Head Tracking Using Learned Linear Subspaces

Ramakrishnan Ravindran
rravind@clemson.edu
CES Department
Clemson University SC 29631

Sandeep Hiremath
shirema@clemson.edu
ECE Department
Clemson University SC 29631

Abstract

This paper presents a simple and efficient tracking algorithm based on representing the appearance of objects using learned linear subspaces. The tracker updates this subspace and maintains an updated appearance model of the object making the tracking problem simpler. Using the L^2 reconstruction norm in this framework we provide a simple algorithm for finding a subspace whose uniform L^2-reconstruction error norm for a given collection of data samples is below some threshold. We show some experimental results and some intermediate results to illustrate the various steps involved in this algorithm.

1 Introduction

The main purpose of a visual tracker is to locate an object of interest in a given image sequence and lock on to the target and follow it during the progress of the sequence while maintaining proper focus or placement over the object. In simpler tracking algorithms the shape of the tracking window is kept constant and hence this restricts the tracker to objects which do not have any sort of form or shape variation during the sequence. This works efficiently but has very little use in the real world since most objects tend to move in a given sequence and subtle variations in angle and location can lead to a change of shape or orientation. Hence the simplicity of a tracker which uses a fixed window restricts its real world functionality and hence requires enhancements. The enhancement can be in the form of using several windows to find the exact location of the target so that varying window shapes and sizes can cope with the change in shape of the object. But if this technique is used along with a simple algorithm such as likelihood computation or intensity matching, the performance of the algorithm becomes very poor as for each window; the likelihood or intensity computation has to be made which makes it a tedious process hence slowing it down significantly. An alternative to these techniques is to use previous frames in the sequence to maintain an up to date reference model so as to enhance tracking. Since all the above mentioned techniques use only the current frame or at best the previous frame, the tracker has to check every frame for all possibilities. If several frames are retained, then the results of those frames can be used to create a model of the object of interest and hence the tracking problem is now essentially a detection problem.

This paper proposes an adaptive appearance model for tracking complex natural objects based on the subspace technique. Within the subspace framework, updating the model becomes how to define a subspace L that best approximates a given set of data $\{x_1, \cdot \cdot \cdot, x_N\}$, the observations from the previous frames. What constitutes a good approximation depends on the underlying metric one uses to define the quality of the approximation.

The simplicity of the algorithm lies in the fact that only image intensity values are used to track the object, there is no complex probabilistic estimation nor is there any form of optimization. The tracker still works effectively and is robust against illumination changes and pose variations. The main objective of this paper is to explain how such a simple algorithm works. The organization of the paper is as follows, the basic algorithm is explained first without getting into much detail about the techniques involved within, and this will deal specifically with the computation and updating of the subspace. The next section explains some of the techniques and mathematic operations used in this algorithm in detail. The final section shows a few preliminary results on an image sequence used to test the working of this algorithm. We conclude with a short summary and some ideas for extending this algorithm to make it more efficient.

2 Tracking Algorithm

This section deals with the tracking algorithm. The tracking window is first initialized for the first frame. This can be done manually or using a likelihood map of the image. The tracking window has a rectangular shape and the size and tilt of this window are initialized in the first frame. The window has 5 parameters [a, b, x, y, theta], these are the minor axis, major axis, x co-ordinate of center, y co-ordinate of center and the angle made to the principle axis respectively. Once these have been initialized the tracker can have variations in both scale and angle, meaning there can be a change in the minor and major axes and the angle theta so as to change the size and tilt of the window. To simplify the tracking procedure we have implemented only tilt and hence the minor axis and major axis are fixed. Theta can be varied to compensate for pose variation (e.g. tilting of a head). After the first frame, at each frame, the tracker maintains an up-to-date model and the tracking task becomes a detection problem.

To estimate the location of the target in the current frame, we sample S windows of different sizes and locations near the targets location in the previous frame. The content of each window is then rectified to a fixed size. By rasterizing the rectified windows, we obtain the content of the windows as a set of points in some vector space IR^k. At any given frame the appearance model is represented as a linear subspace L in the vector space IR^k. The L^2-distance between each x_i and L is computed and the state of the target at current frame is defined to be the window w_i such that its corresponding x_i minimizes the distance to the subspace L among all $\{x_1, \cdot\ \cdot\ \cdot, x_s\}$.

3 Updating the Subspace

Under the subspace framework the update strategy is to search for a linear subspace L that best approximates the collection of data samples. These are the observations from the previous frames. A pair of input parameters is specified, (N, ∂).

Here N denotes the number of frames who's tracking results we retain and ∂ is the threshold. For the first N frames the results get stored one after the other and the vector space in effect grows in size. When the frame N+2 is encountered, the first frame in the sequence is pushed out and the whole sequence is shifted one position left adding the result of the frame N+1 to the end. Now the vector space has been updated and the window approximation is done using these values. As the sequence progresses, this operation are repeated hence maintaining an up-to-date appearance model of the target.

3.1 Rectification

Rectification is the process of converting the various windows into a window of fixed size. Since the window size in the image sequence is quite large and consists of several intensity values, the size is reduced to improve the performance of the tracker. The complexity of this algorithm can vary but to have an efficient tracker we have used a very simple algorithm for rectification which effectively reduces the size of the window by half. If repeated the window gets halved yet again. This can be done using minimal steps but it still retains much of the detail. This is done by taking the mean of consecutive pixels within the window in both the X and the Y direction. This in effect reduces the window size in half. This is repeated till the window is significantly smaller but still contains enough detail of the target so that the operation of the tracker is not deterred.


The actual window


Window rectified once


Window rectified twice [1]

---

[1] In this example the image was large and hence the window size was also significantly large, so the window had to be rectified twice to bring it to $1/4^{th}$ the original size.

It can be seen from the above example that much of the detail within the window has been retained even after it has been rectified twice effectively reducing the window size to $1/4^{th}$ it's original size.

3.2 The Gramm-Schmidt Algorithm

The Gramm-Schmidt algorithm is used to compute the orthonormal basis for the subspace V that a set of vectors span. This can be used to compute the closest approximation of the current window to the previous results obtained.

$$Uj = [\ 1 / |Wj|\ ].Wj$$

where $$Wj = Vj - \sum_{i=1}^{j-1} (Vj.Ui).Ui$$

E.g.: If the orthonormal basis is to be found for two vectors V_1 and V_2, V_1 is first normalized to compute U_1, then a vector W_2 is constructed orthogonal to U_1.

$$W_2 = V_2 - a.U_1.$$

Once this is found, the second element U_2 is found by normalizing W_2.

This process is continued for the next vector V_3 and so on

$$W_3 = V_3 - (V_3.U_1).U_1 - (V_3.U2).U_2 \text{ and}$$
$$U_3 = [\ 1 / |W_3|\ ].W_3$$

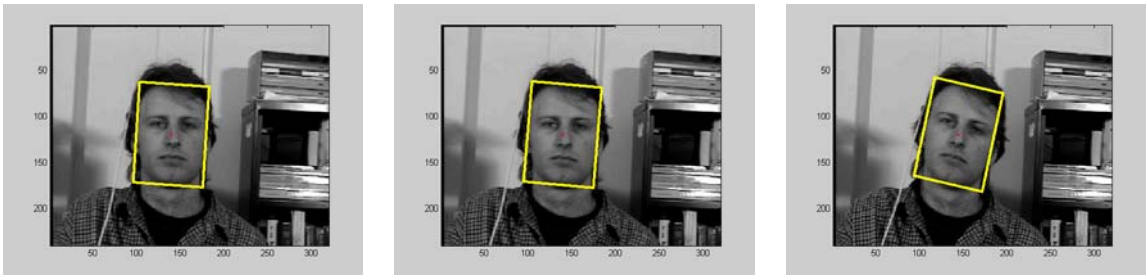
Once the orthonormal basis has been found, finding the closest match to the current frame is just a matter of computing distance between the vectors.

4 Remarks

The most prominent feature of the algorithm is its simplicity: the tracking algorithm simply takes the tracking results over a constant interval and uses these to form the linear subspace. No prior model learned off-line is used by the algorithm. The algorithm operates on the pixel intensity values only, and there is no sophisticated probability estimates, non-linear optimization or filtering of images. One of the problems with this algorithm has to do with the problem of drift. Without learning the model offline, or having a reference model, it is nearly impossible to guarantee a drift free algorithm.

One possible method for enhancing the tracker's ability against drift is to always include the observation made in the first frame in the appearance model. Among all the tracking results made along the video sequence, only the first observation provides an accurate model of the object. Observations made in the subsequent frames will invariably be associated with non-zero probabilities that they are not the tracked object. Therefore, it makes sense to include the first result in the subspace so as to improve the algorithms performance against drift issues. With this small variation, a substantial improvement can be noticed in the performance of the tracker against drift.

5 Results



Tracking the head in an image sequence

Rectified windows for the above frames of image sequence

6 Conclusion

In this paper, we have introduced a technique for learning on-line a representation of the appearances of an object that is being tracked. This is done by representing the appearances as a linear subspace and choosing a constraint using a well chosen metric, the resulting tracker is both simple and fast. The simplicity yet impressive performance of this tracker is one of the primary reasons for implementing this algorithm. There are several enhancements that can be made to this algorithm to make it more efficient.

One of the ways to improve the performance is to divide the N samples from previous results into batches of size k. This follows the reasoning that the target will not have significant pose variations within k frames and the variations will be minor. This can be set as a threshold and k can be determined accordingly. Once this is done the batch mean can be computed and the results which are closest to the batch mean are retained while the rest are dumped. The batch means are used as the subspace and using the Gramm-Schmidt process, the closest approximation can be found.

Another obvious extension to this algorithm is the inclusion of scaling. When scaling is introduced into the algorithm, rectification becomes a much more complicated algorithm and hence may require a lot more computation. But scaling gives the tracker the ability to track the target even if it moves towards or away from the camera. The tracker implemented without scaling will not be able to handle such scenarios.

One of the major parts which can be looked upon to affect the performance of the tracker is the variable N. If N is too small, the number of frames retained is lesser and hence the trackers ability to cope with pose variations might be affected, but the speed of the algorithm is improved. On the other hand if N is too large, the speed is affected but the tracker will have a lot more information on the target and hence will be better at tracking. A slight problem might occur if the tracker loses the target for a few frames (especially when closer to N and if N isn't that large), then the appearance model is not the required target and the tracker will not be able to acquire the target again. This can be prevented by using a global model which is also initialized in the first few frames and this can be referred occasionally to check if the tracker is still locked on to the target or whether it has totally lost the target. But a threshold level has to be set for this also since due to pose variation and lighting changes, the target might be different from the global appearance model. These are areas where further work can be done to extend this algorithm.

References

[1] Ho, J. Kuang-Chih Lee, Ming-Hsuan Yang Kriegman, D. Visual tracking using learned linear subspaces. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages* I-782- I-789 Vol.1*, 2004*

[2] S. Birchfield. An elliptical head tracker. In *Proc. of the 31st Asilomar Conf. on Signals, Systems and Computers*, 1997.

[3] Eric Carlen. Notes on the Gramm-Schmidt Procedure for Constructing Orthonormal Bases.

[4] Kuang-Chih Lee, Jeffrey Ho, David Kriegman. Acquiring Linear Subspaces for Face Recognition under Variable Lighting

[5] A. M. Baumberg and D. C. Hogg. An efficient method for contour tracking using active shape models.

In *Proceedings of the IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 194–199, 1994.

[6] M. Black and A. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *Proc.European Conf. on Computer Vision*, pages 329–342, 1996.

[7] Sumit Basu, Irfan Essa, and Alex Pentland. "Motion Regularization for Model-Based Head Tracking." In Proceedings of the IEEE Int'l Conference on Pattern Recognition (ICPR '96). Vienna, Austria. 1996.

[8] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*, volume 2. Reading, Mass.: Addison-Wesley, 1993.

[9] H. P. Graf, E. Cosatto, D. Gibbon, M. Kocheisen, and E. Petajan. Multi-modal system for locating heads and faces. In *Proc. of the Second Intl. Conference on Automatic Face and Gesture Recognition*, pages 88–93, 1996.

[10] La Cascia, M., Sclaroff, S., and Athitsos, V. Fast. Reliable Head Tracking under Varying Illumination: An Approach Based on Robust Registration of Texture-Mapped 3D Models, In *Proceedings of IEEE Conference on Pattern Analysis and Machine Intelligence (PAMI), 22(4), April, 2000.*

[11] Gregory D. Hager, and Peter N. Belhumeur. Efficient Region Tracking With Parametric Models of Geometry and Illumination. In *Proceedings of IEEE Conference on Pattern Analysis And Machine Intelligence, 1998.*

[12] David Ross, Jongwoo Lim, and Ming-Hsuan Yang. Adaptive Probabilistic Visual Tracking with Incremental Subspace Update