STATISTICAL AND GEOMETRIC METHODS FOR SHAPE-DRIVEN SEGMENTATION AND TRACKING

A Thesis Proposal
Presented to
The Academic Faculty

by

Samuel Dambreville

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
March 2006

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

Computer Vision aims at developing techniques to extract information from images or other multidimensional data. Active contours are ubiquitous tools in Computer Vision applications. These contours evolve and deform according to energy functionals to perform certain tasks, such as segmenting images or tracking objects in video sequences, for instance.

In this report, we present two examples, where statistical methods and shape-driven techniques are combined to perform these typical computer vision tasks, within the geometric active contours (GAC) methodology:

1. In the first example, a novel tracking algorithm is presented. The algorithm can robustly track the position as well as the shape of the object of interest.

2. In the second example, we present a method for including prior knowledge on shapes in the evolution process of geometric active contours. The method can deal with very complex variations in shapes and is shown to constrain the shape of contours in a meaningful manner.

In our future work, we plan to focus on developing novel segmentation techniques and frameworks that combine information extracted from images as well as information learned a priori from examples. In particular, we plan to develop a novel region-based segmentation technique that uses the result of image thresholding to drive the evolution of the contour. Our goal is to develop robust techniques to segment complex images or track difficult video sequences.

Besides, we plan to study a novel type of contour evolution methodology that is based on the theory of interacting particle systems. This novel approach is envisioned to lead to a stochastic interpretation of geodesic active contours.

CHAPTER I

INTRODUCTION

Computer Vision aims at developing techniques to extract information from images or other multidimensional data. The applications of computer vision are multiple and include monitoring manufacturing processes, detecting events for surveillance purposes, or finding special structures in medical images, for instance. Some of the most challenging tasks performed by computer vision systems are segmentation: partitioning an image into meaningful regions; and tracking: following the evolution of an object in a video sequence.

Active contours have been proven to be quite valuable for performing the afore mentioned tasks. The main idea of active contours is to evolve a contour to decrease a "well chosen" energy. By "well chosen", we mean that the energy is defined so that its (local) minimum is reached when the contour delineates the borders of the object of interest in an image or when the contour adopts a particular shape, for instance.

In the geometric active contour (GAC) methodology, curves are represented implicitly with level sets. Level set representations were introduced by Osher and Sethian [37, 25] to model the motion of interfaces. The idea consists of representing the contour by the zero level set of a smooth continuous function (a common choice is to embed the contour in a signed distance function). The contour is evolved implicitly by evolving the embedding function to decrease an energy functional. Implicit representations present the advantage of avoiding to deal with complex re-sampling schemes of control points. Moreover, the contour represented implicitly can naturally undergo topological changes, such as splitting and merging.

Whether they are used to control the shape, position or evolution of the curve,

statistical methods are intimately related to the active contour methodology in general and the GAC framework, in particular. Also, the concept of shape has received a great deal of attention recently, and the use of shape-driven approaches has been shown to often drastically improve the performance of active contours. In what follows, we present two examples, where statistical methods and shape-driven techniques are combined to perform typical computer vision tasks, within the GAC methodology :

1. In Chapter 2, a novel tracking algorithm is presented, in which different statistical approaches are combined. The algorithm can track the position as well as the shape of the object of interest. Besides, in contrast with many other GAC-based tracking schemes, the method maintains the temporal coherency of the state of the object, while being computationally efficient.

2. In the second example, we present a method for including prior knowledge on shapes in the evolution process of geometric active contours. The proposed method is inspired from recent advancements in the Machine Learning community. The advantage of the method is that it can elegantly deal with very complex variations in shapes, while constraining the shape of contours in a meaningful manner.

The following two sections provide background information and references in relation to each of the two examples.

1.1 Literature review for the tracking example

The problem of tracking dynamic deformable objects has been a topic of substantial research in the field of controlled active vision (see [1, 38] and the references therein). In Chapter 2, we propose a scheme that combines the advantages of the unscented Kalman filter (UKF) and geometric active contours, for dynamic tracking.

In order to appreciate the proposed methodology, we briefly review some previous related work. Various finite dimensional parameterizations of continuous curves

have been proposed, perhaps most prominently the B-spline representation used for a "snake model", as in [38]. Isard and Blake (see [1] and references therein) applied the B-spline representation for contours of objects and proposed the Condensation algorithm in [12]. The authors in [20, 4] also use B-splines along with the unscented Kalman filter for rigid object tracking. Since these approaches only track the parameters of a finite dimensional group (e.g., Euclidean, affine transformations) they cannot handle severe deformations of the object of interest (see e.g., the fish example in Section 2.3.1). One possible solution proposed in [47], is to use deformable templates to model prior shapes, allowing for a few possible modes of deformation.

Another approach for representing contours is via the level set technique [36, 26]. To segment an object using level sets, an initial guess of the contour is deformed until it minimizes an image-based energy functional. Different energy functionals, which utilize different features of the image have been used in the literature (see e.g., [23, 3, 19, 2, 17, 45]). Some previous work on tracking with GACs is given in [24, 28, 46, 13]. In [46], the authors propose a definition for the global motion and shape deformation of a deformable object. Motion is parameterized by a finite dimensional group action (e.g., Euclidean or affine), while shape deformation is the total deformation of the object contour (infinite dimensional group) modulo the finite dimensional motion group. This is called the *deformotion* model. This approach, as well as most approaches using GAC cited above, rely only on the observed images for tracking and do not use any prior information on the dynamics of the group action or of the deformation. As a result, it is likely to fail if there is an outlier observation or if there is occlusion. To address this problem, the authors in [13] propose a generic local observer to incorporate prior information about the system dynamics in the deformotion framework.

Other approaches closely related to our work are given in [38, 29, 30]. In these papers, the authors use a Kalman filter in conjunction with active contours to track

nonrigid objects. The Kalman filter was used for predicting the possible movements of the object, while the active contours allowed for tracking the deformations of the object.

In [31], the authors use particle filters in combination with geometric active contours for tracking deformable objects. Compared with the approach in [31], our method has the advantage of only requiring a small number of deterministic sample points, and is therefore computationally very efficient.

In this work, we combine the advantages of the unscented Kalman filter (e.g., computational efficiency, second order accuracy) and geometric active contours to propose a novel method for tracking deformable objects. The proposed algorithm can maintain the temporal coherency of the state of the system and deal robustly with large deformations, partial occlusions and identity maintenance.

1.2 Literature review for the Learning example

Segmentation consists of extracting an object (or objects) from a given image, a 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 [19, 40, 45, 46] and the references therein). The active contour methodology has proven to be very effective 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.

A number of methods, which use a parameterized or an explicit representation for contours have been proposed [6, 44, 7] for active contour segmentation. In [8], the authors use the B-spline parametrization to build shape models in the kernel space [22]. The distribution of shapes in kernel space was assumed to be Gaussian and a

Mahalanobis distance was minimized during the segmentation process to control the shape of the contour.

Within the GAC framework, the authors in [19] obtain 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 as described in [40, 32], notably, where SDFs were used to learn shape variations.

However, when the object considered for learning undergoes complex or non-linear deformations, linear PCA can lead to unrealistic shape priors, by allowing linear combinations of the learned shapes that are unfaithful to the possible shapes of the object. Cremers *et al.* [9] successfully pioneered the use of kernel methods to address this issue within the GAC framework, using a Parzen estimator to model the distribution of shapes in kernel space.

In Chapter 3, we propose to use kernel PCA to introduce shape priors for GACs. Kernel PCA (KPCA) was proposed by Scholkopf *et al.* [22], and allows one to combine the precision of kernel methods with the reduction of dimension in training sets. This is the first time, to our knowledge, that kernel PCA is explicitly used to introduce shape priors in the GAC framework.

CHAPTER II

TRACKING DEFORMABLE OBJECTS WITH UNSCENTED KALMAN FILTER AND GEOMETRIC ACTIVE CONTOURS

In this chapter, we present a novel algorithm that combines several statistical and shape-based methods to perform tracking. This chapter is based on [11] and is organized as follows: In the next section, we discuss the unscented transformation, the unscented Kalman filter, The Chan and Vese model for region-based segmentation as well as the use of principal component analysis to learn shapes. In Section 2.2, we describe the proposed state space model and the algorithm in detail. Experimental results are given in Section 2.3.

2.1 Preliminaries

2.1.1 The Unscented Transformation

Julier and Uhlmann [15, 14] proposed a novel approach to generalizing the application of the Kalman filter to nonlinear systems. This approach is based on a statistical technique known as the *unscented transformation.* The unscented transformation leads to a more accurate filter than the traditional extended Kalman filter and avoids the costly computation of Jacobians [15, 43, 42].

Let $\mathbf{x}$ denote a n-dimensional random variable with mean $\hat{\mathbf{x}}$ and covariance $\mathbf{P}$. Let $\mathbf{g}$ be any arbitrary nonlinear function such that $\mathbf{y} = \mathbf{g}(\mathbf{x})$. To calculate the statistics of $\mathbf{y}$, a set of $2n+1$ weighted points or *sigma points* are first deterministically chosen as

$$\chi_0 = \hat{\mathbf{x}}, \quad \omega_0 = \kappa/(n+\kappa) \tag{1}$$

$$\chi_i = \hat{\mathbf{x}} + (\sqrt{(n+\kappa).\mathbf{P}})_i, \quad \chi_{i+n} = \hat{\mathbf{x}} - (\sqrt{(n+\kappa).\mathbf{P}})_i,$$

$$\omega_i = \omega_{i+n} = 1/\{2(n+\kappa)\}, \quad i = 1, ..., n$$

where κ is a scaling parameter, and $(\sqrt{(n+\kappa).\mathbf{P}})_i$ refers to the i^{th} column of the matrix square root of $(n+\kappa) \times \mathbf{P}$. The parameter ω_i is the weight associated with the i^{th} sigma point. Note that $\sum_{i=0}^{2n+1} \omega_i = 1$, and the obtained sigma points have same mean and covariance as $\mathbf{x}$. Each sigma point is then propagated through the nonlinear function $\gamma_i = g(\chi_i)$, with $i = 0, ..., 2n+1$. The estimated mean and covariance of $\mathbf{y}$ are computed as

$$\hat{\mathbf{y}} = \sum_{i=0}^{2n+1} \omega_i \gamma_i, \qquad \mathbf{P}_y = \sum_{i=0}^{2n+1} \omega_i (\gamma_i - \hat{\mathbf{y}})(\gamma_i - \hat{\mathbf{y}})^T$$

2.1.2 The Unscented Kalman Filter

In this section, we briefly present the specificity of the unscented Kalman filter (please refer to [16] for the basics concerning the Kalman filtering approach).

Let $\mathbf{x}$(k) denote the n-dimensional state at time k. The system evolves according to the equation

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{v}(k+1)) \tag{2}$$

where $f(.)$ is the state transition function, and $\mathbf{v}$(k+1) is a q-dimensional process noise vector. The m-dimensional measurement vector $\mathbf{y}$ is linked to the state of the system through the equation

$$\mathbf{y}(k+1) = h(\mathbf{x}(k+1), \mathbf{u}(k+1), \mathbf{w}(k+1)) \tag{3}$$

where $h(.)$ is the observation function in which $\mathbf{u}(k+1)$ is new information available at time $(t+1)$, and $\mathbf{w}$(k) is a r-dimensional measurement noise vector. Although more general assumptions about noise can be carried by the unscented Kalman filter, we assume in what follows that for all integers (i, j)

$$E[\mathbf{v}(k)] = E[\mathbf{w}(k)] = 0 \quad E[\mathbf{v}(i)\mathbf{w}^T(j)] = 0$$

$$E[\mathbf{v}(i)\mathbf{v}^T(j)] = \delta(i,j).\mathbf{Q} \quad E[\mathbf{w}(i)\mathbf{w}^T(j)] = \delta(i,j).\mathbf{R}$$

with $\mathbf{Q}$ and $\mathbf{R}$ being constant matrices of dimensions $(q \times q)$ and $(r \times r)$ respectively. $\mathbf{Q}$ is the process noise covariance matrix, $\mathbf{R}$ is the observation noise covariance matrix. The prediction and update steps of the Kalman filtering algorithm are carried as described in Section 2.1.2.1 and 2.1.2.2.

2.1.2.1 Prediction

Assume that the mean $\hat{\mathbf{x}}(k)$ and covariance matrix $\mathbf{P}(k|k)$ of the state $\mathbf{x}$(k) at time $t = k$ are known. The unscented transform is applied to the state vector $\mathbf{x}$(k) to obtain a set of $2n + 1$ sigma points $\chi_i(k|k)$ as presented above in Section (2.1.1).

The predicted state can be computed by applying the state transition function $f(\cdot)$ to each of the $\chi_i(k|k)$ to obtain a new set of sigma points $\chi_i(k+1|k)$, defined as

$$\chi_i(k+1|k) = f[\chi_i(k|k)] \quad i = 0, ..., 2n+1 \tag{4}$$

Considering the process noise as additive and independent of the state prediction, the predicted mean and covariance of the state, can be computed as

$$\hat{\mathbf{x}}(k+1|k) = \sum_{i=0}^{2n+1} \omega_i.\chi_i(k+1|k) \tag{5}$$

$$\mathbf{P}(k+1|k) = \sum_{i=0}^{2n+1} \omega_i\{\chi_i(k+1|k) - \hat{\mathbf{x}}(k+1|k)\} \cdot \{\chi_i(k+1|k) - \hat{\mathbf{x}}(k+1|k)\}^T + \mathbf{Q} \tag{6}$$

The predicted observation (measurement) is computed by applying the observation function h(.) to each of the $\chi_i(k+1|k)$ to obtain a new set of sigma points $\gamma_i(k+1|k)$ as

$$\gamma_i(k+1|k) = h[\chi_i(k+1|k), \mathbf{u}(k+1)] \qquad i = 0, ..., 2n+1 \tag{7}$$

Considering the measurement noise as additive and independent of the measurement prediction, the predicted mean and covariance of the measurement, can be computed as

$$\hat{\mathbf{y}}(k+1|k) = \sum_{i=0}^{2n+1} \omega_i.\gamma_i(k+1|k) \tag{8}$$

$$\mathbf{P}_{yy}(k+1|k) = \sum_{i=0}^{2n+1} \omega_i\{\gamma_i(k+1|k) - \hat{\mathbf{y}}(k+1|k)\} \cdot \{\gamma_i(k+1|k) - \hat{\mathbf{y}}(k+1|k)\}^T + \mathbf{R} \tag{9}$$

The predicted cross correlation is given by

$$\mathbf{P}_{xy}(k+1|k) = \sum_{i=0}^{2n+1} \omega_i\{\chi_i(k+1|k) - \hat{\mathbf{x}}(k+1|k)\} \cdot \{\gamma_i(k+1|k) - \hat{\mathbf{y}}(k+1|k)\}^T \tag{10}$$

2.1.2.2 Update

The update step is carried out in the same manner as within the traditional Kalman filter framework. The Kalman gain L, for time $t = k+1$ is

$$L(k+1) = \mathbf{P}_{xy}(k+1|k).\mathbf{P}_{yy}(k+1|k)^{-1}$$

The actual measurement $\mathbf{z}$ can be taken into account, leading to the estimate of the state statistics:

$$\mathbf{x}(k+1) = \hat{\mathbf{x}}(k+1|k) + L.(\mathbf{z} - \hat{\mathbf{y}}) \tag{11}$$

and

$$\mathbf{P}(k+1) = \mathbf{P}(k+1|k) - L.\mathbf{P}_{yy}(k+1|k).L^T \tag{12}$$

2.1.3 The Model of Chan and Vese

Active contours evolving according to edge-based and/or region-based flows are very commonly used for image segmentation (see [33] and the references therein). The level set representation is the most widely used tool for implementing such geometric curve evolution equations (see [25]).

Many methods [32, 39, 19], which incorporate geometric and/or photometric (color, texture, intensity) information have been shown to segment images in the presence of noise and clutter. In the present work, we have used the Mumford-Shah

functional [23] as modelled by Chan and Vese [3] to obtain the curve evolution equation. We seek to minimize the energy functional

$$E_{image} = \nu \int_{\Omega} |\nabla H(\Phi)| dx\ dy + \int_{\Omega} (I - c_1)^2 H(\Phi) dx\ dy + \int_{\Omega} (I - c_2)^2 (1 - H(\Phi))\ dx\ dy \tag{13}$$

where c_1 and c_2 are defined as

$$c_1 = \frac{\int I(x,y) H(\Phi) dx\ dy}{\int H(\Phi) dx\ dy}, \quad c_2 = \frac{\int I(x,y)(1 - H(\Phi)) dx\ dy}{\int (1 - H(\Phi)) dx\ dy}$$

and $H(\Phi)$ is the Heaviside function defined as

$$H(\Phi) = \begin{cases} 1 & if \quad \Phi \geq 0\ , \\ 0 & otherwise \end{cases}$$

$I(x, y)$ is the image and Φ is the level set function corresponding to the segmenting curve. The above energy functional E_{image} can be minimized using calculus of variations. The Euler-Lagrange equation for minimizing this functional can be implemented by the following gradient descent [3, 23]:

$$\frac{\partial \Phi}{\partial t} = \delta_\epsilon(\Phi) \left[\nu\ \text{div} \left(\frac{\nabla \Phi}{|\nabla \Phi|} \right) - (I - c_1)^2 + (I - c_2)^2 \right] \tag{14}$$

where $\delta_\epsilon(s) = \frac{\epsilon}{\pi(\epsilon^2 + s^2)}$.

2.1.4 Shape Statistics

In [19], the authors apply principal component analysis on a set of signed distance functions to obtain the major modes of shape variation.

Let Φ_i represent the signed distance function corresponding to the curve C_i. All the Φ_i's are aligned using a suitable method of registration [5]. The mean surface μ is computed by taking the mean of the signed distance functions:

$$\mu = \frac{1}{n} \sum \Phi_i$$

The variance in shape is computed by first subtracting the mean shape μ from each Φ_i to create a mean-offset map $\bar{\Phi}_i$. Then, each such map $\bar{\Phi}_i$ is placed as a column

vector in an $N^d \times n$-dimensional matrix M, where $\Phi_i \in \mathbf{R}^{N^d}$. The covariance matrix can then be computed as

$$C = \frac{1}{n} M M^T$$

Using Singular Value Decomposition (SVD), C can be decomposed as

$$U \Sigma U^T = \frac{1}{n} M M^T \tag{15}$$

where U is a matrix whose column vectors represent the set of orthogonal modes of shape variation, and Σ is a diagonal matrix of corresponding singular values.

Any novel shape Φ of the same class of object (*and registered with the learned shapes*) can be represented by an m-dimensional vector of coefficients, $\alpha = U_m^T(\Phi - \mu)$, where U_m is a matrix consisting of the first m columns of U. Given the coefficients α, an estimate $\tilde{\Phi}$ of the shape Φ can be obtained as

$$\tilde{\Phi} = U_m \alpha + \mu \tag{16}$$

2.2 The proposed Algorithm

2.2.1 The State Space Model

The authors in [46] separate the motion of an object into two distinct parts: a "global" rigid motion, and a "deformation" (any departure from rigidity). They also show that the overall motion of a moving and deforming object can be described by a set of non-unique rigid motion parameters and a deformation function. Accordingly, in this work, we assume that the global motion of an object is given by the translation of its centroid and any other deformation is captured by the curve evolution equation (14) described in the previous section.

We propose to combine the advantages of the unscented Kalman filter and geometric active contours in order to track both aspects of the motion of an object, position and deformation. A curve being an element of the infinite dimensional space $S^1 \mapsto \mathbb{R} * \mathbb{R}$, a finite dimensional approximation is needed to include the contour

into the state space and control it through the unscented Kalman filter. This finite dimensional approximation is obtained using the method presented in 2.1.4, which results in a projection in an orthogonal PCA base (of finite dimension).

In our framework, the state vector is thus composed of the coordinates of the centroid of the object (x_c, y_c) and the curve coordinates in the PCA base (m-dimensional coefficient vector α):

$$\mathbf{x}(k) = \begin{pmatrix} x_c \\ y_c \\ \alpha \end{pmatrix}(k) = \begin{pmatrix} X \\ \alpha \end{pmatrix}(k)$$

Consequently, the dimension of the state vector is $m + 2$. The observation space in this model is also a $m + 2$-dimensional vector given by

$$\mathbf{y}(k) = \begin{pmatrix} x_m \\ y_m \\ \beta \end{pmatrix}(k) = \begin{pmatrix} Y \\ \beta \end{pmatrix}(k)$$

where β is the m-dimensional coefficient vector representing the measured contour and (x_m, y_m) is the measured position. The predicted covariance matrices for the state $\mathbf{x}$ and the measurement $\mathbf{y}$ can be written as

$$\mathbf{P}(k+1|k) = \begin{pmatrix} P & 0 \\ 0 & \Sigma \end{pmatrix},\ \mathbf{P}_{yy}(k+1|k) = \begin{pmatrix} P_{yy} & 0 \\ 0 & \Sigma_{yy} \end{pmatrix}$$

where P and P_{yy} are 2×2 covariance matrices for the centroid co-ordinates, and Σ and Σ_{yy} are $m \times m$ diagonal matrices obtained as given in (15). Note that we have assumed that the centroid location is independent of the deformation in shape. The process noise matrix $\mathbf{Q}$ and measurement noise $\mathbf{R}$ are assumed to be constant throughout the state evolution process and are given by

$$\mathbf{Q} = \begin{pmatrix} Q_X & 0 \\ 0 & Q_\Sigma \end{pmatrix}, \mathbf{R} = \begin{pmatrix} R_X & 0 \\ 0 & R_\Sigma \end{pmatrix}$$

2.2.2 Measurement model

The measurement function at time k, $h(\mathbf{X}(k), \mathbf{I}(k))$, where $\mathbf{X}(k)$ is an $(m+2)$-dimensional seed point (corresponding to a curve centered at a certain position), and $\mathbf{I}$(k) is the image that becomes available at time $t = k$, can be described as

1. Build a cloud of l points $\mathbf{X}^{(i)}(k)$, for $i \in [|1..l|]$ around $\mathbf{X}(k)$ (these $(m+2)$-dimensional points correspond to curves centered at certain positions). One way to build this cloud of points is to define a fixed set of $(m+2)$ dimensional vectors $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_l\}$ and to take $\mathbf{X}^{(i)}(k) = \mathbf{X}(k) + \mathbf{s}_i$.

2. Run equation (14), for r iterations for each of the $\mathbf{X}^{(i)}(k)$. The number of iterations r can be chosen according to the expected dynamics of deformation of the object. This results in a local exploration of both the position and shape spaces in the neighborhood of $\mathbf{X}(k)$.

3. Select the curve with the minimum Chan-Vese energy (best fitting curve) as the measurement. The centroid of the selected curve is then taken as a measurement for the centroid of the system. The selected curve is registered to the learned shapes and projected in the PCA base, using equation (16). The projection of the selected curve provides the measurement of the coordinates of the shape.

As can be noticed, the measurement function $h(\mathbf{X}, \mathbf{I})$ is highly nonlinear because of the cumulative effects of the curves evolutions, the selection of the best-fitting curve and its projection into the PCA base. This non-linearity of the measurement function justifies the use of the unscented Kalman filter over more classical Kalman filtering approaches in our framework.

2.2.3 Algorithm

Based on the description above, the algorithm can be summarized as

1. Assume the state $\mathbf{x}(k)$ and covariance $\mathbf{P}(k)$ are known, at time $t = k$.

2. Obtain the $2(m+2)+1$ sigma points χ_i as presented in equation (1).

3. Obtain the state predictions using equation (4) to (6).

4. Obtain the measurements predictions using equation (7) to (9), taking $\mathbf{X}(k+1) = \chi_i(k+1|k)$ (for $i = 0, ..., 2(m+2)+1$) as seed points in the definition of the measurement function presented above, which also take $\mathbf{I}$(k+1) as an argument. Typically, using $l = 1$ (and $\mathbf{s}_1 = 0$), in the definition of h above, seemed to provide reasonable estimates for the predicted measurement statistics, for the data tested.

5. Obtain the actual measurement $\mathbf{z}$ by taking $\mathbf{X}(k+1) = \mathbf{x}(k)$ (for example) as a seed point and apply the measurement function on $\mathbf{I}$(k+1). A typical set $\mathbf{S}$ can be chosen to be $\mathbf{S} = \{ \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} \pm 3 \\ 0 \\ 0 \\ \vdots \end{pmatrix}, \begin{pmatrix} 0 \\ \pm 3 \\ 0 \\ \vdots \end{pmatrix} \}$

6. Complete unscented Kalman filtering process by computing equations (10) to (12).

2.3 Experiments

In this section, we describe some experiments performed to test the proposed tracking algorithm. Results of applying the proposed method on three image sequences are given below. Geometric active contour implementation was done using the narrow band method [37] and the model of Chan and Vese [3]. In these experiments, the state transition function f in equation (2) was chosen to be identity. For each sequence, a PCA base of the object to track was computed off-line as described before. A small training set of s images was extracted from each sequence and used to compute

m components representing the possible modes of variation of the tracked object (eigenvector matrix U_m).

2.3.1 Fish Sequence

The Fish video demonstrates the tracking ability of the proposed method under large deformations and partial occlusions. This type of deformations are difficult to track using the standard Condensation filter [1]. The number of images in the training set and of principal directions was 8, i.e., $s = m = 8$. The total number of images in the sequence was 356. The tracking was accurate throughout the whole sequence, despite the relatively small number of directions of variation allowed for the shape of the fish (Figure 1).

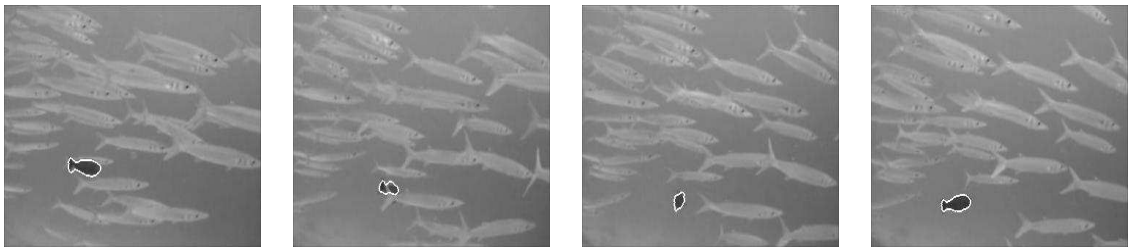
2.3.2 Car Sequence

In this sequence, the car is partially occluded as it passes behind a lamp post. The Chan-Vese model applied to tracking this sequence fails at the occlusion (Figure 2). Including the curve in the state space allows only a few directions of variation in the shape of the car. Hence, the overall shape of the curve does not vary too much from one frame to the next (Figure 3). The results shown on this figure are for $s = m = 6$. Note that even though the vector α provides a shape prior with only 6 possible directions of variation, the proposed model can indeed track large deformations and overcome partial occlusions.

2.3.3 Walking Couple Sequence

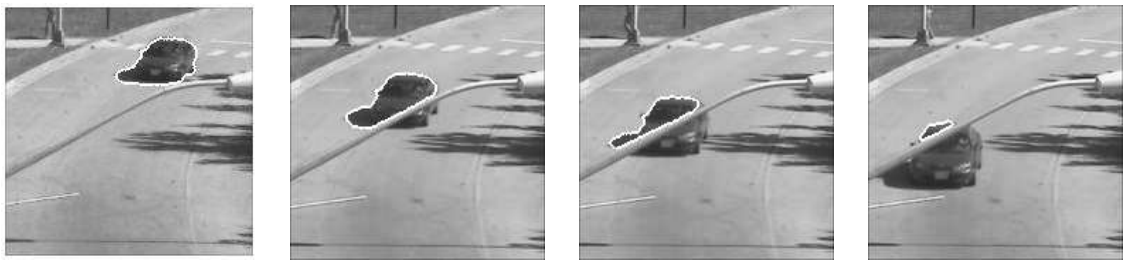
In the classical Walking Couple video, the difficulty resides in maintaining the identity of each person during tracking. Throughout this sequence, the two persons often touch each other, and the contour usually leaks from one person to the other and ends up encompassing the two people (Figure 4). The identity of each person can be maintained using the proposed method, since leaks are rejected by projecting the

contour onto the PCA base (this operation constrains the curve to adopt a shape resembling a walking person). For this sequence, the PCA base was computed from shapes of the person on the right; s and m were 8 and 6, respectively. The person on the left was accurately tracked throughout the sequence, highlighting the robustness of the PCA representation (Figure 5).



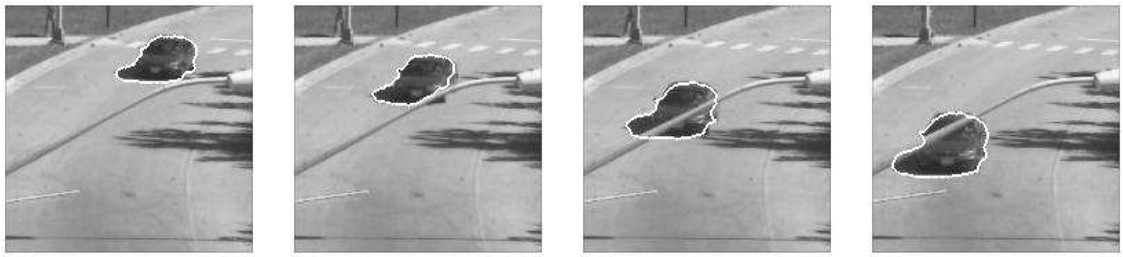
(a) Frame 110 (b) Frame 175 (c) Frame 195 (d) Frame 219

Figure 1: Fish Sequence: Tracking with the proposed method.



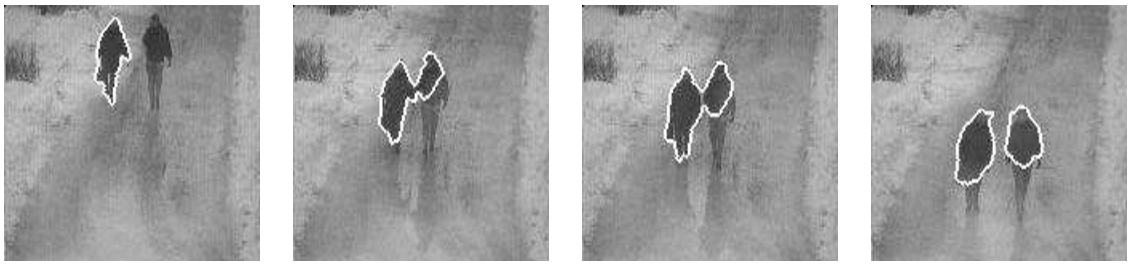
(a) Frame 20 (b) Frame 40 (c) Frame 50 (c) Frame 67

Figure 2: Car Sequence: Tracking with Chan-Vese model alone. the obstacle cannot be handled.



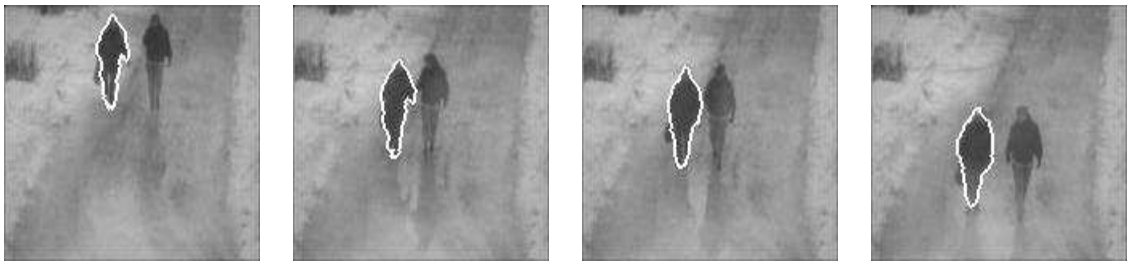
(a) Frame 20 (b) Frame 34 (c) Frame 50 (d) Frame 65

Figure 3: Car Sequence: Tracking with the proposed method.



(a) Frame 3 (b) Frame 11 (c) Frame 13 (d) Frame 20

Figure 4: Walking Couple Sequence: Tracking with Chan-Vese model. Identity cannot be maintained.



(a) Frame 3 (b) Frame 11 (c) Frame 13 (d) Frame 20

Figure 5: Walking Couple Sequence: Tracking with the proposed method.

CHAPTER III

NONLINEAR SHAPE PRIOR FROM KERNEL SPACE FOR GEOMETRIC ACTIVE CONTOURS

In this chapter, we present a novel shape learning framework for geometric active contours using kernel principal components analysis. This chapter is based on [10] and is organized as follows: In the next section, we propose a consistent method to introduce shape priors within the GAC framework, using linear and kernel PCA. In Section 3.2, we compare the performances of linear and kernel PCA and demonstrate the superiority of kernel PCA on various challenging examples.

3.1 Linear and Kernel PCA for Shape Priors

In this section, we recall a general formulation allowing one to perform linear PCA as well as kernel PCA on any data set [34, 18]. Then, we present specific kernels allowing to perform linear or non-linear principal component analysis on training sets of shapes. Finally, we propose an energy functional allowing to introduce shape priors obtained from either linear or kernel PCA, within the GAC methodology.

3.1.1 Kernel PCA

Kernel PCA can be considered to be a generalization of linear principal component analysis. This technique was introduced by Scholkopf [22], and was proven to be a powerful method to extract nonlinear structures from a data set. The idea behind KPCA 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 squares sense.

The nonlinear map $\varphi : \mathcal{I} \mapsto F$ typically does not need to be known, through the use of Mercer kernels. A *Mercer kernel* is a function $k(\cdot, \cdot)$ such that for all data points χ_i, the kernel matrix $\mathbf{K}(i,j) = k(\chi_i, \chi_j)$ is symmetric positive definite [22]. According to Mercer's Theorem (see [21]), computing $k(.,.)$ as a function of $\mathcal{I} \times \mathcal{I}$, amounts to computing the inner scalar product in F: $k(\chi_a, \chi_b) = \big(\varphi(\chi_a) \cdot \varphi(\chi_b)\big)$, with $(\chi_a, \chi_b) \in \mathcal{I} \times \mathcal{I}$. This inner scalar product in F defines a distance d_F, such as

$$d_F^2(\varphi(\chi_a), \varphi(\chi_b)) = \parallel \varphi(\chi_a) - \varphi(\chi_b) \parallel^2 = k(\chi_a, \chi_a) - 2k(\chi_a, \chi_b) + k(\chi_b, \chi_b)$$

We now briefly describe the KPCA method, please refer to [22] for further details. Let $\tau = \{\chi_1, \chi_2, ..., \chi_N\}$ be a set of training data. The centered kernel matrix $\tilde{\mathbf{K}}$ corresponding to τ, is defined as

$$\tilde{\mathbf{K}} = \big((\varphi(\chi_i) - \bar{\varphi}) \cdot (\varphi(\chi_j) - \bar{\varphi})\big) = (\tilde{\varphi}(\chi_i) \cdot \tilde{\varphi}(\chi_j)) = \tilde{k}(\chi_i, \chi_j), \ for\ i \in [|1, N|] \quad (17)$$

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

$$\tilde{\mathbf{K}} = \mathbf{U}\mathbf{S}\mathbf{U}^t \quad (18)$$

where $\mathbf{S} = \text{diag}(\gamma_1, ..., \gamma_N)$ is a diagonal matrix containing the eigenvalues of $\tilde{\mathbf{K}}$. $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_N]$ is an orthonormal matrix. The column-vectors $\mathbf{u}_i = [u_{i1}, ..., u_{iN}]^t$ are the eigenvectors corresponding to the eigenvalues γ_i's. Besides, it can easily be shown that $\tilde{\mathbf{K}} = \mathbf{H}\mathbf{K}\mathbf{H}$, where $\mathbf{H} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^t$. The vector $\mathbf{1} = [1, ..., 1]^t$ is of dimension $N \times 1$.

Let $\mathbf{C}$ denote the covariance matrix of the elements of the training set mapped by $\tilde{\varphi}$. Within the KPCA methodology, the covariance matrix $\mathbf{C}$, which is possibly of very high dimension, does not need to be computed explicitly. Only $\tilde{\mathbf{K}}$ needs to be

known to extract features from the training set, since the eigenvectors of $\mathbf{C}$ are simple functions of the eigenvectors of $\tilde{\mathbf{K}}$ (see [34]). 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* or *KPCA 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 KPCA space will be denoted by $P^l\varphi(\chi)$ [1]. The projection $P^l\varphi(\chi)$ can be obtained as described in [22]. The squared distance d_F^2 between a test point χ mapped by φ and its projection on the KPCA space is given by

$$\begin{aligned} d_F^2[\varphi(\chi), P^l\varphi(\chi)] &= \| \varphi(\chi) - P^l\varphi(\chi) \|^2 \\ &= k(\chi,\chi) - 2\big(\varphi(\chi)\cdot P^l\varphi(\chi)\big) + \big(P^l\varphi(\chi)\cdot P^l\varphi(\chi)\big). \end{aligned} \tag{19}$$

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

$$d_F^2[\varphi(\chi), P^l\varphi(\chi)] = k(\chi,\chi) + \frac{1}{N^2}\mathbf{1}^t\mathbf{K}\mathbf{1} - \frac{2}{N}\mathbf{1}^t\mathbf{k}_\chi + \tilde{\mathbf{k}}_\chi^t\mathbf{M}\tilde{\mathbf{K}}\mathbf{M}\tilde{\mathbf{k}}_\chi - 2\tilde{\mathbf{k}}_\chi^t\mathbf{M}\tilde{\mathbf{k}}_\chi \tag{20}$$

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

Figure 6 recapitulates the kernel PCA methodology as well as the projection operation on the space learned from the data.

3.1.2 Kernels for linear and non-linear PCA

3.1.2.1 Linear PCA

In [19], the authors present 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. For all SDFs ϕ_i and $\phi_j : \mathbf{R}^2 \mapsto \mathbf{R}$, linear PCA can be performed

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

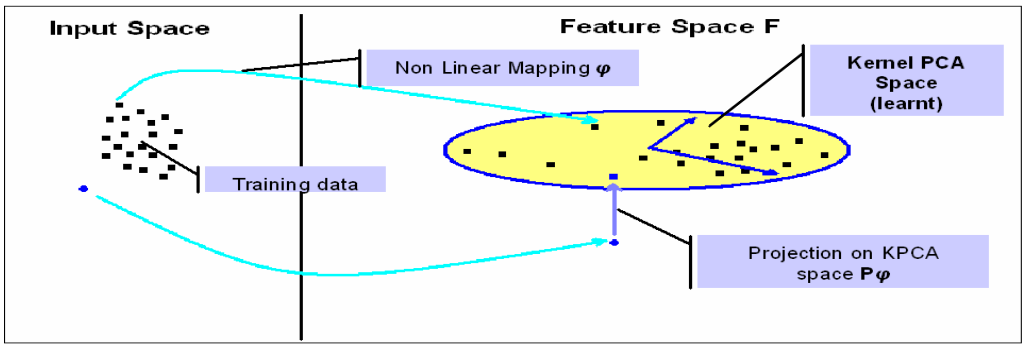


Figure 6: Kernel PCA methodology. A training set is mapped from Input space $\mathcal{I}$ to Feature space F, via a non-linear function φ. PCA is performed in F to determine the principal directions defining the kernel PCA space (learned space): Oval area. Any element of $\mathcal{I}$ can then be mapped to F and projected on the kernel PCA space via $P^l\varphi$.

using the kernel

$$k_{id}(\phi_i, \phi_j) = (\phi_i.\phi_j) = \int\int \phi_i(u,v)\phi_j(u,v)du.dv \tag{21}$$

The subscript id stands for the identity function. Indeed, when performing linear PCA the kernel used is the inner scalar product in the input space, hence the corresponding mapping function $\varphi = id$.

A different representation for shapes is to use binary maps, i.e., to set pixels located inside the shape to 1, and to set pixels located outside to 0 (see Figure 7). One can change the shape representation from SDFs to binary maps using the Heaviside function $H(\phi) = H\phi = \begin{cases} 1 & \text{if} \quad \phi \geq 0 \ , \\ 0 & \text{else} \ . \end{cases}$. Note that, in this case, the kernel allowing one to perform linear PCA is given by

$$k_{id}(H\phi_i, H\phi_j) = (H\phi_i.H\phi_j) \tag{22}$$

In numerical applications, a smooth version $H_\epsilon\phi$ of $H\phi$ can be obtained by taking $H_\epsilon\phi = \frac{1}{2} + \frac{1}{\pi}\arctan(\frac{\phi}{\epsilon})$, for ϵ small.

3.1.2.2 Nonlinear PCA

Choosing a nonlinear kernel function $k(\cdot,\cdot)$ is the basis of 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 (see e.g., [18]).

Using SDFs for representing shapes, this kernel is given by

$$k_{\varphi_\sigma}(\phi_i,\phi_j) = e^{-\frac{\|\phi_i-\phi_j\|^2}{2\sigma^2}} \tag{23}$$

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. The subscript φ_σ stands for the non-linear mapping corresponding to the exponential kernel. This mapping also depends on the choice of σ. If the shapes are represented by binary maps, the corresponding kernel is

$$k_{\varphi_\sigma}(H\phi_i,H\phi_j) = e^{-\frac{\|H\phi_i-H\phi_j\|^2}{2\sigma^2}} \tag{24}$$

This exponential kernel is one among many possible choice of Mercer kernels. Other kernels may be used to extract other specific features from the training set (see [22]).

3.1.3 Shape Prior for GAC

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

$$E^{\mathrm{F}}_{\mathrm{shape}}(\chi) := d_F^2[\varphi(\chi), P^l\varphi(\chi)] \tag{25}$$

A similar idea was presented in [35], for the purpose of pattern recognition and denoising. The superscript F in $E^{\mathrm{F}}_{\mathrm{shape}}$ denotes the fact that the shape knowledge is expressed as a distance in feature space. In (25), χ is a test shape represented using either a SDF ($\chi=\phi$) or a binary map ($\chi=H\phi$) and φ refers to either id (linear PCA) or φ_σ (kernel PCA). Minimizing $E^{\mathrm{F}}_{\mathrm{shape}}$ as in Equation (31) amounts to driving the test shape χ toward the kernel PCA space computed a priori from a training set

of shapes using (18). This contour evolution involving only the minimization of $E^{\mathrm{F}}_{\mathrm{shape}}$ (no image information) will be referred to as "warping", in what follows.

A number of researchers have proposed to minimize the distance between the current shape and the mean shape obtained from a training set. The assumption is, indeed, often made that the underlying distribution of familiar shapes, in either the input or feature space, is Gaussian [19, 7, 8]. Following this assumption, driving the curve toward the mean shape is a sensitive choice. Here, however, we deliberately chose to use the projection of the (mapped) current SDF to drive the evolution because we would like to deal with objects of different geometry in the training set (see e.g., Figure 7, third line; a training set of 4 words was used for the experiments). When dealing with objects of very different shapes, the underlying distribution can be quite non-Gaussian (e.g., multi-modal).

Thus, the average shape would not be meaningful in this case, since it would amount to mixing shapes belonging to different clusters. As a consequence, driving the (mapped) current shape toward its projection on the kernel PCA space appears to be a more sensible choice for our purpose. In addition, choosing the projection as a model of shape allows for comparing the given learning methods without image information, i.e., by warping the same initial contour for each method and comparing the final results in terms of their resemblance to the elements of the training set. The final shape obtained can be interpreted as *the most probable shape of the initial contour given what is known from the data* (see the warping experiments in Section 3.2.2). If the mean shape is chosen as a shape model, warping would result in the initial contour converging to this mean shape, and no comparison would be possible among learning methods. Thus, in this latest case, image information must be included, and it is difficult to conclude whether differences between segmentation performances are the result of differences in the accuracy of the learning methods or simply to poor balancing between image and shape information.

The gradient of $E^{\mathrm{F}}_{\mathrm{shape}}$ can be computed by applying the calculus of variations on (25), using the expression of $d^2_F(\varphi(\xi), P^l\varphi(\xi))$ in equation (20). In the GAC framework, the minimization of $E^{\mathrm{F}}_{\mathrm{shape}}(\chi)$, can then be performed as

$$\frac{d\phi}{dt} = -\nabla_\phi E^{\mathrm{F}}_{\mathrm{shape}} = -\nabla_\chi E^{\mathrm{F}}_{\mathrm{shape}} \times \frac{d\chi}{d\phi} \tag{26}$$

The minimization of E_{shape} for any arbitrary contour (no image information) results in the deformation of the contour toward a familiar shape (as presented in Section 3.2.2).

For the exponential kernel involving SDFs and given in (23), the following result is obtained:

$$\nabla_\phi E^{\mathrm{F}}_{\mathrm{shape}} = -\frac{\Sigma_{i=1}^N g_i(\phi) k_{\varphi_\sigma}(\phi, \phi_i)[\phi - \phi_i]}{\sigma^2} \tag{27}$$

with $[g_1(\phi), ..., g_N(\phi)] = -\frac{2}{N}\mathbf{1}^t + 2\tilde{\mathbf{k}}_\phi^t \mathbf{M}\tilde{\mathbf{K}}\mathbf{M}\mathbf{H} - 4\tilde{\mathbf{k}}_\phi^t \mathbf{M}\mathbf{H}$, and $\tilde{\mathbf{k}}_\phi$, $\mathbf{M}$ and $\tilde{\mathbf{K}}$ computed for k_{φ_σ}.

For the exponential kernel involving binary maps and given in (24), one derives that

$$\nabla_\phi E^{\mathrm{F}}_{\mathrm{shape}} = -\frac{\Sigma_{i=1}^N g_i(\phi) k_{\varphi_\sigma}(H\phi, H\phi_i)\delta(\phi)[H\phi - H\phi_i]}{\sigma^2} \tag{28}$$

where $\tilde{\mathbf{k}}_\phi$, $\mathbf{M}$ and $\tilde{\mathbf{K}}$ are computed for the kernel $k_{\varphi_\sigma}(H., H.)$.

For the kernel given in (21), corresponding to linear PCA on SDFs, the following result is obtained:

$$\nabla_\phi E^{\mathrm{linear}}_{\mathrm{shape}} = 2\phi + \Sigma_{i=1}^N g_i(\phi).\phi_i \tag{29}$$

where $\tilde{\mathbf{k}}_\phi$, $\mathbf{M}$ and $\tilde{\mathbf{K}}$ are computed for the kernel k_{id}.

Finally, for the kernel given in (22), corresponding to linear PCA on binary maps, one finds that

$$\nabla_\phi E^{\mathrm{linear}}_{\mathrm{shape}} = 2\delta(\phi) + \Sigma_{i=1}^N g_i(H\phi).H\phi_i\delta(\phi) \tag{30}$$

where $\tilde{\mathbf{k}}_\phi$, $\mathbf{M}$ and $\tilde{\mathbf{K}}$ are computed for the kernel $k_{id}(H., H.)$.

3.2 Experiments

3.2.1 Training sets and learning

In the experiments presented below, three training sets of shapes were used. Shapes were represented alternatively by signed distance functions and (smoothed) binary maps. The first training set of shapes consists of 22 shapes of a man playing soccer. The second training set is composed of 28 shapes of a shark. These shapes were aligned using an appropriate registration scheme [40] to remove differences between them resulting from translation, rotation, and scale. In order to test for the ability of the proposed framework to learn and deal with multi-modal distributions, a third training set was built. This training set consists of four words, *orange*, *yellow*, *square* and *circle* each written using 20 different fonts, leading to a training set of 80 shapes total. The size of the fonts was chosen to lead to words of roughly the same length. The obtained words were then registered according to their centroid. No further effort, such as matching the letters of the different words, was pursued. The binary maps corresponding to the diverse training shapes are presented in Figure 7. The shapes depicted are from the original training sets, before alignment. The first row in the figure presents a few elements of the Soccer Player Training Set, and the second row presents a few shapes from the Shark Training Set. The third row of Figure 7 presents a few of the words used to build the 4-Words Training Set. Shape learning was performed on each training set, as presented in Section 3.1. The familiar spaces of shapes (kernel PCA spaces) were built for each of the kernels presented in Equations (21) to (24), whether linear PCA or KPCA was performed on binary maps or SDFs.

3.2.2 Warping results: Linear PCA vs KPCA for shape priors

In this section, we compare the performances of kernel PCA to linear PCA, as methods for introducing shape priors in the GAC framework. The fact that energy functional E_{shape} is consistently defined for both shape learning methods in equation (25),

ensures that performances obtained from applying linear PCA or KPCA can be accurately compared. No image information was used in these experiments, instead the contour evolution was carried to minimize E_{shape} only (warping). This further guarantees a meaningful comparison among methods, since performances do not stem from the balancing factor between image information and shape knowledge. In what follows, the same initial shapes were warped and the final contours obtained were compared, for both methods, in terms of their resemblance to the elements of the training sets utilized. Equation (31) was run until convergence, using the expression of the gradients presented in (29) and (30) for linear PCA, and (27) and (28) for KPCA.

3.2.2.1 Warping an arbitrary initial shape

Figure 8 presents the warping results obtained for an arbitrary initial shape representing a cross. The first row of the figure shows the results obtained using kernel PCA (with SDFs as a representation of shape). The second row of the figure presents the results obtained using linear PCA (with shapes represented as SDFs). As can be noticed, results obtained with linear PCA bear little resemblance with the elements of the training sets. By contrast, final contours obtained employing KPCA are more faithful to the learned shapes.

Figure 9(b) shows the warping results obtained by applying linear PCA on SDF. Figure 9(d) shows the warping results obtained by applying linear PCA on binary maps. Note that the 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 learned shapes. Figure 9(c) and (e) present the warping results obtained by applying kernel 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.

3.2.2.2 Robustness to misalignment

In these experiments, we wanted to test the robustness of each framework to misalignments relative to the registered shapes of a training set. This robustness to misalignment is an interesting property to evaluate since, to perform segmentation, transformations (e.g., translations) between the segmenting contour and the registered training shapes need to be considered. Hence, whether a probabilistic method [19] or a gradient descent scheme [40] is used to compensate for these transformations, more or less important misalignments between the contour and the registered training shapes may occur during the evolution process. This can impair the ability of the shape energy to properly constrain the shape of the segmenting contour.

To test the robustness to misalignment, two elements of the training set were translated as compared to the registered training shapes and were used as initial contours for warping. For the first initial contour used, a small translation of about 5 pixels in the x and y-directions was performed. The second initial contour was translated of about 15 pixels in the x and y-directions. Figure 10 and Figure 11 present the warping results obtained for the slightly and more greatly misaligned initial shape, respectively. The first line of each of these figures shows the results obtained using kernel PCA, the second line presents the results obtained using linear PCA. In both cases, SDFs were chosen as representation of shapes. As can be noticed again, results obtained with linear PCA bear little resemblance with the elements of the training sets. By contrast, final contours obtained using kernel PCA are more faithful to the learned shapes. Besides, whether the initial contour is slightly or heavily misaligned, the contour evolution results in a translation of the contour when kernel PCA is used. It is interesting to note that the shape of the final contour obtained using KPCA is very much like the shape of the initial contour, for both experiments. Similar satisfying results were obtained with kernel PCA using binary maps as a representation of shapes. Hence, KPCA appears to offer a higher level of robustness than linear PCA

not only to slight but also to rather important misalignment of the initial contour (at least in the experiments we performed).

3.2.2.3 Multi-modal learning

Kernel methods have been used to learn complex multi-modal distributions in an unsupervised fashion [22]. The goal of this section is to investigate the ability of kernel PCA to simultaneously learn objects of different shapes and to constrain the contour evolution in a meaningful fashion. Besides, we want to contrast performances obtained with KPCA to performances obtained with linear PCA. The 4-Words Training Set was used for these experiments. Diverse contours, which shapes bore some degree of resemblance to any of the four words (*orange*, *yellow*, *square* or *circle*), were then used as initial contours for warping.

Each line in Figure 12 presents the warping results obtained for the diverse initial contours used, for both linear and KPCA, using SDFs as representations of shape. The initial contour presented on the first line of Figure 12 is the word *square*, in which letters are partially erased. The warping result using linear PCA bears little resemblance with the word *square*. By contrast, using kernel PCA, the word is accurately reconstructed. In particular, a police close to the original one used in the initial contour is obtained. The initial contour presented on the second line of Figure 12 is the word *circle*, occluded by a line. Again, the warping result using linear PCA bears little resemblance with the word *circle*. In fact, the obtained result appears as a mixing between different words. Using KPCA, the word *circle* is not only accurately reconstructed but the line is completely removed. Besides, the original police used for the initial contour (which belongs to the training set) is preserved. The initial contour presented on the third line of Figure 12 is one of the registered training words *yellow*, slightly translated. The letter "y" is also replaced by a rectangle. Employing kernel PCA, the word *yellow* is perfectly reconstructed. In particular, the letter "y"

is recovered, the contour is translated back and the original police used for the initial contour is preserved. In comparison, the word *yellow* is barely recognizable from the final contour obtained using linear PCA.

In each of the experiments above, the accurate word (i.e., closest to the word used to build the initial contour) is detected and reconstructed, using kernel PCA. Similar robust performance were obtained with kernel PCA on binary maps. The shape of the final contour obtained with linear PCA, was oftentimes the result of a mixing between words of different classes. This mixing between classes can lead to unrealistic shapes. Thus, KPCA appears to be a strong method for introducing shape priors within the GAC framework, when training sets involving different types of shapes are used.

3.2.2.4 Influence of the parameter σ in exponential Kernels

The goal of this section is to study the influence of the parameter σ, when kernel PCA is performed using exponential kernels. Figure 13 presents warping results of an arbitrary shape using different values of the parameter σ for the kernel defined in Equation (24), which corresponds to performing kernel PCA on binary maps. The Shark training set was used for the shape learning.

As can be noticed in Figure 13, as the value of σ increases more and more mixing among the shapes of the training set are allowed. Such shape mixing naturally occurs, when linear PCA is used for learning (e.g., in Figure 9, second row (d), this type of mixing can be observed). Similar results were obtained for the SDF representation and the kernel given in equation (23). Hence, the parameter σ allows for *controlling* the degree of mixing allowed amongst the learned shapes in the shape prior, which is another advantage of kernel PCA over linear PCA. The choice of σ should typically depends upon how much shape variation occurs within the data set. To emphasize, larger σ's allow for more mixing among the shapes, whereas smaller σ's lead to more selective shape priors.

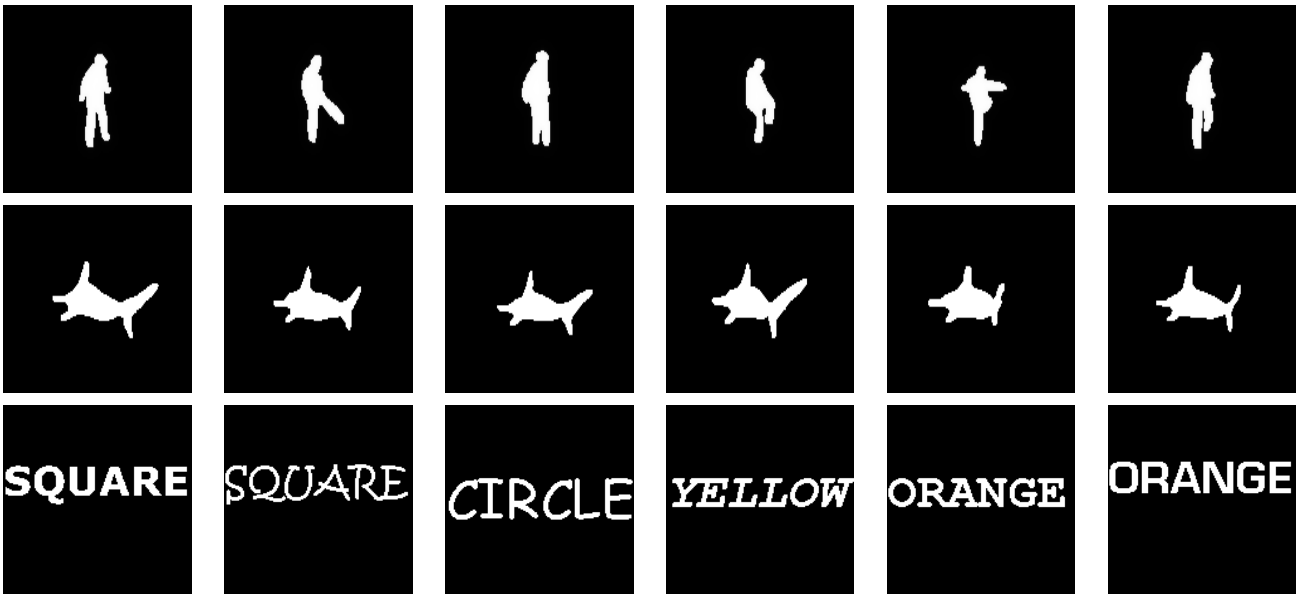


Figure 7: Three training sets (Before alignment - Binary images are presented here). First row, Soccer Player Training Set (6 of the 22 used). Second row, Shark Training Set (6 of the 15 used). Third row, 4-Words Training Set (6 of the 80 learned; 20 fonts per word).

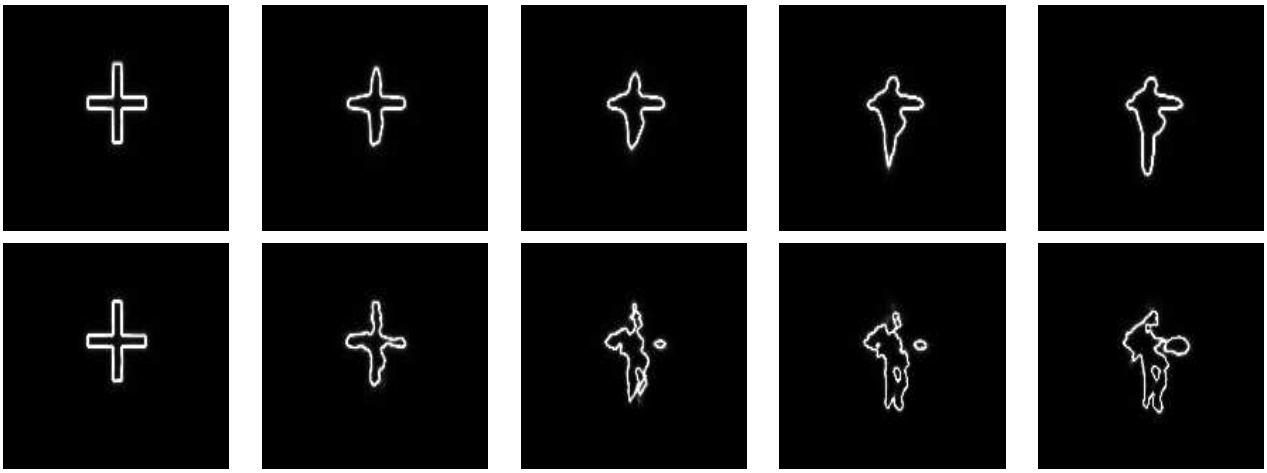
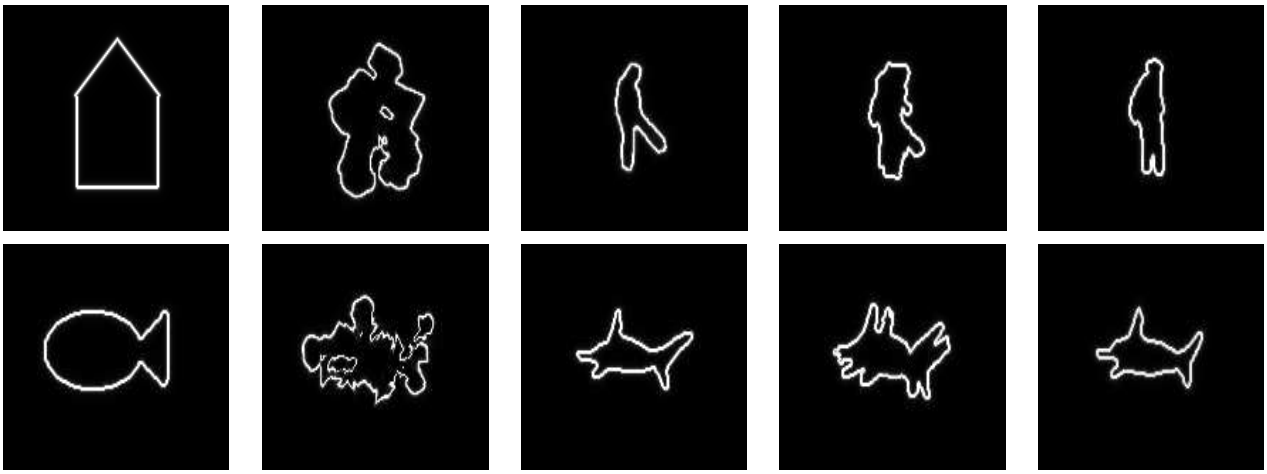


Figure 8: Warping results obtained for the Soccer Player Training Set, starting with an initial contour representing a cross. First line: Evolution obtained using kernel PCA; Second line: Evolution obtained using linear PCA.



(a) (b) (c) (d) (e)

Figure 9: Warping 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.

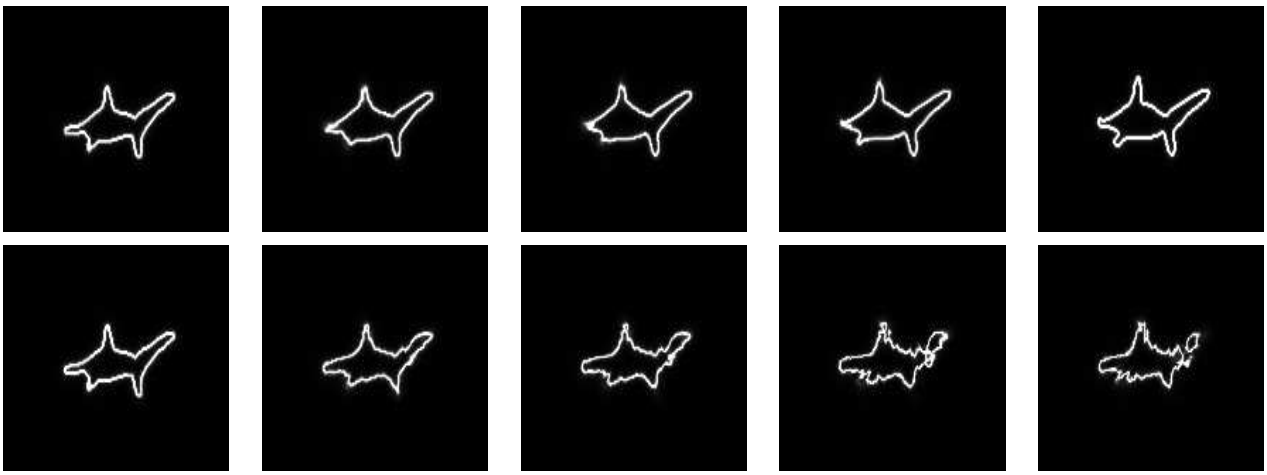


Figure 10: Warping results obtained for the Shark Training Set. The initial contour (leftmost image) represents one of the learned shapes slightly misaligned (5 pixels) with the corresponding element of the registered training set. First line: Evolution obtained using kernel PCA; Second line: Evolution obtained using linear PCA.

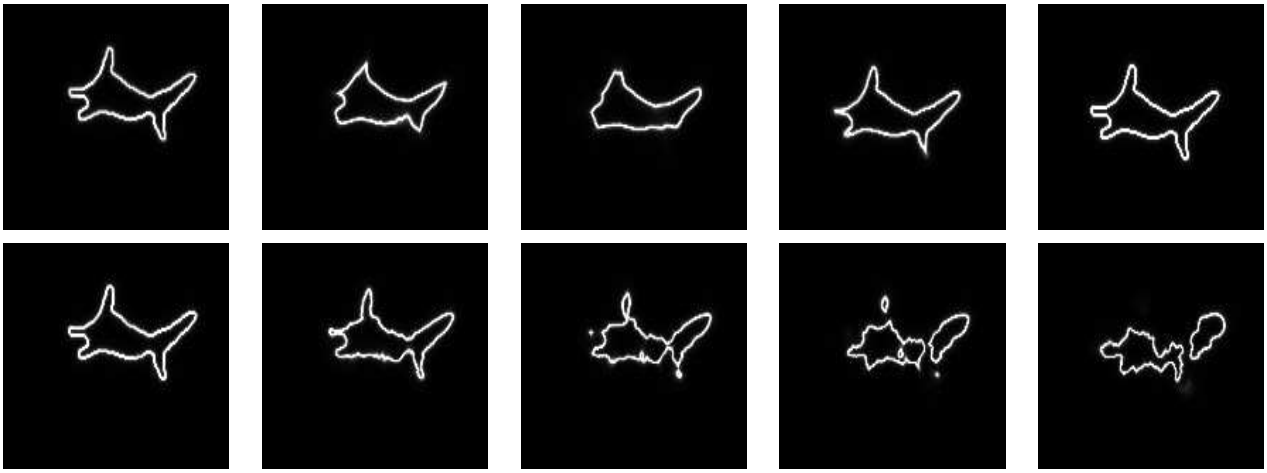


Figure 11: Warping results obtained for the Shark Training Set. The initial contour (leftmost image) represents one of the learned shapes misaligned of approximately 15 pixels. First line: Evolution obtained using kernel PCA; Second line: Evolution obtained using linear PCA.

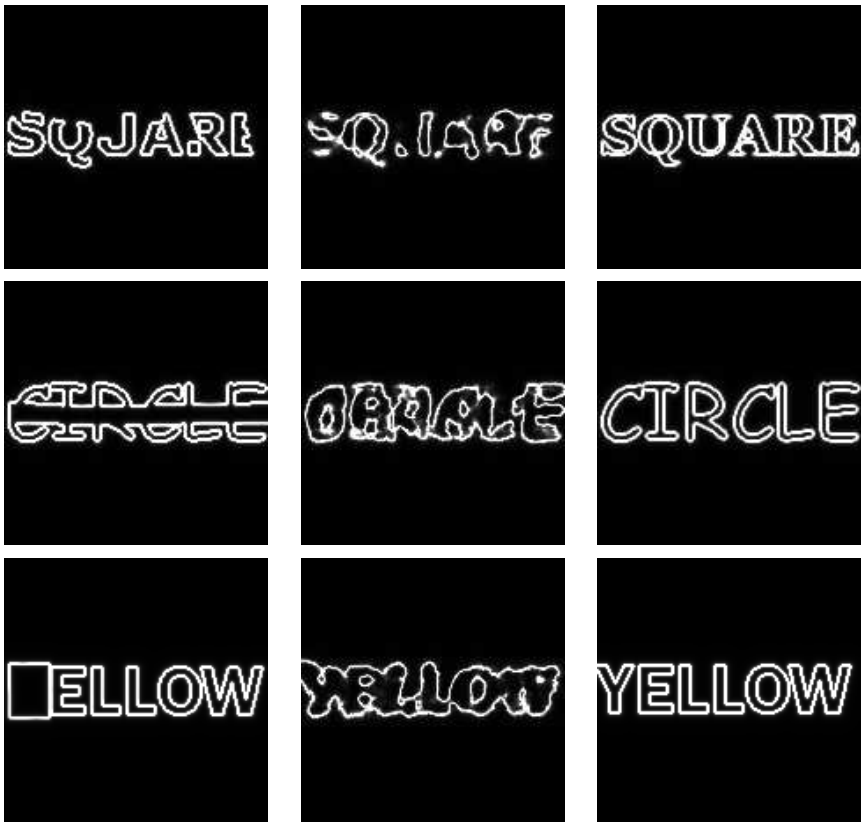
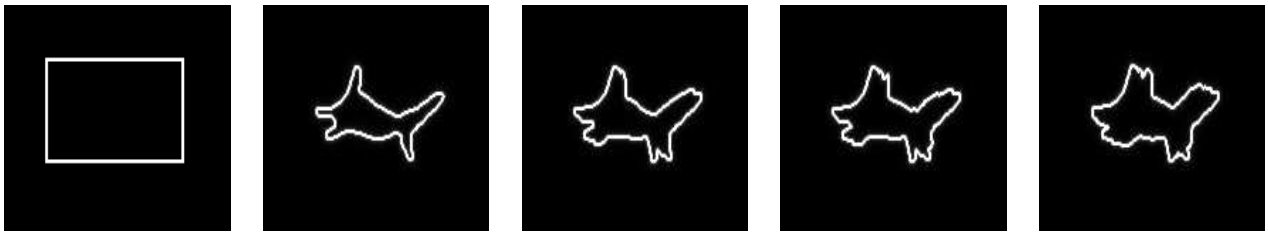


Figure 12: Warping Results for the 4-Words Training Set. *Left column*: Initial contours; *Middle Column*: Warping using linear PCA; *Right Column*: Warping using kernel PCA.



(a) (b) (c) (d) (f)

Figure 13: Influence of σ for the kernel PCA method (exponential kernel) applied on binary maps. Warping results of an arbitrary shape are presented for the Shark Training Set. (a): Initial shape, (b): Warping result for $\sigma = 3$, (c): $\sigma = 7$, (d): $\sigma = 9$, (e): $\sigma = 15$.

CHAPTER IV

FUTURE WORK

4.1 Proposed Research

The object of the proposed research is to develop statistical as well as shape-based tools for geometric/geodesic active contours. A particular emphasis will be put on the differences, in terms of performance notably, with currently available techniques. The developed tools are expected to be useful for both segmentation and tracking. Our goal is to develop robust techniques to segment complex images or track difficult video sequences that are corrupted with noise, occlusion, or turbulence.

The focus of our future work will be on developing novel segmentation techniques and frameworks that combine information extracted from images as well as information learned a priori from examples. In particular, we plan to study a region-based segmentation technique that uses the result of image thresholding to drive the evolution of the contour. Preliminary results show that the envisioned method can lead to very sensible segmentations, where currently available methods perform poorly. Also, we plan to study other learning methods, such as locally linear embedding (LLE) or kernel LLE, to perform shape analysis and compare them with the kernel PCA technique presented in this report.

Besides, we plan to pursue the work begun in [41] and study a novel methodology for evolving contours. The evolution technique is based on the theory of interacting particles systems and is envisioned to lead to a stochastic interpretation of geodesic active contours. In [41], the case of convex curves was considered. We propose to investigate the general case of non-convex curves and consider applying the technique to the segmentation of images.

4.2 Work Remaining to be done

The use of kernel PCA to learn shapes, as presented in Section 3, is completed. The use of other learning techniques, such as Locally Linear Embedding (LLE) or kernel LLE, is well advanced.

Three main tasks remain to be carried to develop robust techniques for segmenting challenging images or tracking complex video sequences:

1. Investigate novel region-based segmentation techniques
2. Develop novel frameworks combining the knowledge of shapes with region-based techniques
3. Investigate a novel method to perform the stochastic curvature evolution of active contours.

It is expected that this work will be entirely completed by Spring 2008.

4.3 Facilities and Equipment Needed

To perform the above tasks, the following equipments and facilities are needed:

	Facility	Status
Hardware	Computer workstation	acquired
Commercial Software	Matlab	installed
	VC++ compiler	installed
	Windows XP	installed
Noncommercial Software	ITK	installed
	VTK	installed

Table 1: Needed facilities.

All the commercial softwares are available through a site license at the Georgia Institute of Technology.

APPENDIX A

LEVEL-SET EVOLUTION

Level set representations were introduced by Osher and Sethian [37, 25] to model interface motion and became a popular tool in the fields of image processing and computer vision. The idea consists of representing a contour by the zero level set of a smooth Lipschitz continuous function. A common choice is to use a signed distance function for embedding the contour. The contour is propagated implicitly by evolving the embedding function to decrease a chosen energy functional. Implicit representations present the advantage of avoiding to deal with complex re-sampling schemes of control points. Moreover, the contour represented implicitly can naturally undergo topological changes, such as splitting and merging.

The signed distance function used to represent the contour of interest is usually denoted by ϕ. The signed distance function ϕ is a function $\phi : \Omega \mapsto \mathbf{R}$ and the contour corresponds to the zero level set of ϕ, i.e. $\{(x, y) \in \Omega / \phi(x, y) = 0\}$. The convention $\phi(x, y) \geq 0$ can be taken for points (x,y) belonging to the interior of the contour.

Typically, the energy functional is minimized via a gradient descent approach. Accordingly, one defines an energy functional in terms of ϕ, namely $E(\phi)$ (see e.g., [27, 25, 37] and references therein), and then the contour evolution is derived via the following gradient descent flow:

$$\frac{d\phi}{dt} = -\nabla_{\phi} E \tag{31}$$

This amounts to deforming the contour from an initial contour ϕ_0 in order to minimize the energy $E(\phi)$.

REFERENCES

[1] Blake, A. and Isard, M., eds., *Active Contours*. Springer, 1998.

[2] Caselles, V., Catte, F., Coll, T., and Dibos, F., "A geometric model for active contours in image processing," *Numerische Mathematik*, vol. 66, pp. 1–31, 1993.

[3] Chan, T. and Vese, L., "Active contours without edges," *IEEE Trans. on Image Processing*, vol. 10, no. 2, pp. 266–277, 2001.

[4] Chen, Y., Huang, T., and Rui, Y., "Parametric contour tracking using unscented kalman filter," *Proceedings of the International Conference on Image Processing*, vol. 3, no. 3, pp. 613–616, 2002.

[5] Cootes, T., Beeston, C., G.Edwards, and Taylor, C., "Unified framework for atlas matching using active appearance models," in *Int'l Conf. Information Processing in Med. Imaging*, pp. 322–333, Springer-Verlag, 1999.

[6] Cootes, T., Taylor, C., and Cooper, D., "Active shape models-their training and application," in *Comput.Vis. Image Understanding*, vol. 61, pp. 38–59, 1995.

[7] Cremers, D., Kohlberger, T., and Schnoerr, C., "Diffusion snakes: introducing statistical shape knowledge into the mumford-shah functional," in *International journal of computer vision*, vol. 50.

[8] Cremers, D., Kohlberger, T., and Schnoerr, C., "Shape statistics in kernel space for variational image segmentation," in *Pattern Recognition*, vol. 36, pp. 1292–1943, 2003.

[9] Cremers, D., Osher, S., and Soatto., S., "Kernel density estimation and intrinsic alignment for knowledge-driven segmentation: teaching level sets to walk," in *Proc. of DAGM*, vol. 3157, pp. 36–44, 2004.

[10] Dambreville, S., Rathi, Y., and Tannenbaum., A., "Non linear shape prior from kernel space for geometric active contours," in *Proceedings of SPIE*, vol. 6064, pp. 404–412, 2006.

[11] Dambreville, S., Rathi, Y., and Tannenbaum, A., "Unscented kalman filtering applied to geometric active contours for tracking deformable objects," in *Invited paper in Automatic Control Conference*, 2006.

[12] Isard, M. and Blake, A., "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[13] Jackson, J., Yezzi, A., and Soatto, S., "Tracking deformable moving objects under severe occlusions," in *Conf. decision and control*, Dec, 2004.

[14] Julier, S. and Uhlmann, J., "A new method for the nonlinear transformation of means and covariances in filters estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2000.

[15] Julier, S. and Uhlmann, J., "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–420, 2004.

[16] Kalman, R. E., "A new approach to linear filtering and prediction problems," *Transactions of the ASME - Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[17] Kim, J., Fisher, J., Yezzi, A., Cetin, M., and Willsky, A., "Nonparametric methods for image segmentation using information theory and curve evolution," in *Proc. ICIP*, 2002.

[18] Kwok, J. and Tsang, I., "The pre-image problem in kernel methods," in *IEEE transactions on neural networks*, vol. 15, pp. 1517 –1525.

[19] Leventon, M., Grimson, E., and Faugeras, O., "Statistical shape influence in geodesic active contours," in *Proc. CVPR*, pp. 1316–1324, IEEE, 2000.

[20] Li, P., Zhang, T., and Ma, B., "Unscented kalman filter for visual curve tracking," *Image and Vision Computing*, vol. 22, no. 2, pp. 157–164, 2004.

[21] Mercer, J., "Functions of positive and negative type and their connection with the theory of integral equations," in *Philos. Trans. Roy. Soc. London*, 1909.

[22] Mika, S., Scholkopf, B., and Smola, A., "Kernel pca and de-noising in feature spaces," in *Advances in neural information processing systems*, vol. 11, 1996.

[23] Mumford, D. and Shah, J., "Optimal approximation by piecewise smooth functions and associated variational problems," *Commun. Pure Applied Mathematics*, vol. 42, pp. 577–685, 1989.

[24] Niethammer, M. and Tannenbaum, A., "Dynamic geodesic snakes for visual tracking," in *Proc. CVPR*, vol. 1, pp. 660–667, 2004.

[25] Osher, S. and Fedkiw, R., *Level Set Methods and Dynamic Implicit Surfaces*. Springer Verlag, 2003.

[26] Osher, S. J. and Sethian, J. A., "Fronts propagation with curvature dependent speed: Algorithms based on hamilton-jacobi formulations," *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988.

[27] Paragios, N., Chen, Y., and Faugeras, O., *Handbook of Mathematical Models in Computer Vision*. Springer, 2005.

[28] Paragios, N. and Deriche, R., "Geodesic active contours and level sets for the detection and tracking of moving objects," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 3, pp. 266–280, 2000.

[29] Peterfreund, N., "Robust tracking of position and velocity with Kalman snakes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 6, pp. 564–569, 1999.

[30] Peterfreund, N., "The velocity snake: deformable contour for tracking in spatio-velocity space," *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 346–356, 1999.

[31] Rathi, Y., Vaswani, N., Tannenbaum, A., and Yezzi, A., "Particle filtering for geometric active contours with application to tracking moving and deforming objects," in *CVPR 2005*.

[32] Rousson, M. and Paragios, N., "Shape priors for level set representations," in *Proceedings of European Conference on Computer Vision*, pp. 78–92, 2002.

[33] Sapiro, G., ed., *Geometric Partial Differential Equations and Image Analysis*. Cambridge Press, 20010.

[34] Scholkopf, B., Mika, S., and Muller, K., "Nonlinear component analysis as a kernel eigenvalue problem," in *Neural Computation*, vol. 10, 1998.

[35] Scholkopf, B., Mika, S., and Muller, K., "An introduction to kernel-based learning algorithms," in *IEEE Trans. on Neural Networks*, vol. 12, 2001.

[36] Sethian, J. A., "A review of recent numerical algorithms for hypersurfaces moving with curvature dependent speed," *J. Differential Geometry*, vol. 31, pp. 131–161, 1989.

[37] Sethian, J. A., *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 2nd ed., 1999.

[38] Terzopoulos, D. and Szeliski, R., *Active Vision*, ch. Tracking with Kalman Snakes, pp. 3–20. MIT Press, 1992.

[39] Tsai, A. and Yezzi, A., "Model-based curve evolution technique for image segmentation," in *Proceedings of Computer Vision and Pattern Recognition*, vol. 1.

[40] Tsai, A., Yezzi, T., and Wells, W., "A shape-based approach to the segmentation of medical imagery using level sets," *IEEE Trans. on Medical Imaging*, vol. 22, no. 2, pp. 137–153, 2003.

[41] Unal, G. B., Nain, D., Ben-Arous, G., Shimkin, N., Tannenbaum, A., and Zeitouni, O., "Algorithms for stochastic approximations of curvature flows," in *Proceedings of ICIP*, vol. 2, pp. 651–654, 2003.

[42] van der Merwe, R. and Wan, E., "The unscented kalman filter for nonlinear estimation," *Proceedings of IEEE Symposium*, 2000.

[43] van der Merwe, R. and Wan, E., "The square-root unscented kalman filter for state and parameter estimation," *Proceedings of the International Conference on Accoustics, speech, and Signal Processing(ICASSP)*, 2001.

[44] Wang, Y. and Staib, L., "Boundary finding with correspondance using statistical shape models," in *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 338–345, 1998.

[45] Yezzi, A., Kichenassamy, S., and Kumar, A., "A geometric snake model for segmentation of medical imagery," in *IEEE Trans. Medical Imag.*, vol. 16, pp. 199–209, 1997.

[46] Yezzi, A. and Soatto, S., "Deformotion: Deforming motion, shape average and the joint registration and approximation of structures in images," *International Journal of Computer Vision*, vol. 53, no. 2, pp. 153–167, 2003.

[47] Yuille, A., Cohen, D., and Halliman, P., "Feature extraction from faces using deformable templates," in *Proc. CVPR*, pp. 104–109, IEEE, 1989.