Object Tracking Using Grayscale Appearance Models and Swarm Based Particle Filter

Bogdan Kwolek

Rzeszów University of Technology, W. Pola 2, Rzeszów, Poland
bkwolek@prz.rzeszow.pl

Abstract. We propose a hybrid tracking algorithm consisting of two trackers built on grayscale appearance models. In a first tracker we employ an object template that consists of several grayscale image patches. Every patch votes for the possible positions of the object undergoing tracking. A grayscale appearance model that is learned on-line is used in a supplementing tracker. A particle swarm optimization algorithm is utilized to shift particles toward more promising regions in the probability density function. Experimental results show that the hybrid tracker outperforms each of the trackers.

Keywords: Natural computation, hybrid systems, particle swarm optimization.

1 Introduction

Natural computation refers to computational systems that use ideas and get inspiration from natural systems, including biological, sociological, ecological and physical systems [1]. The aim of such research is to develop methods for dealing with large, complex, and dynamic problems. Computing inspired by nature takes nature as inspiration for the development of more robust and effective techniques. This often leads to combination of natural patterns and behaviors, and the use of visual systems to gather the knowledge. The human visual system enables us to perform tasks such as object recognition and categorization. Understanding the internal processes that lead to such perceptual capabilities is one of aims of this emerging discipline.

The work [2] presents arguments for the hypothesis that the human visual system consists of a number of interacting but autonomous systems that process different modalities and complement each other. Such subsystems can serve mutually in process of learning and bootstrapping the object representations. Cognitive science states that complex entities are perceived as composition of simple elements. Objects are represented through such components and the relations between them [3].

Visual tracking is one of the central problems and has received considerable attention in past years. The goal of tracking is to automatically find the same object in adjacent frames in a video sequence. Despite attempts to make visual tracking resistant to loosing the object undergoing tracking, most currently available algorithms inevitably fails under large visual perturbations including rapid unexpected motions, changes in ambient illumination, and occlusions. In this work we present a hybrid tracker that learns on-line within a co-training framework. To improve the tracking efficiency and effectiveness we employ particle swarm optimization and multi-part object representation.

2 Swarm Based Particle Filtering

2.1 Particle Filtering

Particle filtering [4] is recursive process in which particles are repeatedly selected, moved forward according to a probabilistic motion model that is dispersed by an additive random noise component, evaluated against the observation model, and resampled according to their weights to avoid degeneracy. The key idea underlying particle filtering is to approximate a probability distribution by a weighted particle set $S=\{(\mathbf{s}_n,\pi_n)\mid n=1,\ldots,N\}$. In such representation each particle $\mathbf{s}$ stands for a hypothetical state of the object, whereas the corresponding non-negative weight π represents the sampling probability. The weights are normalized such that $\sum_{n=1}^{N}\pi_n=1$. Two important components of each particle filter (PF) are motion model $p(\mathbf{z}_t\mid\mathbf{z}_{t-1})$ describing the state propagation and observation model $p(\mathbf{y}_t\mid\mathbf{z}_t)$ describing the likelihood that the state $\mathbf{z}_t$ causes the observation $\mathbf{y}_t$. The evolution of the sample set takes place by drawing new samples from a suitably chosen proposal distribution, which may depend on the old state and the new measurements, i.e. $\mathbf{z}_{n,t}\sim q(\mathbf{z}_{n,t}\mid\mathbf{z}_{n,t-1},\mathbf{y}_t)$, and then propagating each sample according to probabilistic motion model of the target. To give a particle representation of the posterior density the particles are set to $\pi_{n,t}\propto p(\mathbf{y}_t\mid\mathbf{z}_{n,t})p(\mathbf{z}_{n,t}\mid\mathbf{z}_{n,t-1})/q(\mathbf{z}_{n,t}\mid\mathbf{z}_{n,t-1},\mathbf{y}_t)$. The particles should be re-sampled according to their weights to avoid degeneracy.

The particle filter converges to the optimal filter in some sense as the number of particles grows to infinity. The most important property of the particle filter is its ability to handle complex, non-Gaussian and multimodal posterior distributions. However, the number of particles required to adequately approximate the conditional density grows exponentially with the dimensionality of the state space. This poses practical difficulties in applications such as articulated body tracking [5]. In such applications, weakness of the particle filter consists in that the particles can not cluster around the true state of the object and instead they can migrate towards local maximas in the posterior distribution. If particles are too diffused the track of the object can be lost. When the observation likelihood lies in the tail of prior distribution, most of the particles will become insignificant weights. If the system model is inaccurate, the prediction based on the system model may not be a good one. Predictions of poor quality can also be caused by the simulation in the particle filtering, when the system noise is large and the number of particles is not sufficient. In order to cope with the mentioned effects several improvements to basic algorithm have been proposed, among others solutions combining extended Kalman filter/unscented Kalman filter with generic particle filter [4]. Such filters incorporate the current observation to generate the better importance density than the generic particle filter that uses the prior as the importance density.

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a stochastic, population-based evolutionary algorithm for solving nonlinear, multimodal optimization problems [6]. This optimization technique, which is based on swarm intelligence has been inspired by social behavior of fish schooling or bird flocking. PSO shares several similarities with evolutionary computation techniques. The swarm is initialized with a population of individuals representing random solutions and then it searches for optima by updating particle values. The particles iteratively evaluate the fitness of the candidate solutions and remember the locations $\hat{\mathbf{x}}_i$, where they achieved their best fit so far. Another important value for the swarm is the location $\hat{\mathbf{g}}$ corresponding to the best fitness, which was obtained so far by any particle. Therefore, while exploring the hyperspace spanned by the possible parameters the particles employ reasoning capabilities about their own best location and the knowledge of the global best one. Through changing the velocity of each particle toward best fit $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{g}}$ locations, the particle swarm finds $\hat{\mathbf{x}}^* \subseteq \mathcal{X} \subseteq \mathrm{R}^m$ such that $\hat{\mathbf{x}}^* = \arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) = \{\mathbf{x}^* \in \mathcal{X} : f(\mathbf{x}^*) \le f(\mathbf{x})\ \forall \mathbf{x}\in \mathcal{X}\}$, and $f : \mathrm{R}^m \to R$ is a fitness function.

Let there be N particles, each with associated locations $\mathbf{x}_i \in \mathrm{R}^m$ and velocities $\mathbf{v}_i \in \mathrm{R}^m$. The optimization algorithm takes the following form:

- Initialize $\mathbf{x}_i$ and $\mathbf{v}_i$. Then $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$, $\hat{\mathbf{g}} = \arg\min_{\mathbf{x}_i} f(\mathbf{x}_i)$
- While stopping criteria are not satisfied
- For each particle
  - Update the particle locations: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
  - Update particle velocities: $\mathbf{v}_i \leftarrow \omega \mathbf{v}_i + c_1 \mathbf{r}_1 \circ (\hat{\mathbf{x}}_i - \mathbf{x}_i) + c_2 \mathbf{r}_2 \circ (\hat{\mathbf{g}} - \mathbf{x}_i)$
  - Update the local best values, if $f(\mathbf{x}_i) < f(\hat{\mathbf{x}}_i)$, then $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$
  - Update the global best, if $f(\mathbf{x}_i) < f(\hat{\mathbf{g}})$, then $\hat{\mathbf{g}} \leftarrow \mathbf{x}_i$

The symbol ω represents an inertial constant, $\circ$ stands for array multiply, c_1 and c_2 are constants, which balance the influence of the particle's local best and the global best, respectively, $\mathbf{r}_1$ and $\mathbf{r}_2$ are vectors of uniform random numbers between 0 and 1.

The above algorithm was inspired by animal behavior. A few simple rules result in complex action. It has been demonstrated that in many problems PSO achieves better results in a faster way in comparison with other methods. PSO has been successfully applied in wide range of applications. Another reason that PSO is attractive is that it needs a few parameters to adjust.

2.3 Swarm Based Particle Filter

PSO algorithm and PF employ different schemes to concentrate particles near best locations. PSO shifts particles towards the optimum by updating position and velocity of each particle using its best previous value and global best value of all particles. Particle filter utilizes general prediction-update framework in which probabilistic motion and observation models are used alternately during approximating the conditional density. The algorithm for swarm-based particle filter is as follows:

1. Initialization. Sample $\mathbf{z}_{1,0},\ldots,\mathbf{z}_{N,0}$ i.i.d from initial density p_0
2. Importance Sampling/Propagation. Sample $\mathbf{z}_{i,t}$ from $p(\mathbf{z}_t \mid \mathbf{z}_{i,t-1})$, $i = 1,\ldots,N$

```
3. Initialize PSO. Assign initial values $\mathbf{v}_i$, then $\mathbf{x}_i \leftarrow \mathbf{z}_{i,t}$, $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$, $\hat{\mathbf{g}} = \arg\min_{\mathbf{x}_i} f(\mathbf{x}_i)$
   3a. While not stop
         For each particle
           Update the particle locations: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
           Update particle velocities: $\mathbf{v}_i \leftarrow \omega \mathbf{v}_i + c_1 \mathbf{r}_1 \circ (\hat{\mathbf{x}}_i - \mathbf{x}_i) + c_2 \mathbf{r}_2 \circ (\hat{\mathbf{g}} - \mathbf{x}_i)$
           Update the local best values, if $f(\mathbf{x}_i) < f(\hat{\mathbf{x}}_i)$, then $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$
           Update the global best, if $f(\mathbf{x}_i) < f(\hat{\mathbf{g}})$, then $\hat{\mathbf{g}} \leftarrow \mathbf{x}_i$
   3b. $\mathbf{z}_{i,t} \leftarrow \hat{\mathbf{x}}_i$
4. Updating. Compute $\hat{p}(\mathbf{z}_t \mid \mathbf{y}_{1:t}) = \sum_{i=1}^{N} \pi_{i,t} \delta(\mathbf{z} - \mathbf{z}_{i,t})$ using normalized weighs:
      $\pi_{i,t} = p(\mathbf{y}_t \mid \mathbf{z}_{i,t}), \quad \pi_{i,t} = \pi_{i,t} / \sum_{j=1}^{N} \pi_{j,t}, \quad i = 1,\dots,N$
5. Resampling. Sample $\mathbf{z}_{1,t},\dots,\mathbf{z}_{N,t}$ i.i.d from $\hat{p}(\mathbf{z}_t \mid \mathbf{y}_{1:t})$
6. $t \leftarrow t+1$, go to step 2.
```

The algorithm was compared with generic particle filter on example from work [4]:

$$x_{k+1} = 1 + \sin(0.04\pi k) + \rho x_k + u_k, \; x_0 = 0,$$
$$y_k = \begin{cases} 0.2x_k^2 + v_k & k \le 30 \\ 0.5x_k - 2 + v_k & k > 30 \end{cases} \qquad \textbf{(1)}$$

where ρ is a system parameter. The system noise u_k follows a Gamma distribution $\Gamma(3,\theta)$, where θ is the scale parameter. The observation noise follows a Gaussian distribution $\mathcal{N}(0, 1.0e\text{-}5)$. We simulated 100 independent runs of length 60 with random initializations. To compare filters we calculated the root mean square (RMS) error of the estimated states for each run, and the means of the RMS of the 100 runs. Using the generic particle filter built on 200 and 100 particles, respectively, we obtained the following RMS: 0.07, 0.2, in time 4.4 and 2.2 sec., respectively. Using the swarm-based particle filter built on 50 particles and performing 3 and 5 iterations in PSO, respectively, we obtained the following RMS: 0.11, 0.07 in time 2.2 and 3.1 sec., respectively. From the above results we can observe that in time 2.2 sec. the generic particle filter produces estimates with RMS equal to 0.2, whereas the swarm based particle filter gives RMS equal to 0.11. The generic particle filter needs 4.4 sec. to yield estimates with RMS equal to 0.07, whereas the swarm-based particle filter takes 3.1 sec. The better results are due to swarm intelligence, which considers the distribution both from local and global perspective.

3 Appearance Based Person Tracking

3.1 Adaptive Appearance Models

The intensity model has been inspired by the *WSL* model [7]. The model consists of three components called wandering, stable and lost. During model learning each component votes according it its level of creditability. The model can adapt to slowly changing appearance and provides robustness to partial occlusions. It has been shown to yield reliable tracing of pedestrians, where a second AdaBoost-based tracker with a different failure mode was used to support the learning of the pedestrian's appearance [8].

The appearance model that is assigned to a single gray observation d_t consists of three components, namely, the *W*-component characterizing variations between two consecutive frames, *S*-component expressing stable component structures within the former observations, and *C*-component representing pixel values from the initial object template. Such a model exhibits the object appearances in frames up to time $t-1$. It is a mixture of Gaussians with centers $\{\mu_{i,t} \mid i = w,s,c\}$, their corresponding variances $\{\sigma^2_{i,t} \mid i = w,s,c\}$, and mixing probabilities $\mathbf{m}_t = \{m_{i,t} \mid i = w,s,c\}$. The mixture probability density for a new data d_t conditioned on the former observations is given by:

$$\begin{aligned} p(d_t \mid d_{t-1}, \mu_{s,t-1}, \sigma^2_{s,i-1}, \mathbf{m}_{t-1}) = m_{w,t-1} p_w(d_t \mid \mu_{w,t-1}, \sigma^2_w) + \\ m_{s,t-1} p_s(d_t \mid \mu_{s,t-1}, \sigma^2_{s,t-1}) + m_{c,t-1} p_c(d_t \mid \mu_{c,0}, \sigma^2_c). \end{aligned} \tag{2}$$

The mean $\mu_{w,t-1}$ in the wandering component is set to the observation d_{t-1} and the variance σ^2_w is fixed. The stable component expresses the appearance properties that are relatively stable over time. A Gaussian density function with slowly accommodated parameters $\mu_{s,t-1}$, $\sigma^2_{s,t-1}$ expresses the evolution of such temporally stable image observations. The fixed component accounts for data holding information from the initial object appearance. The mean value is the observation d_0 taken from the initial frame and the variance is fixed at σ^2_c. The model is learned on-line using the EM algorithm. Details of the learning algorithm can be found in [8].

A swarm particle filter build on the learned on-line appearance models has been constructed and then used co track people in surveillance videos. The state transition model is a random walk in which a new predicted state is composed through adding to the previous state a zero mean Gaussian noise with a covariance Σ. A more complex dynamic model can be employed if relevant. The observation likelihood is calculated according to the following formula:

$$p(\mathbf{y}_t \mid \mathbf{z}_t) = \prod_{j=1}^{M} \sum_{i=w,s,c} \frac{m_{i,t}}{\sqrt{2\pi\sigma^2_{i,t}(j)}} \exp\left[-\frac{d_t(j) - \mu_{i,t}(j)}{2\sigma^2_{i,t}(j)} \right] \tag{3}$$

where d_t denotes the value of gray pixel, M is the number of pixels in the appearance model.

3.2 Multi-Patch Object Tracking

In work [9] it has been demonstrated that multi-part object representation leads to better tracking. The authors proposed a method consisting in dividing the object to be tracked into non-overlapping rectangular regions and then computing in each sub-region a color histogram. The observation model has been constructed under assumption that image data in different sub-regions are independent. The object likelihood was proportional to the exponential of negative sum of squared distances between histograms. In [10], rectangle sub-regions that are defined within the extent of the object to be tracked are employed to calculate the averaged color of pixels. The object tracking is achieved through an exhaustive search in a confidence region. An occlusion model has been developed to discriminate between good and spurious measurements.

Cognitive science states that complex entities are perceived as composition of simple elements. Objects are represented through such components and the relations between them [3]. One of the disadvantages of color histograms is the loss of spatial information. To incorporate such information in the object representation we divide the object template into adjoining cells regularly spaced within the extent of the target to be tracked. We compute histograms within such regularly spaced patches using a fast method that has been proposed in [11]. Given the estimated position in the previous frame and the histograms within object at the estimated position we employ the chi-square test between such histograms and histograms extracted from cells within the template at the candidate position. The X^2 test is given by: $X^2 = \sum_i ((h_{e,i} - h_{c,i})^2 / (h_{e,i} + h_{c,i})^2)$, where $h_{e,i}$ and $h_{c,i}$ represent the number of entities in the i-th bin of the histograms, and a low value for X^2 indicates a good match. Such values are transformed into likelihoods through the usage of the exponential function. We seek for the object in the new frame within a search window, which center is located at the previous object position. At each candidate position we compare the corresponding histograms and the result is utilized to vote for the considered position. Every cell votes in its own map. Then we combine the votes in a relevance map, which encodes hypothesis where the target is located in the image. In order to cope with partial occlusions we employ a simple heuristics aiming at detecting outliers. If the difference between corresponding histograms is below a certain level, then in such case the score in the relevance map remains unchanged. The level for each cell is determined individually using the distances between histograms from the initial template and corresponding histograms from few first frames. Similar test is performed with respect to actual medians.

3.3 Nature Inspired Cooperative Tracking of the Object

Obtaining a collection for on-line unsupervised learning is a complex task. In work [2] it has been argued that the human visual system consists of a number of interacting but still autonomously operating subsystems that process different object representations. Within such a mechanism, subsystems can serve mutually in process of learning and bootstrapping of object representations. This motivated us to construct an object tracker consisting of two independent trackers with different failure modes, complementing each other and operating in co-training framework. The co-training approach has been utilized in work [12], where a tracker starts with a small training set and increases it by co-training of two classifiers, operating on different features. In work [8] a co-training mechanism supports boosting of features as well as unsupervised learning of adaptive appearance models.

In the multi-patch based object tracker an accommodation of the histograms over time takes only place if there is a significant overlap between the object templates, i.e. between the cell-based object template and the adaptive appearance-based template. The multi-patch object tracker takes the advantages of the collaborative tracker that keeps the object location far more precisely. Owing to this the multi-patch tracker is prevented from the template drift, in which the template gradually slips away from the object until the tracker is following something else entirely. The second tracker can fail in case of partial occlusions or when the target appearance undergoes major changes. The cells that are marked as possibly occluded indicate pixels, which are not employed in learning of the stable components.

4 Experiments

The tests were done on a sequence[1] of images 288 high and 384 wide. The tracker has been initialized in frame #700, see Fig. 1 that depicts some results obtained by multi-patch based tracker. Frames #888 - #939 show the tracking during temporal occlusions. We can see that in frame #929 the track has been temporally lost. The tracker recovered the track in frame #939. Frames #1554 and #1566 illustrate the tracking during another occlusion. The tracker failed in frame #1645. The results were achieved using four patches. The size of the search window was 15 x 15. Due to considerable jitter of the tracked window the histogram has not been adapted in the course of the tracking.

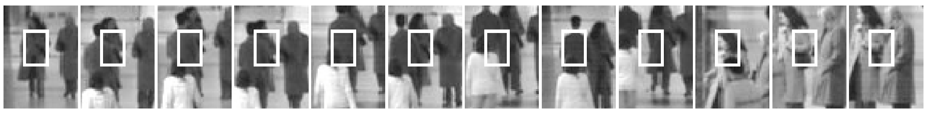

Fig. 1. Person tracking using multi-patch algorithm. Frames #700, #888, #889, #899, #900, #909, #928, #929, #939, #1554, #1566, #1645

Figure 2 depicts some results that were obtained using swarm-particle filter built on adaptive appearance models. The tracking has been done using only 20 particles. As we can observe the tracker keeps the specific region of the person far more precisely. However, it fails in frame #1566. The generic particle filter loses the person earlier. Additionally, the jitter of the tracking window in such a tracker is considerable.

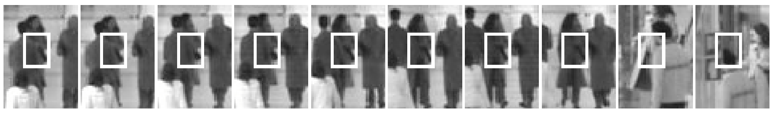

Fig. 2. Person tracking using swarm particle filter built on adaptive appearance models. Frames #888, #889, #899, #900, #909, #928, #929, #939, #1554, #1566

The hybrid tracker outperforms each of the trackers. In particular, the jitter of the window is comparable to the jitter in the swarm-particle filter based tracker. This allows us to carry out the adaptation of the histogram in the multi-patch based tracker. Both adaptation of the histogram and learning of the appearance models is done only in case of significant overlap between windows determined by each of the trackers. The

[1] Downloaded from site at: http://groups.inf.ed.ac.uk/vision/CAVIAR/WalkByShop1cor

person has been tracked through the whole sequence, see Fig. 3. The deterministic, muli-patch based tracker is several times slower than the particle based tracker but it is more resistant to partial occlusions. The hybrid tracker takes the advantages of both trackers. In consequence it is able to precisely track the object with small window jitter, see frames #888 - #939 as well as to cope with occlusions, see frames #1554 and #1566.

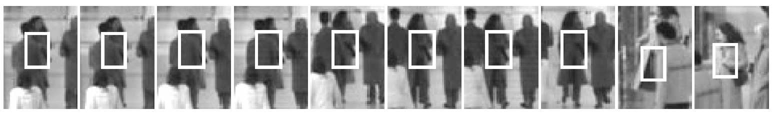

Fig. 3. The location of the tracking window determined by the hybrid tracker. Frames #888, #889, #899, #900, #909, #928, #929, #939, #1554, #1566

5 Conclusions

The main contribution of this work is a hybrid algorithm consisting of two trackers that have different failure modes and complement each other. This cooperative framework leads to better tracking. The main ingredient of the tracking algorithm is swarm-based particle filter that has been tested in simulation as well as on real video data. The multi-patch tracker acknowledged its great usefulness in tracking objects undergoing occlusions.

References

1. de Castro, L. N.: Fundamentals of Natural Computing: An Overview. Physics of Life Reviews, vol. 4, no. 1, pp. 1-36 (2007)
2. Zeki, S.: Localization and Globalization in Conscious Vision. Annual Review Neuroscience, 24, pp. 57-86 (2001)
3. Ommer, B., Buhmann, J. M.: Learning Compositional Categorization Models. In: European Conf. on Computer Vision, pp. III:316-329 (2006)
4. van der Merve, R., de Freitas, N., Doucet, A., Wan, E.: The Unscented Particle Filter. Advances in Neural Information Processing Systems, 13, pp. 584-590 (2001)
5. Schmidt, J., Fritsch, J., Kwolek, B.: Kernel Particle Filter for Real-Time 3D Body Tracking in Monocular Color Images. In: IEEE Int. Conf. on Face and Gesture Rec., pp. 567-572 (2006)
6. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proc. IEEE Int. Conf. on Neural Networks, pp. 1942-1948. IEEE Press, Piscataway, NJ, USA (1995)
7. Jepson, A. D., Fleet, D. J., El-Maraghi, T.: Robust Online Appearance Models for Visual Tracking. IEEE Trans. on PAMI, vol. 25, no. 10, pp. 1296–1311 (2003)
8. Kwolek, B.: Learning-Based Object Tracking Using Boosted Features and Appearance Adaptive Models. In: Int. Conf. on ACIVS, LNCS, vol. 4678, pp. 144–155. Springer (2007)
9. Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-Based Probabilistic Tracking. In: European Conf. on Computer Vision, LNCS, vol. 2350, pp. 661-675. Springer (2002)
10. Fieguth, P., Terzopoulos, D.: Color-Based Tracking of Heads and Other Mobile Objects at Video Frame Rates. In: Proc. IEEE Conf. on Comp. Vision and Patt. Rec., pp. 21-27 (1997)
11. Porikli, F.: Integral Histogram: A Fast Way to Extract Histogram in Cartesian Spaces. In: IEEE Computer Society Conf. on Pattern Rec. and Computer Vision, pp. 829-836 (2005)
12. Levin, A., Viola, P., Freund, Y.: Unsupervised Improvement of Visual Detectors Using Co-Training. In: Proc. Int. Conf. on Comp. Vision, pp. 626-633 (2004)