

Reinforcement Learning for Robust and Efficient Real-World Tracking

Andre Cohen, Vladimir Pavlovic
Dep. of Computer Science, Rutgers University
Piscataway, NJ 08854
{acohen, vladimir}@cs.rutgers.edu

Abstract

In this paper we present a new approach for combining several independent trackers into one robust real-time tracker. Unlike previous work that employ multiple tracking objectives used in unison, our tracker manages to determine an optimal sequence of individual trackers given the characteristics present in the video and the desire to achieve maximally efficient tracking. This allows for the selection of fast less-robust trackers when little movement is sensed, while using more robust but computationally intensive trackers in more dynamic scenes. We test this approach on the problem of real-world face tracking. Results show that this approach is a viable method for combining several independent trackers into one robust real-time tracker capable of tracking faces in varied lighting conditions, video resolutions, and with occlusions.

1 Introduction

There is the wide range of contexts where one may desire to track faces for instance in airports for security, in classrooms for student authentication, and in video repositories which require indexing. Each context, however, presents a different combination of image resolutions, lighting conditions, frame rates, camera angles, and changes in face poses/expressions. Inevitably each different context adds a certain amount of complexity to the tracker which makes real-time tracking difficult and imprecise.

Knowledge based classifiers such as those that use color models [2] and facial shape methods [3] while computationally efficient, are often restricted in the contexts which they can operate. Issues of selecting an appropriate set of faces to be part of the prior knowledge is also a difficult dilemma. Face detectors such as Haar-like feature AdaBoost-based classifiers have been shown to be very robust, particularly in contexts with

few pose and illumination changes [8, 5]. While generally more computationally demanding, they often are only as good as the dataset used to train them. This severely restricts the number of contexts which they can operate in. Statistical and adaptive trackers, based on the Kalman filter paradigm [1], or mean shift trackers [12] while computationally efficient are prone to drifting.

To design an accurate tracker robust to changes in illumination and pose [6] has shown that weighted combination of different trackers is advantageous, despite being very computationally inefficient. Similarly, a two tracker model [7] using skin color and shape models have also proved effective, but insufficiently accurate. Dynamic methods using semi-supervised learning, c.f., [10] have also been effective, but often unable to reach real-time tracking. We propose a tracker that takes advantage of adaptive trackers and detectors while explicitly minimizing the amount of time necessary to produce accurate object position estimates. Instead of having fixed beliefs about each independent tracker, we chose, at each time, an optimal tracker among the set of possible tracker so that the expected future loss of accuracy *and* the cost of running the trackers is minimized.

2 Notation

Video I is a sequential set of images I_t . Every $I_t \in I$ has a target described by the bounding box $x_t = [c_x, c_y, \rho, \phi]^T$, where the first two elements are the center of the bounding box, ρ is the scale, and ϕ is the (in-plane) rotation wrt horizontal axis.

3 Proposed method

The task of combining several independent trackers into one robust and efficient tracker can be modeled as a Markov decision process (MDP). Generally an MDP is defined as a set of states S , set of actions A , an immediate reward function $R(s, a)$ for having executed action

a in state s , and a transition function $T(s, a, s')$ describing the probability of getting to state s' given action a was taken at state s . At every time step the agent, or in this case the tracker selector algorithm, given a particular state must pick the action (tracker) which has the highest expected reward. To do so an optimal policy π must be learned which specifies the action $\pi(s)$ that the agent must make given the current state s .

While there are many ways to describe the states while performing tracking, to make learning manageable we constrain the state space to two components. First, the amount of movement sensed in the video is used to aid in selecting the most suitable tracker. This is based on the observation that different trackers perform at varying levels of accuracy depending on the amount of movement in the video. Second, we want to avoid any single tracker from being used too many times in a row. This is because many trackers are prone to drifting if used alone for an extended amount of time. Moreover, using a single computationally expensive track may result in subpar computational performance in real-world settings.

In the context of this paper states are represented as $s = [\hat{F}, \kappa]$, where $\hat{F} \in \mathbb{I}$ is a discretized measure of average optical flow magnitude, and $\kappa \in \mathbb{N}^{|A|}$ is the number of times an action (tracker) has been used in a row. To calculate \hat{F} the Horn–Schunck method [4] is used to determine the flow from I_{t-1} to I_t . Since we are interested only in the flow pertaining to the target, only the area around the previous target state x_{t-1} is used in the optical flow calculation.

3.1 Learning

To learn the optimal policy π reinforcement learning (RL) is used. The optimal policy we look for is one that maximizes the expected future accuracy of the tracking while at the same time minimizing the total computational cost of tracking. In this work we measure the computational cost using the average CPU time necessary to execute our individual tracking algorithms, per one frame, denoted as $C(a_t)$. The immediate reward function $R(I_t, a_t)$ is thus devised where α controls the tradeoff between accuracy and efficiency.

$$R(I_t, a_t) = (1 - \alpha) \cdot \sqrt{(\hat{X}(I_t, a_t) - X_t)^2} - \alpha \cdot C(a_t), \quad (1)$$

where X_t is an estimate of the true target location, which can be typically obtained from labeled instances.

Temporal difference learning [11], $TD(\lambda)$, is used for estimating the optimal policy which aims to discover which actions maximize the total expected reward given a particular state of video and tracking s . This involves



(a) Youtube dataset - 161



(b) Lecture dataset - 1clp1

Figure 1. Tracking results. RL tracker in red, adaptive with optical flow in blue, Haar tracker in green, and predictive tracker in gray

the agent learning $Q(s_i, a_j) = E[R|s, \pi, a_t]$ which estimates the expected reward for executing action a_j at state s_i disregarding the consequence of the action. At every time step the expected reward is updated based on the new observed reward. The δ parameter is used to set the learning rate while λ specifies how much weight past trials have.

$$Q(s_t, a_t) = Q(s_t, a_t) + \delta[r_{t+1} + \lambda \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (2)$$

3.2 Tracking

Given the policy π , choosing the actions which always maximizes the total reward is not the best method for tracking in realistic scenarios. This is because there is no assurance that given the summary and potentially suboptimal state s_t representation, the action a_t will produce consistent rewards. For this reason we use stochastic action model where given a state s_t , we sample an action a_t according to a distribution proportional to $e^{Q(s_t, a_t)/\beta}$, where β is the scaling constant.

4 Face trackers

Three distinct face trackers are employed in this paper. None of the trackers individually are able to perform robustly in real time as our results show.

4.1 Adaptive face tracker

To be able to adapt to different poses and lighting conditions the Incremental Visual Tracker (IVT) is used [9]. Samples are generated by a Gaussian filter $G(\mu, \sigma^2)$ where the expected value is $\mu = x_{t-1} + c_b \cdot \hat{F}_t$ and the variance is $\sigma^2 = c_c \cdot \hat{F}_t$. The number of samples generated, $c_d \cdot \hat{F}$, is used to match the size of

the search space specified by the variance. Each sample must be evaluated in terms of quality in relation to the energy $E(I_t)$. At time $t = 1$ the energy fixed to be $E(I_1) = I_1$ otherwise $E(I_t) = d(I_t, I_{t-1})$ where $d(\cdot, \cdot)$ is a distance measure in the image space.

4.2 Haar tracker

While the adaptive face tracker offers good tracking due in contexts where the target appearance varies, it is inevitably susceptible to drift. To correct the adaptive tracker and to increase accuracy we use a Haar feature-based cascade detector first introduced in [13].

4.3 Predictive tracker

The adaptive tracker and Haar detector are very robust, but also very CPU intensive. To lessen the CPU load a simple predictive, optical flow based, face tracker is employed. Since the estimation comes from the optical flow, repeated use of this tracker causes for a very quick decline in accuracy. However, if used sparingly it offers a significant lessening of CPU load without any noticeable decline in accuracy. Given the previous bounding box location x_{t-1} and the current estimate of the local optical flow $[F_u, F_v]$, $c_{x,t} = c_{x,t-1} + c_d \cdot \bar{F}_u$ and $c_{y,t} = c_{y,t-1} + c_d \cdot \bar{F}_v$.

5 Results

To evaluate the independent trackers and RL based tracker discussed two datasets were used: the Youtube dataset[6] which contains 1500 clips with different resolutions, lighting and poses, and the Lecture dataset which contains 550 clips comprised of randomly selected videos from VideoLectures.net that offers higher resolution and longer video lengths.¹ A single policy π (with $\delta = 10^{-4}$ and $\lambda = .2$) for the RL tracker was created using a subset of the Youtube videos and then used throughout all other datasets without any modifications. For the adaptive tracker and predictive tracker $c_b = 1$, $c_c = [2, 2]$, and $c_d = 900$. The Haar tracker used the default parameters supplied by OpenCV. The parameters shown were not in either learning or tracking phases.

Since we are interested maximizing efficiency and accuracy, two measures were used to evaluate how well each tracker performed in the various datasets. First was the the number of frames per second that the tracker was able to process, this was calculated using a windowed

running average of 15 frames. Second, to measure accuracy, the manhattan distance between the tracker’s result and the ground truth is used to determine the average error. The ground truth is estimated by averaging the bounding boxes of all the individual face detectors used in the RL tracker. The adaptive, Haar tracker, and predictive tracker were used individually to set a benchmark for comparison. The more CPU intensive IVT tracker that uses constraints[6] was also used for comparison.

None of the individual face trackers discussed performed either very accurately or efficiently. Both adaptive based trackers performed poorly due to drifting. Optical flow did improve significantly the performance and accuracy in the Youtube dataset however due to the greater resolution of the Lecture videos the results were not as significant. The Haar tracker performed very well in the Youtube dataset, mostly because of featureless backgrounds found in the videos which reduced the overall number of false positives. The Lecture dataset which often had detailed backgrounds increased the number of false positives and thus increased the average error. The predictive tracker surprisingly did not perform much worse than the adaptive with constraints tracker. This once again indicates that the inclusion of optical flow to constrain trackers is a viable method for increasing efficiency and moderately increasing accuracy.

Figure 2. Comparison of different face tracking methods and the Manhattan pixel distance from the ground truth

Tracker	FPS	Mean	Std. Dev.
Adaptive + Constraints [6]	.5	38	7
Adaptive + Optical Flow	3.8	31	6
Haar Detector	3.7	17	3
Predictive	38	35	6
RL Tracker	35	5	2
Random Tracker	32	7	3

(a) Youtube dataset

Tracker	FPS	Mean	Std. Dev.
Adaptive + Constraints [6]	.4	23	18
Adaptive + Optical Flow	3.7	19	27
Haar Detector	3.7	10	11
Predictive	38	30	18
RL Tracker	36	8	2
Random Tracker	32	11	4

(b) Lecture dataset

For training the RL tracker 750 randomly selected clips from the Youtube dataset were used. This policy was then used for tracking in the remainder of the Youtube dataset, Lecture dataset, and Dudek video [9].

¹Both the Youtube and Lecture datasets are publicly available for download <http://seqam.rutgers.edu>

To compare the optimal policy for our proposed RL tracker we also included a random tracker which selected actions randomly. The random tracker performed very well in both the Youtube and Lecture datasets. This can be attributed to the fact that the Haar tracker was used about one third of the time in conjunction with the adaptive method which removed any false negatives. The RL tracker was unable to drastically outperform the random tracker in accuracy, however the improvement was observed on all datasets in efficiency. Both random and RL tracker did manage to reach near real-time tracking with very high accuracy.

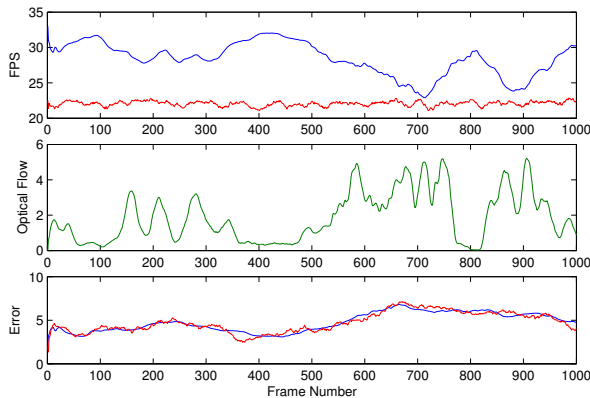


Figure 3. Analysis during the Dudek video sequence. RL tracker represented in blue, random tracker in red

We finally examined the performance of the RL approach on the benchmark Dudek video. As shown in Fig. 3, one sees that the FPS decreased in areas with greater movement to attempt to increase accuracy in tracking. While the average error did increase when movement also increased, the change is negligible given the high resolution of the video (720×489 pixels). Unlike the random tracker that maintained a constant rate of 20fps, the RL tracker was able to adapt without any noticeable loss of accuracy. Thus while additional components could be added to the state space for further gains in efficiency, optical flow proved more than sufficient.

6 Conclusion

We introduced a new reinforcement learning based tracker which combines several independent trackers into one robust, computationally efficient tracker. This approach allows for only one tracker to be used at a time based on current conditions of the video sequence allowing the tracker to maximize efficiency and accuracy.

While a random combination of trackers may produce high rates of accuracy, the trained RL based tracker was shown to jointly improve the efficiency and the tracking accuracy, leading to a viable tracking solution.

References

- [1] K. H. An, D. H. Yoo, S. U. Jung, and M. J. Chung. Robust multi-view face tracking. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 1905–1910, Aug. 2005.
- [2] W. Chuan-xu and L. Zuo-yong. A new face tracking algorithm based on local binary pattern and skin color information. In *Computer Science and Computational Technology, 2008. ISCSCT '08, 2008*.
- [3] P. Gejguš and M. Šperka. Face tracking in color video sequences. In *Proceedings of the 19th spring conference on Computer graphics*, pages 245–249, New York, NY, USA, 2003. ACM.
- [4] B. K. P. Horn and B. G. Schunck. Determining optical flow. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1980.
- [5] C. Huang, H. Ai, Y. Li, and S. Lao. High-performance rotation invariant multiview face detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(4):671–686, 2007.
- [6] M. Kim, S. Kumar, V. Pavlovic, and H. Rowley. Face tracking and recognition with visual constraints in real-world videos. In *CVPR08*, pages 1–8, 2008.
- [7] H.-S. Lee and D. Kim. Robust face tracking by integration of two separate trackers: Skin color and facial shape. *Pattern Recogn.*, 40(11):3225–3235, 2007.
- [8] C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Proceedings of the Sixth International Conference on Computer Vision*, pages 555–, Washington, DC, USA, 1998. IEEE Computer Society.
- [9] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1):125–141, May 2008.
- [10] S. Stalder, H. Grabner, and L. V. Gool. Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In *Proceedings ICCV09 WS on On-line Learning for Computer Vision*, 2009.
- [11] R. S. Sutton. Learning to predict by the methods of temporal differences. In *MACHINE LEARNING*, pages 9–44. Kluwer Academic Publishers, 1988.
- [12] V. Vilaplana and D. Varas. Face tracking using a region-based mean-shift algorithm with adaptive object and background models. *Image Analysis for Multimedia Interactive Services, International Workshop on*, 0:9–12, 2009.
- [13] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:511–I–518 vol.1, 2001.