$ is symmetric positive definite [8]. It can be shown that using $k(.,.)$ one can obtain the inner scalar product in F: $k(\chi_a, \chi_b) = (\phi(\chi_a) \cdot \phi(\chi_b))$, with $(\chi_a, \chi_b) \in \mathcal{I}$.

We now briefly describe the Kernel PCA method [6, 8]. Let $\tau = \{\chi_1, \chi_2, \ldots, \chi_N\}$ be a set of training data. The centered kernel matrix $\tilde{\mathbf{K}}$ corresponding to τ, is defined as

$$\tilde{\mathbf{K}} = (\varphi(\chi_i) - \bar{\varphi} \cdot \varphi(\chi_j) - \bar{\varphi}) = (\tilde{\varphi}(\chi_i) \cdot \tilde{\varphi}(\chi_j)) = \tilde{\mathrm{k}}(\chi_i, \chi_j), for i \in [|1, N|] \quad (1)$$

with $\bar{\varphi} = \frac{1}{N}\sum_{i=1}^{N} \varphi(\chi_i)$, $\tilde{\phi}(\chi_i) = \phi(\chi_i) - \bar{\phi}$ being the centered map corresponding to χ_i and $\tilde{k}(.,.)$ denotes the centered kernel function. Since $\tilde{\mathbf{K}}$ is symmetric, using Singular Value Decomposition, it can be decomposed as

$$\tilde{\mathbf{K}} = \mathbf{U}\mathbf{S}\mathbf{U}^t \quad (2)$$

where $\mathbf{S} = \mathrm{diag}(\gamma_1, \ldots, \gamma_N)$ is a diagonal matrix containing the eigenvalues of $\tilde{\mathbf{K}}$. $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_N]$ is an orthonormal matrix. The column-vectors $\mathbf{u}_i = [u_{i1}, \ldots, u_{iN}]^t$ are the eigenvectors

corresponding to the eigenvalues γ_i's. Besides it can easily be shown that $\tilde{\mathbf{K}} = \mathbf{HKH}$, where $\mathbf{H}=\mathbf{I}-\frac{1}{N}\mathbf{1}\mathbf{1}^t$. $\mathbf{1} = [1, \ldots, 1]^t$ is an $N \times 1$ vector.

Let $\mathbf{C}$ denote the covariance matrix of the elements of the training set mapped by $\tilde{\phi}$. Within the Kernel PCA methodology, $\mathbf{C}$ does not need to be computed explicitly, only $\mathbf{K}$ needs to be known to extract features from the training set [13]. The subspace of the feature space F spanned by the first l eigenvectors of $\mathbf{C}$, will be referred to as the Kernel PCA space, in what follows. The Kernel PCA space is the subspace of F, obtained from learning the training data.

Let χ be *any* element of the input space $\mathcal{I}$. The projection of χ on the Kernel PCA space will be denoted by $P^l\phi(\chi)$[1]. The projection $P^l\phi(\chi)$ can be obtained as given in [8]. In the feature space F, the squared distance d_F^2 between a test point χ and its projection on the Kernel PCA space is given by [8]:

$$d_F^2[\varphi(\chi), P^l\varphi(\chi)]=\|\varphi(\chi)-P^l\varphi(\chi)\|^2=k(\chi,\chi)-2\varphi(\chi)^tP^l\varphi(\chi)+P^l\varphi(\chi)^tP^l\varphi(\chi)$$

Using some matrices manipulations, this squared distance can be expressed only in terms of kernels as:

$$d_F^2[\varphi(\chi), P^l\varphi(\chi)]=k(\chi,\chi)+\frac{1}{N^2}\mathbf{1}^t\mathbf{K}\mathbf{1}-\frac{2}{N}\mathbf{1}^t\mathbf{k}_\chi+\tilde{\mathbf{k}}_\chi^t\mathbf{M}\tilde{\mathbf{K}}\mathbf{M}\tilde{\mathbf{k}}_\chi-2\tilde{\mathbf{k}}_\chi^t\mathbf{M}\tilde{\mathbf{k}}_\chi \quad (3)$$

where, $\mathbf{k}_\chi = [k(\chi, \chi_1)\ k(\chi, \chi_2), \ldots, k(\chi, \chi_N)]^t$, $\tilde{\mathbf{k}}_\chi=\mathbf{H}(\mathbf{k}_\chi-\frac{1}{N}\mathbf{K}\mathbf{1})$ and $\mathbf{M}=\sum_{i=1}^{l}\frac{1}{\gamma_i}\mathbf{u}_i\mathbf{u}_i^t$.

2.1. Kernel for linear PCA

In [7], the authors presented a method to learn shape variations by performing PCA on a training set of shapes (closed curves) represented as the zero level sets of signed distance functions. Using the following kernel in the formulation of Kernel PCA presented above, amounts to performing Linear PCA on SDFs[2]:

$$k_{id}(\Phi_i,\Phi_j)=(\Phi_i.\Phi_j)=\int\int\Phi_i(u,v)\Phi_j(u,v)du.dv \quad (4)$$

for all SDFs Φ_i and $\Phi_j : \mathbf{R}^2 \mapsto \mathbf{R}$.

A different representation for shapes is to use binary maps, i.e., to set to 1 the pixels located inside the shape and to 0 the pixels located outside (see figure 1). One can change the shape representation from SDFs to binary maps using the Heaviside function $H\Phi=\begin{cases}1 & \Phi\geq 0,\\ 0 & \text{else.}\end{cases}$

Note that, in this case, the kernel allowing to perform linear PCA is given by $k_{id}(H\Phi_i, H\Phi_j) = (H\Phi_i.H\Phi_j)$.

[1] In this notation l refers to the first l eigenvectors of $\mathbf{C}$ used to build the Kernel PCA space.

[2] id here stands for the identity function: when performing linear PCA the kernel used is the inner scalar product in input space, hence the corresponding mapping function $\phi = id$.

2.2. Kernel for nonlinear PCA

Choosing a nonlinear kernel function $k(.,.)$ leads to performing nonlinear PCA. The exponential kernel has been a popular choice in the machine learning community and has proven to nicely extract nonlinear structures from data sets. Using SDFs for representing shapes, this kernel is given by

$$k_{\varphi_\sigma}(\Phi_i, \Phi_j) = e^{-\frac{\|\Phi_i - \Phi_j\|^2}{2\sigma^2}}, \quad (5)$$

where σ^2 is the variance parameter computed *a-priori* and $\|\Phi_i - \Phi_j\|^2$ is the squared L_2-distance between two SDFs Φ_i and Φ_j. If the shapes are represented by binary maps, the corresponding kernel is

$$k_{\varphi_\sigma}(H\Phi_i, H\Phi_j) = e^{-\frac{\|H\Phi_i - H\Phi_j\|^2}{2\sigma^2}}. \quad (6)$$

This exponential kernel is one among many possible choice of Mercer kernels: Other kernels could possibly be used to extract other specific features from the training set [8].

2.3. Shape Prior for GAC

To include prior knowledge on shape in the GAC framework, we propose to use the projection on the Kernel PCA space as a model and to minimize the following energy:

$$E^{\mathrm{F}}_{\mathrm{shape}}(\chi) := d^2_F[\varphi(\chi), P^l\varphi(\chi)] \quad (7)$$

A similar idea was proposed in [13, 8], for the purpose of pattern recognition. In (7), χ is a test shape represented using either a SDF ($\chi = \varphi$) or a binary map ($\chi = H\varphi$) and ϕ refers to either *id* (linear PCA) or ϕ_σ (Kernel PCA). Minimizing $E^{\mathrm{F}}_{\mathrm{shape}}$ amounts to driving the test shape χ towards the Kernel PCA space computed a priori from a training set of shapes using (2). In the GAC framework, the minimization of $E^{\mathrm{F}}_{\mathrm{shape}}(\chi)$, can be undertaken as follows:

$$\frac{d\phi}{dt} = -\nabla_\phi E^{\mathrm{F}}_{\mathrm{shape}} = -\nabla_\chi E^{\mathrm{F}}_{\mathrm{shape}} \cdot \frac{d\chi}{d\phi} \quad (8)$$

The gradient of $E^{\mathrm{F}}_{\mathrm{shape}}$ can be computed by applying calculus of variation and (3). For the kernel given in (6), the following result is obtained:

$$\nabla_\phi E^{\mathrm{F}}_{\mathrm{shape}} = -\frac{\Sigma_{i=1}^N g_i(\phi) k_{\varphi_\sigma}(H\phi, H\phi_i)\delta(\phi)[H\phi - H\phi_i]}{\sigma^2}$$

$$\text{with } [g_1(\phi), \ldots, g_N(\phi)] = -\frac{2}{N}\mathbf{1}^t + 2\tilde{\mathbf{k}}^t_{H\phi}\mathbf{M}\tilde{\mathbf{K}}\mathbf{M}\mathbf{H} - 4\tilde{\mathbf{k}}^t_{H\phi}\mathbf{M}\mathbf{H}$$

2.3.1 Linear PCA vs Kernel PCA—In this section we compare linear PCA with non-linear PCA for two different representations of shapes, i.e., SDF and binary map. Two training set of shapes were used: The first training set consists of various shapes of a man playing soccer and the second training set consists of various shapes of a shark (see Figure 1). These shapes were aligned using an appropriate registration scheme (see, [14]) to discard differences between them due to Euclidian transformations. The Kernel PCA space

corresponding to each of the kernels presented in Sections 2.1 and 2.2 were then built for the two training sets. Starting from an arbitrary shape, Figure 2(a), the contour was deformed by running equation (8) until convergence: We will refer to this operation as "morphing", in what follows.

Figure 2(b) shows the morphing results obtained by applying linear PCA on SDF. Figure 2(d) shows the morphing results obtained by applying linear PCA on binary maps. As can be noticed, results obtained for the SDF representation bear little resemblance with the elements of the training sets. Results obtained for binary maps are more faithful to the learnt shapes. Figure 2(c) and (e) present the morphing results obtained by applying nonlinear PCA on SDF and binary maps, respectively. In both cases, the final contour is very close to the training set and results are better than any of the results obtained with linear PCA.

Hence, Kernel PCA outperforms linear PCA as a means to introduce shape priors and binary maps seem to be an efficient shape representation. Besides, the learning process using Kernel PCA comes with no significant additional cost compared to linear PCA, thanks to the kernel formulation [8, 13]. Another advantage of using the exponential kernel, is that it enables to control the degree to which "mixing" is allowed between the learnt shapes, in the shape prior, larger σ's allowing more mixing. This is shown in Figure 3. The choice of σ typically depends on how much the shapes vary within the data set: If the variation is large, a smaller value for σ is usually preferable.

3. Intensity based segmentation

Different models [18, 1, 10], which incorporate geometric and/or photometric (color, texture, intensity) information, have been proposed to perform region based segmentation using level sets. In what follows, we present a novel intensity based segmentation framework aimed at separating an object from the background, in an image *I*. The main idea behind the proposed method is to build an "image shape model" (denoted by $G_{[I,\Phi]}$) by thresholding the image *I* based on the estimates of the intensity statistics of the object (and background), available at each step *t* of the contour evolution: $G_{[I,\Phi]}$ is interpreted as the most likely shape of the object of interest, based on the available information. The contour at time *t* is deformed towards this "image shape model" by minimizing the following energy:

$$E_{\text{image}} := \|H\Phi - G_{[I,\Phi]}\|^2 = \int_{\Omega} (H\Phi - G_{[I,\Phi]})^2 dxdy. \quad (9)$$

This energy functional amounts to measuring the distance between two binary maps, e.g.: $H\Phi$ and $G_{[I,\Phi]}$. This is quite valuable in the present context, where shapes are represented using binary maps as in earlier sections. Thus, when the shape energy term described before is combined with the following formulation for image segmentation, all the elements are expressed in terms of shapes. This is one of the unique contributions in this work. In what follows, we describe two particular cases of this general framework.

3.1. Object and background with different mean intensities

As in [1, 18], we assume that the image is composed of two regions having different intensity means: μ_o (respectively μ_b) is the mean intensity of the object (respectively of the background). Given an initial guess for the shape of the object and representing the contour as the zero level set of a SDF Φ, one can calculate the mean intensity inside (μ_1) and outside (μ_2) the curve as $\mu_1 = \frac{\int I(x,y) H\Phi dxdy}{\int H\Phi dxdy}$ and $\mu_2 = \frac{\int I(x,y)(1-H\Phi) dxdy}{\int (1-H\Phi) dxdy}$. The goal is to

deform this initial contour so that $\mu_1 = \mu_o$ and $\mu_2 = \mu_b$. To achieve this, the "image shape model" $G_{[I,\Phi]}$ is generated at each step *t*, in the following manner:

$$\text{if}\mu_1>\mu_2, G_{[I,\Phi]}(x,y)=\begin{cases} 1 & I(x,y)\geq\frac{\mu_1+\mu_2}{2}; \\ 0 & \text{else.}\end{cases} \quad \text{if}\mu_1\leq\mu_2, G_{[I,\Phi]}(x,y)=\begin{cases} 1 & I(x,y)\leq\frac{\mu_1+\mu_2}{2}; \\ 0 & \text{else.}\end{cases}$$

Notice that $G_{[I,\Phi]}$ is the image shape model (binary map) obtained from thresholding the image intensities so that values closer to μ_1 are classified as object (set to 1) and others are classified as background (set of 0). For numerical experiments, the function $G_{[I,\Phi]}$ is calculated as follows:

$$\text{if}\mu_1>\mu_2; G_{[I,\Phi,\varepsilon]}=\frac{1}{2}+\frac{1}{\pi}\arctan\left(\frac{I-\frac{\mu_1+\mu_2}{2}}{\varepsilon}\right) \text{ else} G_{[I,\Phi,\varepsilon]}=\frac{1}{2}-\frac{1}{\pi}\arctan\left(\frac{I-\frac{\mu_1+\mu_2}{2}}{\varepsilon}\right)$$

where ε, a parameter such that $G_{[I,\Phi,\varepsilon]} \to G_{[I,\Phi]}$ as $\varepsilon \to 0$.

3.2. Object and background with different variances

Following [10], we assume that the image is composed of two regions, with different variance in intensity. The mean intensities are computed as before, while the variances inside (σ_1) and outside (σ_2) the curve are computed as follows: $\sigma_1^2=\frac{\int(I-\mu_1)^2H\Phi dxdy}{\int H\Phi dxdy}$ and $\sigma_2^2=\frac{\int(I-\mu_2)^2(1-H\Phi)dxdy}{\int(1-H\Phi)dxdy}$. In this case, the image shape model $G_{[I,\Phi]}$ is obtained as follows:

$$\text{if}\sigma_1<\sigma_2, G_{[I,\Phi]}=\begin{cases} 1 & I_2\geq I(x,y)\geq I_1; \\ 0 & \text{else.}\end{cases} \quad \text{if}\sigma_1>\sigma_2, G_{[I,\Phi]}=\begin{cases} 1 & I(x,y)\geq I_1 \text{or} I(x,y)<I_2; \\ 0 & \text{else.}\end{cases}$$

where,

$$I_1=\frac{\sigma_2^2\mu_1-\sigma_1^2\mu_2-\alpha}{\sigma_2^2-\sigma_1^2}\text{and}I_2=\frac{\sigma_2^2\mu_1-\sigma_1^2\mu_2+\alpha}{\sigma_2^2-\sigma_1^2},\ \alpha=\sigma_2\sigma_1\sqrt{(\mu_1-\mu_2)^2+2(\sigma_1^2-\sigma_2^2)\log(\frac{\sigma_1}{\sigma_2})}$$

This thresholding ensures that pixels set to 1 in $G_{[I,\Phi]}$ correspond to pixels that are more likely to belong to the object of interest in the image, based on information available at step *t*. In the same way, pixels set to 0 in $G_{[I,\Phi]}$ correspond to pixels that are more likely to belong to the background. Figure 4 shows the different cases justifying the way thresholding is performed in equation (10).

In numerical applications, the binary map $G_{[I,\Phi]}$ in (10) is computed as follows (for ε small):

$$\text{if}\sigma_1<\sigma_2; G_{[I,\Phi,\varepsilon]}=\frac{1}{\pi}\arctan\left(\frac{I-I_1}{\varepsilon}\right)-\frac{1}{\pi}\arctan\left(\frac{I-I_2}{\varepsilon}\right) \text{ else} G_{[I,\Phi,\varepsilon]}=1-\frac{1}{\pi}\arctan\left(\frac{I-I_2}{\varepsilon}\right)+\frac{1}{\pi}\arctan\left(\frac{I-I_1}{\varepsilon}\right)$$

Figure 5, presents results obtained for each of the image shape models presented above.

4. Combining Shape Prior and Intensity information

In this part, we combine shape knowledge obtained by performing nonlinear PCA on binary maps with image information obtained by building an "image shape model", within the GAC framework. As presented above, $E^{\mathrm{F}}_{\mathrm{shape}}$ and E_{image} are squared distances between the shape of the current contour and a model. However, E_{image} is a squared distance in input space, whereas $E^{\mathrm{F}}_{\mathrm{shape}}$ is expressed in the feature space. Thus, equilibrium would be hard to reach between "forces" extracted from $E^{\mathrm{F}}_{\mathrm{shape}}$ and E_{image}[3]. This can be remedied by noticing that, for any SDFs φ_a and φ_b:

$$d^2_{F}(\phi_a,\phi_b)=2-2k_{\varphi_\sigma}(H\phi_a,H\phi_a)=2-2e^{-\frac{\|H\Phi_a-H\Phi_b\|^2}{2\sigma^2}}$$

By defining $E_{\mathrm{shape}}=-2\sigma^2\log(\frac{2-E^{\mathrm{F}}_{\mathrm{shape}}}{2})$, a new shape prior energy functional is obtained[4]. This energy E_{shape}, like E_{image}, is homogeneous to a square distance in input space. This consistent description of energies allows for efficient and intuitive equilibration between image cues and shape knowledge, through the following energy functional:

$$E(\Phi,I)=\beta_1 E_{\mathrm{shape}}(\Phi)+\beta_2 E_{\mathrm{image}}(\Phi,I) \quad (10)$$

4.1. Invariance to Similarity Transformations

Let $\mathbf{p} = [t_x, t_y, \theta, \rho] = [p_1, p_2, p_3, p_4]$ be a vector of parameters corresponding to a similarity transformation; t_x and t_y corresponding to translation according to x and y-axis, θ being the rotation angle and ρ the scale parameter. Let us denote by $\hat{I}(x, \hat{y})$ the image obtained by applying the transformation: $\hat{I}(x, \hat{y}) = I(\rho(x \cos\theta - y \sin\theta + t_x), \rho(x \sin\theta + y \cos\theta + t_y))$. As mentioned above, the elements of the training sets are aligned prior to the construction of the space of shapes. Supposing that the object of interest in I differs from the registered elements of the training set by a similarity of parameter $\mathbf{p}$, this transformation can be recovered by minimizing $E(\Phi, \hat{I})$ with respect to the p_i's. During evolution, the following gradient descent scheme can be performed for $i \in [1, 4]$:

$$\frac{dp_i}{dt}=-\nabla_{p_i}E(\Phi,\hat{I})=-\nabla_{p_i}E_{\mathrm{image}}(\Phi,\hat{I}).$$

5. Experiments

This section presents segmentation results obtained by introducing shape prior using Kernel PCA on binary maps and using our intensity based segmentation methodology: Equation (10) was run until convergence on diverse images.

---

[3] $\nabla_\Phi E^{\mathrm{F}}_{\mathrm{shape}}$ would, indeed, exhibit nonlinear behaviors due to the exponential terms figuring in its expression

[4] By applying the chain rule, one can verify that $\nabla_\varphi E_{\mathrm{shape}}$ and $\nabla_\phi E^{\mathrm{F}}_{\mathrm{shape}}$ have the same direction and similar influence on the evolution.

5.1. Toy Example: Shape Priors Involving Objects of Different Types

Kernel methods have been used to learn complex multimodal distributions in an unsupervised fashion (see [8], and the references therein). The goal of this section is to investigate the ability of the proposed framework to simultaneously learn and accurately detect objects of different shapes. To this end, we built a training set consisting of four words, “orange”, “yellow”, “square” and “circle” each written using twenty different fonts. The size of the fonts was chosen to lead to words of roughly the same length. The obtained words (binary maps, see Figure 1) were then registered according to their centroid. No further effort such as matching the letters of the different words was pursued. The method presented in Section (2) was used to build the corresponding space of shapes for the registered binary maps.

We tested our framework on images where a corrupted version of either of the four words “orange”, “yellow”, “square” or “circle” was present (Figure 6, 1^{st} row). Word recognition is a challenging task and addressing it using geometric active contours may not be a panacea. However, the ability of the level set representation to naturally handle topological changes was found to be useful for this purpose: During evolution, the contour split and merged a certain number of times to segment the disconnected letters of the words. In all the following experiments, β_1 and β_2 were fixed in (10) and the *same* initial contour was used.

Experiment 1: In this experiment, one of the words “square” belonging to the training set was corrupted: The letter “u” was almost completely erased. The shape thus obtained was filled with gaussian noise of mean $\mu_o = .5$ and variance $\sigma_o = .05$. The background was also filled with Gaussian noise of same mean $\mu_b = .5$ but of variance $\sigma_b = .2$. The result of applying our method is presented Figure 6(a). Despite the noise and the partial deletion, a very convincing segmentation is obtained. In particular, the *correct font* is detected and the letter “u” accurately reconstructed. In addition, the final curve is smooth even if no curvature term was used for regularization. Hence, using binary maps to represent shape priors can have valuable smoothing effects, even when dealing with noisy images.

Experiment 2: In this second experiment, one of the elements of the training set was used. A thick line (occlusion) was drawn on the word and a fair amount of gaussian noise was added to the resulting image. The result of applying our method is presented Figure 6(b). Despite the noise and the occlusion, a very convincing segmentation is obtained. In particular, the *correct font* is detected and the thick line completely removed. Once again, the final contour is smooth despite the fairly large amount of noise.

Experiment 3: Here, the word “yellow” was written using a *different* font from the ones used to build the training set. Additionally, a “linear shadowing” was used in the background (completely hiding the letter ”y”) and the letter ”w” was replaced by a grey square. The result of applying our framework is presented in Figure 6(c). The word “yellow” is correctly recognized and segmented. Also, the letters “y” and ”w”, were completely reconstructed.

Experiment 4: In this experiment, the word “orange” was *handwritten* in capital letters roughly matching the size of the letters of the words in the training set. The intensity of the letters was chosen to be rather close to some parts of the background. In addition, the word was blurred and smeared in a way that made its letters barely recognizable. This type of blurring effect is often observed in medical images due to patient motion. This image is particularly difficult to segment, even using shape prior, since the spacing between letters and the letters themselves are very irregular due to the combined effects of handwriting and blurring. Hence, mixing between classes (confusion between either of the 4 words) can be expected in the final result. In the final result obtained, the word “orange” is not only

recognized but satisfyingly recovered; in particular, a thick font was obtained to model the thick letters of the word (Figure 6(d)).

Hence, starting for each experiment from the same initial contour, our algorithm was able to accurately detect which word was present in the image. This highlights the ability of our method not only to gather image information throughout evolution but also to distinguish between objects of different classes (“orange”, “yellow”, “square” and “circle”). Comparing the final contours obtained in each experiments to the final “image shape model” $G_{[I,\varphi]}$ (last row of Figure 6), one can measure the effect of our shape prior model in constraining the contour evolution: The image information alone would lead to a shape that would bear very little resemblance with any of the four words learnt.

5.2. Real Images Example: Tracking of challenging sequences

To test the robustness of the framework, tracking was performed on two challenging sequences. A very simple tracking scheme was used: the same initial contour was used for each image in the sequence. This contour was initially positioned wherever the final contour was in the preceding image. The coefficients β_1 and β_2 were fixed throughout each sequence. Of course, many efficient tracking algorithms have already been proposed. However, convincing results were obtained here without considering the system dynamics, for instance. This highlights the efficiency of including prior knowledge on shape for the robust tracking of deformable objects.

5.2.1 Soccer Player Sequence—In this sequence (composed of 130 images), a man is jingling with a soccer ball. The challenge is to accurately capture the large deformations due to the movement of the person (e.g.: limbs undergo large changes in aspect), while sufficiently constraining the contour to discard clutter in the background. A training set of 22 silhouettes (Figure 1, first row) was used. The version of E_{image} involving the intensity means only was used to capture image information. Despite the small number of shapes used, successful tracking was obtained, correctly capturing the posture of the player.

5.2.2 Shark Video—In this sequence (composed of 70 images), a shark is evolving in a highly cluttered environment. Besides, the shark is oftentimes occluded by other fish and is poorly contrasted. To perform tracking, 15 shapes were extracted from the first half of the video (Figure 1, second row) and used to build shape prior. The version of E_{image} involving the variances was used to make up for the poor contrast of the shark in the images. Once again, despite the small training set, successful tracking performances were observed: The shark was correctly captured, while clutter and obstacles rejected.

6. Conclusion

In this work, we used Kernel PCA to introduce prior knowledge about shapes in the GAC framework. Better performance of Kernel PCA over linear PCA was demonstrated for two representations of shapes (binary maps and SDFs). We also developed a general approach to separate an object from the background using various image intensity statistics. In our algorithm, image information and shape knowledge were combined in a consistent fashion: both energies were expressed in terms of shapes. The proposed method not only allowed to simultaneously learn shapes of different objects but was also robust to noise, blurring, occlusion and clutter. In addition, even if the same parameters and same initial contour were used for each of the image of the sequences, successful tracking was obtained: This further highlights the robustness of the framework.

Acknowledgments

This research was supported in part by grants from NSF, AFOSR, ARO, MURI, MRI-HEL. This work is part of the National Alliance for Medical Image Computing (NAMIC), funded by the National Institutes of Health through the NIH Roadmap for Medical Research, Grant U54 EB005149. This work was also supported by a grant from NIH (NAC P41 RR-13218) through Brigham and Women's Hospital.

References

1. Chan T, Vese L. Active contours without edges. IEEE Trans. on Image Processing. 2001; 10(2): 266–277.
2. Cootes T, Taylor C, Cooper D. Active shape models-their training and application. Comput. Vis. Image Understanding. 1995; volume 61:38–59.
3. Cremers D, Kohlberger T, Schnoerr C. Diffusion snakes: introducing statistical shape knowledge into the mumford-shah functional. International journal of computer vision. volume 50
4. Cremers D, Kohlberger T, Schnoerr C. Shape statistics in kernel space for variational image segmentation. Pattern Recognition. 2003; volume 36:1292–1943.
5. Cremers D, Osher S, Soatto S. Kernel density estimation and intrinsic alignment for knowledge-driven segmentation: teaching level sets to walk. Pattern Recognition Proc. of DAGM 2004. 2004; volume 3157:36–44.
6. Kwok J, Tsang I. The pre-image problem in kernel methods. IEEE transactions on neural networks. volume 15:1517–1525.
7. Leventon, M.; Grimson, E.; Faugeras, O. Proc. CVPR. IEEE; 2000. Statistical shape influence in geodesic active contours; p. 1316-1324.
8. Mika S, Scholkopf B, Smola A. Kernel pca and de-noising in feature spaces. Advances in neural information processing systems. 1996; volume 11
9. Osher SJ, Sethian JA. Fronts propagation with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. Journal of Computational Physics. 1988; 79:12–49.
10. Rousson M, Deriche R. A variational framework for active and adaptative segmentation of vector valued images. IEEE Workshop on Motion and Video Computing. 2002
11. Rousson, M.; Paragios, N. Shape priors for level set representations; Proceedings of European Conference on Computer Vision; 2002. p. 78-92.
12. Sapiro, G., editor. Geometric Partial Differential Equations and Image Analysis. Cambridge University Press; 2001.
13. Scholkopf B, Mika S, Muller K. Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation. 1998; volume 10
14. Tsai A, Yezzi T, Wells W. A shape-based approach to the segmentation of medical imagery using level sets. IEEE Trans. on Medical Imaging. 2003; 22(2):137–153.
15. Wang, Y.; Staib, L. Boundary finding with correspondance using statistical shape models; IEEE Conf. Computer Vision and Pattern Recognition; 1998. p. 338-345.
16. Yezzi A, Kichenassamy S, Kumar A. A geometric snake model for segmentation of medical imagery. IEEE Trans. Medical Imag. 1997; volume 16:199–209.
17. Yezzi A, Soatto S. Deformotion: Deforming motion, shape average and the joint registration and approximation of structures in images. International Journal of Computer Vision. 2003; volume 53:153–167.
18. Yezzi A, Tsai A, Willsky A. A statistical approach to snakes for bimodal and trimodal imagery. Proc. Int. Conf. Computer Vision. 1999; volume 2:898–903.

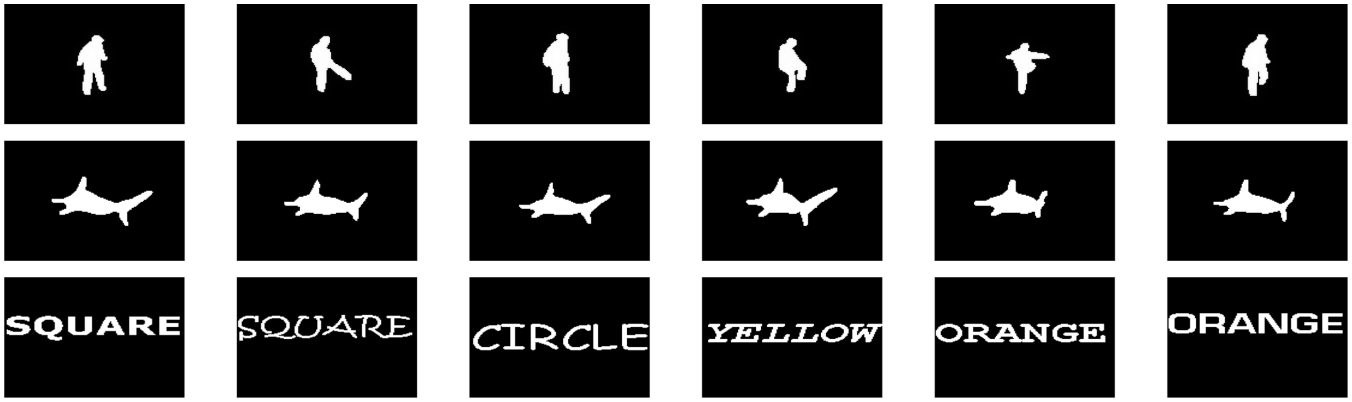

Figure 1.
Three training sets (Before alignment - Binary images are presented here). First row, “Soccer Player” silhouettes (6 of the 22 used). Second row, “Shark” silhouettes (6 of the 15 used). Third row, “4 words” (6 of the 80 learnt; 20 fonts per word).

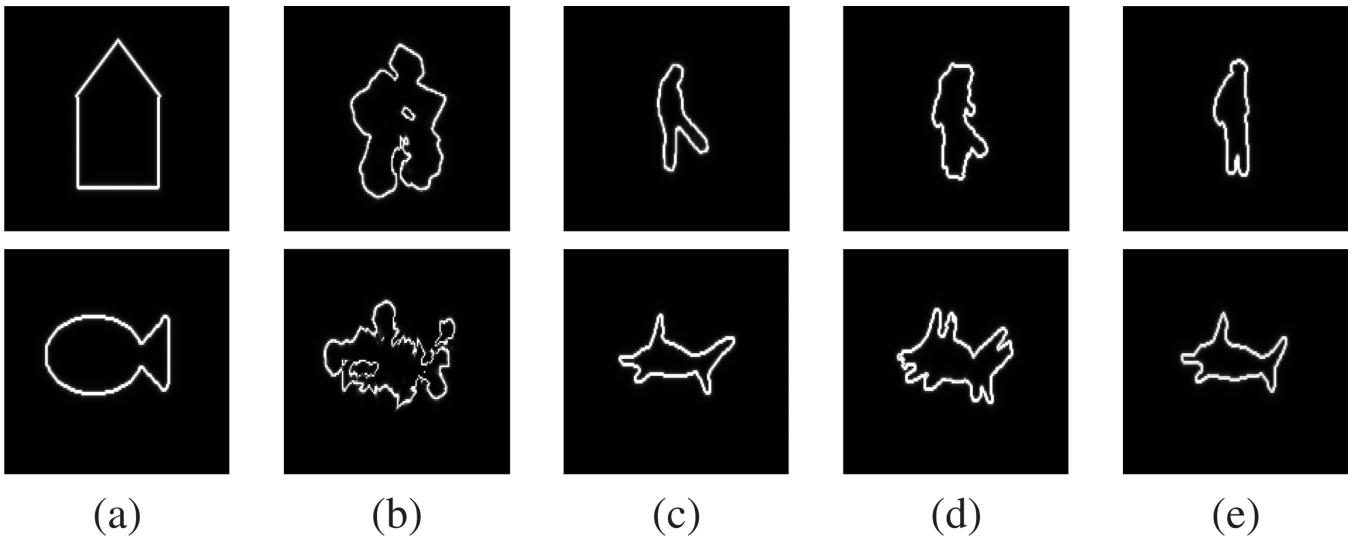

Figure 2.
Morphing results of an arbitrary shape, obtained using Linear PCA and Kernel PCA applied on both Signed Distance Functions and binary maps. First row: Results for the "Soccer Player" training set, Second row: Results for the "Shark" training set. (a): Initial shape, (b): PCA on SDF, (c): Kernel PCA on SDF (d): PCA on binary maps, (e): Kernel PCA on binary maps.

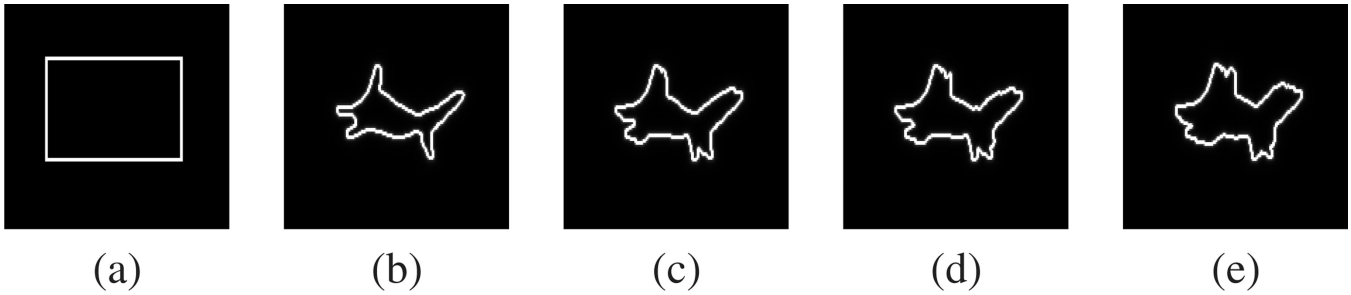

Figure 3.
Influence of σ for the Kernel PCA method (exponential kernel) applied on binary maps. Morphing results of an arbitrary shape are presented for the “Shark” training set. (a): Initial shape, (b): Morphing result for σ = 3, (c): σ = 7, (d): σ = 9, (e): σ = 15,.

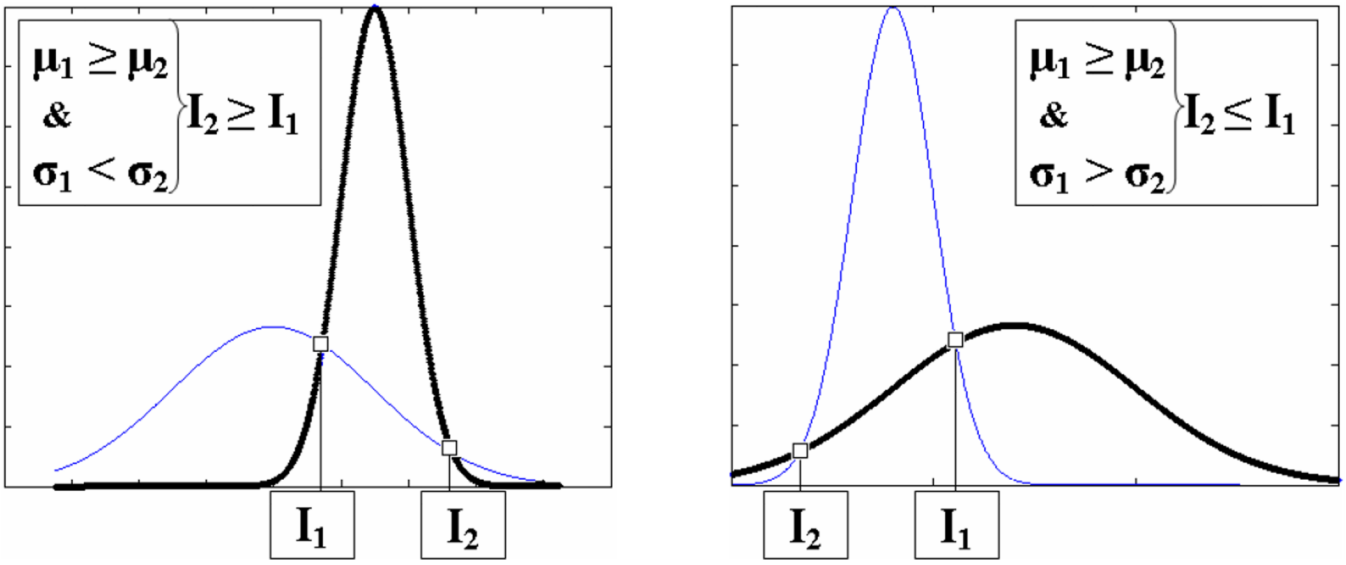

Figure 4.

Probability density functions. Thick line: $p(I \in \text{Object})$; Thin line: $p(I \in \text{background})$. It is straightforward to see that $p(I \in \text{Object}) > p(I \in \text{Object})$ for $I_1 \leq I \leq I_2$, when $\sigma_1 < \sigma_2$ and for $I \geq I_1$ and $I \leq I_2$, when $\sigma_1 > \sigma_2$.

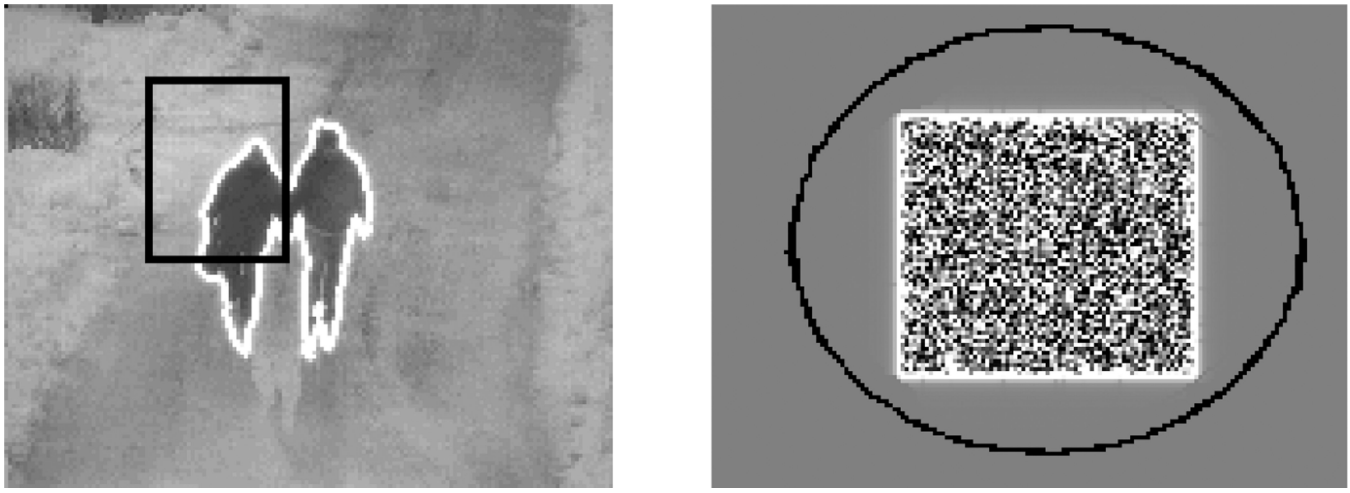

Figure 5.
Segmentation results obtained using E_{image}, equation (9). Initial contour in black, final contour in white. Left: 1^{st} moment only. Right: second order moments (Two regions of same mean intensity and different variances)

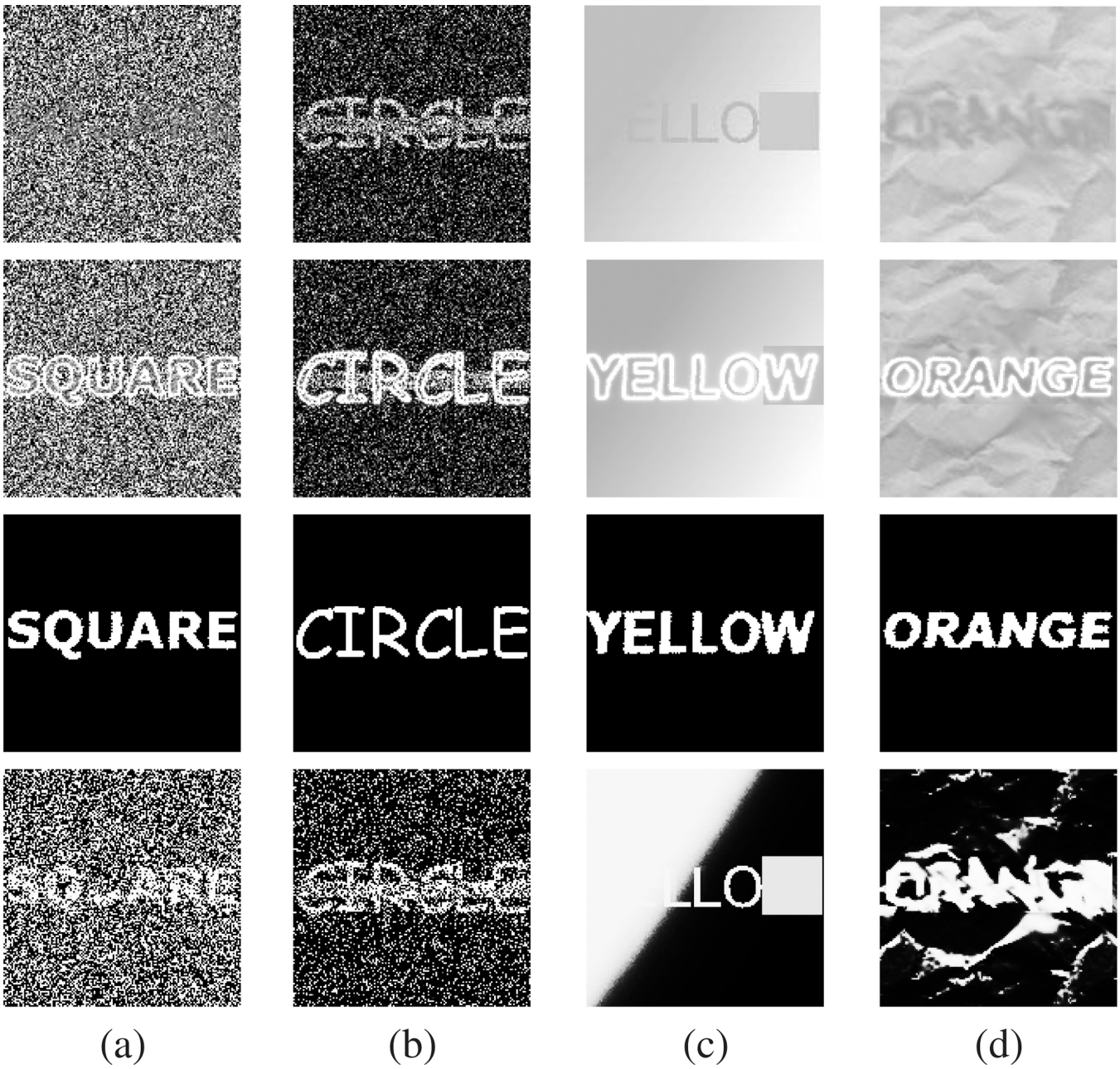

Figure 6.
Segmentation results for the "4 Words" training set. Shape Priors were built by applying the Kernel PCA method on binary maps as presented in Section 2. First row: Original images to segment, Second row: Segmentation results, Third row: shape underlined by the final contour ($H\varphi$), Fourth row: "Image shape model" ($G_{[I,\varphi]}$) obtained when computing E_{image} for the final contour.

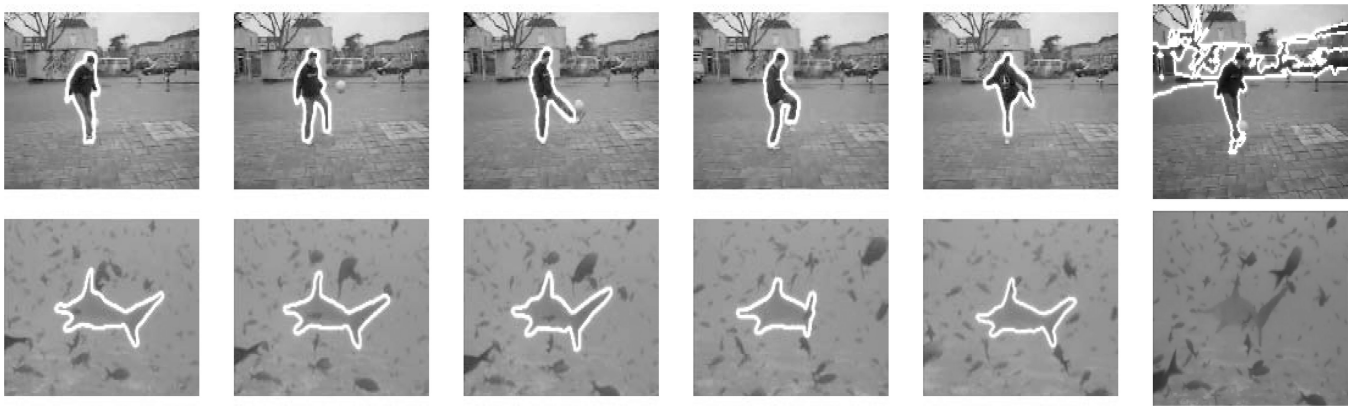

Figure 7.
Tracking results with the proposed method. First row: Soccer Player Sequence (Rightmost frame is the result obtained without shape prior; $\beta_2 = 0$ in (10)). Second row: "Shark" sequence (Rightmost frame is an original image from the sequence, reproduced here to assess the poor level of contrast).