Tracking Non-Rigid Objects using Functional Distance Metric

Pavel Laskov and Chandra Kambhamettu
Department of Computer and Information Sciences
University of Delaware
Newark DE 19718
laskov|chandra@cis.udel.edu

Abstract

A novel method for tracking non-rigid objects is presented. The method is based on the functional representation of objects and is applicable to problems of any dimensionality. This representation can be reconstructed by multi-dimensional non-linear regression. When the metric is defined on the functional representation of objects, the tracking step can be performed by selecting the closest candidate representation. A practical algorithm using the Support Vector Regression is presented which possesses the required metric property.

The method is applied to the tracking of facial features in black-and-white intensity images. The features are defined as fixed-size rectangles, initial positions of which are manually selected. The technique exhibits reasonable accuracy, especially if relatively small motion occurs between two successive images. An adaptive search strategy is proposed to avoid exhaustive hypothesis search. The method is robust to presense of glasses and facial hair, as well as to reflection from glass lenses. Computational complexity of the algorithm needs to be improved in order to achieve real-time performance.

1 Introduction

Tracking of non-rigid objects has received significant attention in computer vision community recently. Applications in video surveillance, motion analysis, recognition and navigation often include tracking as a first step.

Most of current approaches to tracking involve construction of a model of an object or of its motion. Such models can be very successful on the applications they are designed for, and indeed attain the state-of-the-art results. However, come new applications, the algorithms have to be re-designed and re-evaluated.

Can the tracking problem be solved with a minimum *a priori* information? The goal of this paper is to inverstigate a technique which might provide a positive answer to this question. Our main philosophy is that, in the absense of prior information, the models of objects (and possibly, of their motions) should be learned from the observations. This ushers in the methods of machine learning aimed at reconstructing mathematical functions from data. The particular technique involved in the proposed algorithm is Support Vector Machines (SVM), the regression formulation of which allows to reconstruct non-linear functions of *any* dimension. Moreover, it provides a mathematically sound distance metric between reconstructed functions. This distance is used to perform the tracking step.

The price to be paid for the generality of the new algorithm is complexity of computation. The algorithm involves multiple regression problems at each iteration and currently cannot compete in speed with the best tracking methods for video sequences. However, the recent advances in speeding-up SVM training algorithms indicate a promise that in future the proposed algorithm can attain real-time performance.

1.1 Related Work

One of the most popular ideas in tracking of non-rigid objects is that of physics-based deformable models, introduced in [11]. It has been widely used for tracking of 3D objects and for non-rigid motion modeling [4] [5] [17]. Coupled with the Kalman filtering

approach, this method has been extended to handle complex motions of non-rigid objects [15] [16]. However the physics-based methods require knowledge of certain physical properties of the bodies in question, which are not always available. Some approaches ([1] [6] [7]) estimate the model properties using a set of cue points manually chosen at initialization stage. In real-life applications manual interaction with the subject might not be possible.

Another flavor of model-based tracking is the deformable-template approach, conceived for identification of eyes and mouth in facial images [19] [20]. It employs hand-crafted mathematical templates to be fit to the edge pattern of an image. Obviously, such models are only applicable to a narrow set of features. Besides, certain properties of facial features have proved to be very difficult to encode by a template: an eye may close, causing a pupil to disappear, or a wide smile may bring teeth into observation.

A powerful method of tracking contours is CONDENSATION [8]. It employs a general mathematical model of a contour (B-spline), and, given the observation, performs tracking by propagating conditional density of this model. An extension of this method, ICONDENSATION [9], improves the performance in cluttered scenes by using importance sampling, guided by an additional information channel. A limitation of this method is applicablity to one-dimensional contours.

The most successful "model-less" approach to tracking, due to Black and Yacoob [2], is based on a model of motion which is obtained from computing optical flow. Although this method has proved to be extremely practical, we bear in mind that modeling motion is itself a challenging problem, especially for non-rigid motion. It is also unclear how this method can handle the presence of extraneous attributes, such as glasses or facial hair, in the image. The problem arises from the fact that such attributes undergo a different kind of motion, which poses significant difficulty for optical flow estimation.

Lastly, a very interesting recent approach is the "active blobs" [13]. It uses a very simple triangulated shape model, coupled with the appearance model as a texture map. The tracking step is performed by solving the registration problem. This method is conceptually close to the proposed technique in that its models are learned from the data.

1.2 Approach

The main idea of the proposed approach is to construct a functional representation of "features of interest" in successive frames and to select the "closest feature" as the new tracking location. The particular meaning of features can vary among applications. In tracking of facial features, reported as application in this article, such features are image intensities as 2D functions of spacial coordinates in rectangular areas of interest. The only requirement for the representation of functions is that the space of such functions possess a metric. Hence the choice of SVM as a function estimator. Only very weak assumptions have to be made: the class of function to be used in approximation (e.g. polynomials, RBF, splines etc.), and some general parameters of such functions and of the approximation algorithm.

The rest of the paper is organized as follows. We first present the basic formulation of Support Vector Machines. We then describe how a tracking step can be performed with the help of this technique and comment on some implementation details. Lastly, we present experimental results demonstrating viability of the proposed method for tracking of facial features, and finally discuss advantages and limitations of the current method and possibilities for future improvement.

2 Support Vector Machines

The main principle of Support Vector Machines (SVM) is construction of a hyperplane in a high-dimensional linear feature space, to approximate training data in "the best possible way". We will use the regression SVM which recovers a non-linear function from observations, possibly corrupted by noise. Space limitations prevent us from presenting this method in detail, however comprehensive reviews of regression SVM can be found in [14] [18].

Before solving the regression problem we must make an assumption on the class, to which the function solution function belongs (denoted by Q). Even though the actual function may not belong to Q, a good approximation by some $f \in Q$ is considered as

a solution. In our system we used the class of Radial Basis Functions (RBF). Other common choices are polynomials, splines, B-splines, etc.

Given the training sample of labeled data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\}$, $\mathbf{x}_i \in \mathbb{R}^n, y_i \in \mathbb{R}$, we need to find a function $f(\mathbf{x})$ in class Q which best approximates the training data.

The function $f(\mathbf{x})$ is sought so as to minimize the risk functional:

$$R(f(\mathbf{x})) = C \sum_{i=1}^{l} |y_i - f(\mathbf{x})|_\epsilon + \Phi(f(\mathbf{x})) \qquad (1)$$

The first term of R enforces small estimation error, while the second is the regularization term ensuring low complexity of solution. C is the parameter of SVM controlling the trade-off between the two terms. The expression of the error term $|E|_\epsilon$ is peculiar for SVM and denotes the absolute value of E if $E > \epsilon$ and 0 otherwise. ϵ is another parameter of SVM known as "insensitivity".

The solution to the problem of minimizing the functional (1) is sought as a hyperplane in the *feature space*; that is, the space obtained by applying transformation corresponding to Q to all points in the input space $\mathbb{R}^n$. Then the dual to problem (1) can be shown to be the following quadratic programming problem[1]:

$$\begin{aligned} \text{Maximize} \quad & -\epsilon \sum_{i=1}^{l} (\alpha_i^* + \alpha_i) \; + \; \sum_{i=1}^{l} y_i (\alpha_i^* - \alpha_i) \\ & - \; \frac{1}{2} \sum_{i,j=1}^{l} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) K_{ij} \\ \text{s. t.} \quad & \sum_{i=1}^{l} \alpha_i^* \; = \; \sum_{i=1}^{l} \alpha_i \\ & 0 \; \leq \; \alpha_i^* \; \leq \; C, \quad i = 1, \ldots, l \\ & 0 \; \leq \; \alpha_i \; \leq C, \quad i = 1, \ldots, l \end{aligned} \qquad (2)$$

Here α's are the the dual variables being optimized (two for each data point in the training sample), K_{ij} denotes the symmetric *kernel function* $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{\gamma}}$ corresponding to the class of RBF functions, γ is another parameter affecting the smoothness of solution.

[1] Complete derivation of (2) from (1) can be found in [18]

Furthermore, the structure of equation (2) is such that: (a) only one of α_i^*, α_i is non-zero and (b) for some points both α's are zero. The points whose corresponding α's are *not* both zero are called *support vectors*. The set of support vectors is denoted as SV.

From the obtained values of α, α^*, the values of the solution function at any point $\mathbf{x}$ can be computed as:

$$f(\mathbf{x}) = \sum_{i \in \text{SV}} (\alpha_i^* - \alpha_i) K(\mathbf{x}, \mathbf{x}_i)$$

This is expression serves as a model of $f(\mathbf{x})$.

3 Performing a Tracking Step with SVM

The important property of SVM is that they provide a closed-form expression for the inner product between two functions $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ approximated by them. This allows to define an L_2 (Euclidean) distance metric $\|f_1(\mathbf{x}) - f_2(\mathbf{x})\|_2$ in class Q of solutions. This metric can be used to perform the tracking step.

Given two target functions $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ (their arguments from now on will be dropped to simplify notation) it can be shown that:

$$\langle f_1, f_2 \rangle = \sum_{i \in \text{SV}_1} \sum_{j \in \text{SV}_2} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) K(\mathbf{x}_i, \mathbf{x}_j)$$

Then, by definition of the L_2 norm,

$$\begin{aligned} \|f_1 - f_2\|_2 &= \langle (f_1 - f_2), (f_1 - f_2) \rangle \\ &= \langle f_1, f_1 \rangle + \langle f_2, f_2 \rangle - 2\langle f_1, f_2 \rangle \end{aligned}$$

Suppose at time t it has been established that some feature is approximated by a function f_0. Let $R \subset Q$ denote a set of candidate features at time $t+1$ approximated by functions $f_i \in R$. The feature at time $t+1$ is selected as the closest among the candidate features:

$$f^{t+1} \; = \; \underset{f_i \in R}{\text{argmin}} \; \|f_i - f_0\|_2 \qquad (3)$$

Notice that the distance between two features is measured in the abstract space of functions; it bears no physical meaning. It characterizes deformation between two frames in a weak sense, as no explicit model of deformation is constructed.

4 Technicalities

As an application of the proposed tracking algorithm we considered the problem of tracking facial features (the eyes, the nose and the mouth) from black-and-white intensity images. The features are represented as rectangles of pixels. The candidate features are obtained by shifting the area of interest by a certain number of pixels in both directions. The size of the patch remains constant.

In order to reduce computational load we consider every k-th point in the feature, where the number k is a parameter of the system denoted as "grain size". Even though the accuracy of estimation of intensity functions decreases as grain size grows, as long as the relative order of distances between the functions remains intact, the tracking accuracy will not suffer.

Another important technical decision is defining the set of candidate features. A larger set can clearly handle larger feature motions but imposes performance penalty growing quadratically with the "search window" size. A better approach to this problem is the adaptive search algorithm.

The idea of adaptive search is to explore only elementary displacements of a frame at every given search iteration. The size of such displacements in pixels is determined by the parameter "shift step". Let r_0 be the feature in the image before motion, and let r_c denote the "current" feature in the frame after motion, initialized to the direct counterpart of r_0 (i.e. with no displacement). We first identify the best feature among the candidates obtained by elementary displacements to r_c. If the best feature is different from r_c we apply elementary displacements to that feature and continue until either no further decrease of distance is possible among the candidate features, or when the maximal search depth (specified by another parameter) has been reached. The summary of the algorithm is given in Algorithm 1.

Algorithm 1 Adaptive search algorithm

```
Estimate f_0 and f_c (the functions of r_0 and r_c).
while (search_depth < max_search_depth) do
  Compose the set of candidate feature R of all features obtained by applying elementary displacements to r_c. Estimate intensity functions for features in R.
  Let r_n be the best feature in R as in (3).
  if (||f_n − f_0|| < ||f_c − f_0||) then
    r_c = r_n
    search_depth++
  else
    return r_c
  end if
end while
```

5 Experimental Results

We have tested the SVM-based tracker on a number of sequences of different difficulty. The experiments were conducted on an SGI Octane with 195 MHz clock. ϵ-insensitivity was fixed at 0.03, the box constraint $C = 0.4$. RBF kernel with $\gamma = 25$ was used for all experiments. Grain size of 8 was used for the eyes, 10 for the nose and 12 for the mouth. The shift step of 2 and maximum search depth of 15 were used in the adaptive algorithm, which made it possible to account for up to 30-pixel image shifts in either direction (horizontal and vertical).

The first sequence (Figure 1) has been recorded at the rate of 30 frames/sec and contains fairly smooth motion. Four features are tracked: both eyes, the nose and the mouth. One can see that the tracker is quite successful in its job except for a marginal accumulated horizontal error of the nose frame. This sequence also demonstrates the robustness of the SVM-tracker to presense of glass frames and reflection from the lenses.

The second sequence (Figure 2) has also been recorded at 30 frames per second but has more jerky motion and some dropped frames due to limited bandwidth. This sequence shows the tracker's ability to adjust to non-rigid motion within the tracking frame: appearance of the tongue or teeth in the mouth does not confuse it, although it remains bound to a rigid rectangular frame.

The non-smooth motion between two frames can cause a problem in tracking: in frames 28 and 29 (Figure 2(c) and Figure 2(d)) both eyes suddenly open, thereby significantly changing the intensity functions. The left eye is tracked correctly while on the right eye the new frame is misplaced. The difficulty encountered by the tracker stems from the fact that it maintains smoothness of solution to ensure good generalization. As a result, when sudden transition from low to high contrast occurs, it favors the smoother part of the image which lies outside of the feature frame. Because the method carries out no specific information about the features being tracked, recovery from the wrong tracking step is not possible.

The third sequence recorded at the rate of 15 frames/sec (Figure 3) is considerably more difficult. The large translation between the frames is not handled well by the tracker.

Let us look closer at what constitutes the problem with the tracking step on this sequence. Below is the transcript of the tracking step for the right eye between frames 1 and 2 (Figure 3(a)) and Figure 3(b)). This pair of images has large amount of motion between them. The star denotes the smallest distance at current search depth.

```
input_prototype: 249,197
Distance(0,0) = 2.50412
Distance(-2,-2) = 2.43634
Distance(-2,0) = 2.59064
Distance(-2,2) = 2.30136 *
Distance(0,-2) = 2.46373
Distance(0,2) = 2.33327
Distance(2,-2) = 2.64966
Distance(2,0) = 2.63173
Distance(2,2) = 2.40435
>>
Distance(-4,0) = 2.45167
Distance(-4,2) = 2.39456 *
Distance(-4,4) = 2.58394
Distance(-2,4) = 2.54414
Distance(0,4) = 2.41557
>>
Best step: -2,2
```

One can see that although the absolute values of distances are fairly large (a good match usually has the distance of about 1 or less) the smallest distance at depth up to 2 is less than the smallest distance at depth up to 4. This satisfies the termination condition of the adaptive search algorithm and prevents it from further search. However such inaccuracy in not the limitation of the SVM method as such but rather an artifact of the distance surface. Use of exhaustive enumeration of candidate frames results in sufficiently good tracking even with large amount of motion, provided the search depth is large enough to capture the actual motion between two frames. Figure 4 shows the same sequence as Figure 3, tracked with frame search by enumeration. The problem between frames 1 and 2 is rectified (Figure 4(a) and Figure 4(b)); however later in the sequence there happens a jump exceeding maximal search depth (Figure 4(e) and Figure 4(f),) which cannot be handled by the tracker.

6 Conclusions

We have presented a method for tracking non-rigid objects based on the notion of functional distance. On our testbed application problem of tracking facial features from image intensities the method performs reasonalby accurately when the motion between successive frames is not too large ($\leq$ 10 pixels). The method is robust to presence of extraneous attributes, such as glasses and facial hair, as well as to reflection from glass lenses. Since no explicit models are constructed for particular features, one can apply SVM-based tracking to any conceivable facial features, with the precondition that they possess sufficient but not excessive texture[2].

The proposed method differs from the majority of known tracking methods in that it attempts to perform tracking steps utilizing solely the information that can be learned from the data. Even with the large percentage of support vectors, the algorithm captures significantly richer structure than the simple correlation because its representation is based on a non-linear function and automatically learned weights.

[2]To handle stong contrasting texture one would need automatic tuning of parameters of SVM, in order to adapt smoothness requirements to changing image contrast.

(a) Frame 0

(b) Frame 20

(c) Frame 40

(d) Frame 60

(e) Frame 80

(f) Frame 100

Figure 1: Tracking results, small motion, 30 frames/sec

(a) Frame 0

(b) Frame 20

(c) Frame 28

(d) Frame 29

(e) Frame 59

(f) Frame 60

Figure 2: Tracking results, large motion, 30 frames/sec

(a) Frame 1

(b) Frame 2

(c) Frame 5

(d) Frame 7

(e) Frame 9

(f) Frame 10

Figure 3: Tracking results, large motion, 15 frames/sec

(a) Frame 1

(b) Frame 2

(c) Frame 9

(d) Frame 10

(e) Frame 12

(f) Frame 13

Figure 4: Tracking results, large motion, 15 frames/sec, explicit frame search

Our future work will center on improving the accuracy and the running time of the tracking algorithm. Both can be approached in the multi-resolution framework. Since the non-linear regression is expensive to perform over a wide area of interest, we intend to limit it to the set of candidate frames. This set can be obtained from a faster and perhaps less accurate tracker, for example the one based on correlation, operating on coarser-resolution images. The efficiency of SVM training will also be addressed by employing an advanced decomposition algorithm reported in [12]. The future system will include automatic feature detection.

References

[1] Sumit Basu, Nuria Oliver, and Alex P. Pentland. 3D modeling and tracking of human lip motions. In *Proceedings of the International Conference on Computer Vision*, pages 337–343, 1998.

[2] Michael J. Black and Yasser Yacoob. Recognizing facial expressions in image sequences using local parameterized models of image motion. *International Journal of Computer Vision*, 25(1):23–48, October 1997.

[3] Andrew Blake and Alan Yuille, editors. *Active Vision*. MIT Press, 1992.

[4] Roberto Cipolla and Andrew Blake. The dynamic analysis of apparent contours. In *Proceedings of the International Conference on Computer Vision*, pages 616–623, 1990.

[5] R. Curwen, R., Andrew Blake, and Roberto Cipolla. Parallel implementation of lagrangian dynamics for real-time snakes. In *Proceedeings of the British Machine Vision Conference*, pages 29–35, 1991.

[6] Douglas DeCarlo and Demetri Metaxas. The integration of optical flow and deformable models with applications to human face shape and motion estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 231–238, 1996.

[7] Douglas DeCarlo and Demetri Metaxas. Deformable model-based shape and motion analysis from images using motion residual error. In *Proceedings of the International Conference on Computer Vision*, pages 113–119, 1998.

[8] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *Proceedings of the European Conference on Computer Vision*, volume 1, pages 343–356, 1996.

[9] Michael Isard and Andrew Blake. ICondensation: unifying low-level and high-level tracking in a stochastic framework. In *Proceedings of the European Conference on Computer Vision*, volume 1, pages 893–908, 1998.

[10] Thorsten Joachims. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Christopher Burges, and Alexander Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 169–184. MIT Press, 1999.

[11] M. Kass, A. Witkin, and Demetri Terzopoulos. Snakes: active contour models. *International Journal of Computer Vision*, 1:321–331, April 1987.

[12] Pavel Laskov. An improved decomposition algorithm for regression support vector machines. In S.A. Solla, T.K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 484–490. MIT Press, 2000.

[13] Stan Sclaroff and John Isidoro. Active blobs. In *Proceedings of the International Conference on Computer Vision*, pages 1146–1153, 1998.

[14] Alexander Smola and Bernhard Schölkopf. A tutorial on support vector regression. Technical Report NC2-TR-1998-030, NeuroCOLT2, 1998.

[15] Demetri Terzopoulos and Richard Szeliski. Tracking nonrigid 3D objects. In Blake and Yuille [3], pages 75–89.

[16] Demetri Terzopoulos and Richard Szeliski. Tracking with Kalman snakes. In Blake and Yuille [3], pages 3–20.

[17] Demetri Terzopoulos and K. Waters. Analysis of facial images using physical and anatomical models. In *Proceedings of the International Conference on Computer Vision*, pages 727–732, 1990.

[18] Vladimir N. Vapnik. *Statistical Learning Theory*. J. Wiley and Sons, 1999.

[19] X. Xie, R. Sudhakar, and H. Zhuang. On improving eye feature extraction using deformable templates. *Pattern Recognition*, 17:791–799, June 1994.

[20] Alan Yuille, D. Cohen, and P Halliman. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8:99–111, February 1992.