FUSION OF MOTION SEGMENTATION WITH ONLINE ADAPTIVE NEURAL CLASSIFIER FOR ROBUST TRACKING

Sławomir Bąk
Institute of Computing Science, Poznan University of Technology, ul. Piotrowo 2, Poznan, Poland

Sundaram Suresh, Francois Brémond, Monique Thonnat
INRIA Sophia Antipolis, PULSAR group
2004, route des Lucioles, BP93
06902 Sophia Antipolis Cedex, France

Keywords: Object tracking, Neural network, Gaussian activation function, Feature extraction, On-line learning, Motion segmentation, Reliability classification.

Abstract: This paper presents a method to fuse the information from motion segmentation with online adaptive neural classifier for robust object tracking. The motion segmentation with object classification identify new objects present in the video sequence. This information is used to initialize the online adaptive neural classifier which is learned to differentiate the object from its local background. The neural classifier can adapt to illumination variations and changes in appearance. Initialized objects are tracked in following frames using the fusion of their neural classifiers with the feedback from the motion segmentation. Fusion is used to avoid drifting problems due to similar appearance in the local background region. We demonstrate the approach in several experiments using benchmark video sequences with different level of complexity.

1 INTRODUCTION

Visual tracking of objects in complex environments is one of the most challenging problem in the machine vision. Tracking algorithms are developed to faithfully determine the movement of image region in each frame that matches with the given object. In general, developing a robust tracker is a challenging problem due to dynamic change in appearance, background, scale, rapid illumination variation, and occlusion.

For the past two decades, many algorithms with different frameworks have been developed for object tracking. Among various algorithms, detect-then-track, appearance-based and learning-based algorithms are widely used in the literature. In detect-then-track approaches (Stauffer and Brady, 2000), objects are detected and tracked effectively in real-time using the frame differencing or subtracting adaptively estimated background from the current frame. Appearance-based approaches (Cootes et al., 2001) create an object model from the first frame and incrementally follow the model in the subsequent frames. Learning-based algorithms use pattern recognition algorithms to learn the target objects in order to search them in an image sequence (Avidan, 2007). A complete review on different algorithms in object tracking can be found in (Yilmaz et al., 2006).

In this work the problem of object tracking in video sequence is converted into a binary classification problem and a discriminative model is developed to differentiate the object from the background. Similarly, in (Collins et al., 2005) an adaptive on-line feature selection mechanism is used to select best tracker which can efficiently discriminate the object and background in the current situation from the given set of tracker. In (Nummiaro et al., 2003) an adaptive target model for efficient mean-shift tracking is presented. The histogram based target model is updated linearly when the confidence level exceeds threshold value. Nevertheless, this adaptive tracker fails when there is an abrupt change in the appearance or illumination. In (Jepson et al., 2003), short-term and long term image descriptors are constantly updated and re-weighted using online-EM to handle abrupt change in object appearance. The above mentioned approaches require proper initialization of the object

BÄĔk S., Suresh S., BrÃľmond F. and Thonnat M.
FUSION OF MOTION SEGMENTATION WITH ONLINE ADAPTIVE NEURAL CLASSIFIER FOR ROBUST TRACKING.
DOI: 10.5220/0001769604100416
In *Proceedings of the Fourth International Conference on Computer Vision Theory and Applications (VISIGRAPP 2009)*, page
ISBN: 978-989-8111-69-2

bounding box in the first frame and does not handle change in scale. In (Williams et al., 2005) the use of statistical learning algorithm for object localization is extended. A displacement expert is build to estimate displacement from the target region. A fully probabilistic relevance vector machine (RVM) is used to generate observations with Gaussian distributions which can be fused over time. Recently, in (Avidan, 2007), the appearance of object and its background are modeled using an ensemble of classifiers. Each one of the classifiers is trained to identify object and background classes. Then a strong classifier obtained by Ada-boost is used to locate the object in the next frame. All these approaches require proper initialization and are limited to single object.

In this paper, the information from motion segmentation is fused with online adaptive neural classifier to handle aforementioned issues. The online learning sequential classifier is used to differentiate the object region from non-object region. The objects present in the video sequence are detected by using the motion segmentation information combined with the object classification. Then, the neural classifier is initialized to differentiate the object from its local background. In following frames the motion segmentation helps to avoid drifting problems due to similar appearance in the local background region.

The outline of the rest of the paper is the following. In Section 2 the used platform (SUP) and the proposed fusion of motion and neural tracking algorithm are presented. Section 3 illustrates experimental results and Section 4 contains some concluding remarks.

2 SUP PLATFORM

Scene Understanding Platform is developed at the research group PULSAR at INRIA, Sophia Antipolis. It is an environment for combining algorithms for video analysis which allows to flexibly combine and exchange various techniques at the different stages of video understanding process. Furthermore, SUP is oriented to help developers describing their own scenarios and building systems capable of monitoring behaviors, dedicated to specific applications. SUP takes as an input video stream, a geometric description of the unoccupied scene, a geometric description of models appearing on the scene and a set of behaviors of interest specified by experts of the application domain. Afterwards, it is possible to process the mentioned data by combining different modules including algorithms. In our approach the object classification module performs the initialization of tracked objects.

Object classification (Zúñiga et al., 2006) performs the initialization for object tracker. Here, objects are modeled independently from the camera position and object orientation. A simple parallelepiped model is used for 3D object modeling, which is estimated using set of 2D moving regions (obtained in a segmentation phase). These moving regions (also called blobs) are merged to improve the classification performance by assembling 2D moving regions with better 3D model probability. The merged blobs with their associated class label are called *mobiles*. Each mobile is enclosed by a 2D bounding box, which is used for object initialization and also in adaptation phase of the tracker.

2.1 Neural Object Tracker

The neural tracker aims at distinguish foreground objects from the background. In Fig. 1 the general model of the neural tracker is presented. First, mobiles generated using motion information are used for initialization. If the 2D bounding box of a mobile relates to an area where none target exists, a new target is initialized. In this case the target goes through three steps: feature extraction, object/background separation and neural network training. Otherwise, if a target exists in the corresponding area, the tracking algorithm is applied. For each target the new location is determined using the localization step. Then, in the adaptation phase, the dimension of the target is corrected and the neural network is adapted using features computed in the new target region.

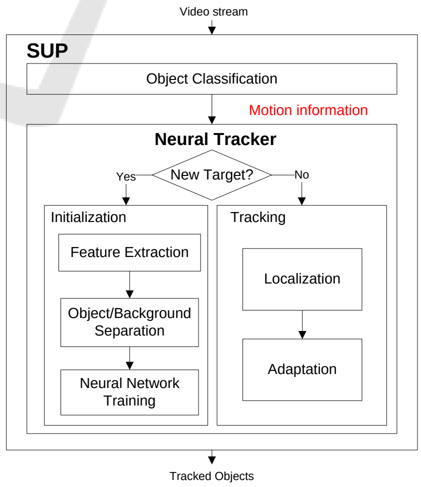

Figure 1: Model of the neural tracking system.

Target Initialization is based on the motion information obtained from the object classification. This information is given in a form of coordinates of 2D bounding boxes of mobiles with labels assigned in the classification phase. Features are extracted for each 2D bounding box together with its local background. The label of the target is used to initialize some constants in the neural network algorithm (e.g. maximal number of neurons in the neural network or size of extracted regions in the feature extraction step).

Feature Extraction returns vectors, which contain values representing features from target region (e.g. RGB values). The neural network can use different types of features such as simple color features, region based features, local gradients or texture. The feature vector is computed for each pixel in case of small objects (max. 1000 pixels). Otherwise we extract region based features like Region based Color Moments (RCM) to speed up the approach. In this case, the target is separated into rectangular regions. For each region the feature vector is calculated using mean values of pixels belonging to the region. These vectors are used for object/background separation.

Object/Background Separation is used for the training step. Tracking an object will be efficient if we can separate the object region from the background accurately. The problem of tracking is converted into a binary classification problem and solved using neural networks. The online learning neural tracker presented in this paper is a generic approach, which is trained to separate the object region from the background. We represent the target as a 2D bounding box. The features inside the target can be labeled as 'object class' and features outside target region can be labeled as 'background class'. Some of the features inside the target region will be similar to the background. Such features have to be labeled as 'background class' for better classification performance. For this purpose, we use a feature-based object-background separation technique. Here, the probability density function of the feature in the target region and its local background is obtained to find the log-likelihood ratio of the sample belonging to the 'object class'. The log-likelihood ratio L_i is obtained as

$$L_i = \log \frac{\max\{h_o(i), \varepsilon\}}{\max\{h_b(i), \varepsilon\}} \tag{1}$$

where $h_o(i)$ and $h_b(i)$ are the probabilities of ith sample belonging to the object and the background, respectively; ε is a small non-zero value to avoid numerical instability. The class label is determined using the threshold τ_o on L_i. Typical value of τ_o is set at 0.65. More details can be found in (Bąk et al., 2008).

Online Neural Classifier proposed in this paper adapts the temporal change in appearance of the object/background model. The basic building block of the neural classifier is the Radial Basic Function Network (RBFN). The RBFN is trained to recognize the object region and its local background by estimating the posterior probability of the feature vector belonging to the object or the background. The RBFN architecture consists of an input layer, a hidden layer and an output layer. The inter-connection weights only exist between the hidden and the output layer. Gaussians units are used in the hidden layer as activation functions because of their localization properties. Generally, the output of the RBFN classifier with K hidden neurons has the following form:

$$\hat{y}_i = f(\mathbf{X_i}) = \sum_{j=1}^{K} \alpha_j exp\left(\frac{-\|\mathbf{X_i} - \mu_\mathbf{j}\|}{2\sigma_j^2}\right) \tag{2}$$

where μ_j and σ_j is the center and width of the j^{th} neuron, respectively; α_j is the weight connection between the j^{th} neuron and the output, $\alpha_j \in \Re$; $\mathbf{X_i}$ is the feature vector of i^{th} sample.

RBFN classifier involves the allocation of new Gaussian hidden neurons, pruning the neurons and also adapting the neuron parameters. The RBFN begins with no hidden neurons (i.e., the network output is zero for the first sample). While the observation data are received sequentially, the network starts growing and shrinking by using some of them to make decision based on certain criterions. The input of the neural network are vectors obtained in the feature extraction phase. Risk sensitive hinge loss function proposed in (Suresh et al., 2008a) is used to compute the learning error. Risk loss function helps in estimating the posterior probability accurately. Growing criterion is based on the error value (e_i) and a distance between the nearest neuron and the training sample. If the distance between the new training sample and the nearest neuron of the same class exceed a threshold a new hidden neuron will be added to the RBF classifier. Its parameters are set as follows: $\alpha_{K+1} = e_i$, $\mu_{K+1} = \mathbf{X}_i$, and $\sigma_{K+1} = \kappa\|\mathbf{X}_i - \mu_{nr}\|$. Here, κ is a positive constant which controls the overlap between the hidden neurons. The value of κ is 0.8 in our experiments. When the new training sample does not satisfy the criterion for adding a new hidden neuron, the network parameters are adapted using a Decoupled Extended Kalman Filter (DEKF). More details on learning algorithm can be found in (Suresh et al., 2008b). In order to maintain a compact network and remove a non-performing neuron, a pruning strategy is incorporated in the algorithm. Pruning of neurons ensures that the neurons that have been added in the past and have not contributed significantly to the network performance are removed from the network. This strategy consists in removing neurons which contributions is less than a threshold for M consecutive observations. The parameter M depends on the number of samples in each class.

Localization phase is needed to calculate a displacement of the target. It is achieved by computing the distance between weighted centers of the target in current and previous frame. Let c^i be the weighted object center obtained from the corresponding probability map $\hat{p}^i$ in i^{th} frame. Then, using the neural network and the location of the target from i^{th} frame we calculate the posterior probability map $\hat{p}^{i+1}$ in $(i+1)^{th}$ frame. Here we assume that there is an overlap between the target region in the subsequent frames. Otherwise, the tracker fails to detect the correct center of the object. It can happen considering fast moving objects. In this case the tracker should use the motion information to compare generated 2D bounding boxes with already existing targets. Assuming an overlap, the new center is estimated as

$$c^{i+1} = \left\lfloor \frac{\sum_j I_j^{i+1} \hat{p}_j^{i+1} \hat{p}_j^i}{\sum_j \hat{p}_j^{i+1} \hat{p}_j^i} \right\rfloor \tag{3}$$

where I^{i+1} is the location of the j^{th} feature. Next, the 2D bounding box is moved by the vector computed using both centers. The probability map is updated considering the new location of the target and the procedure is repeated until the distance between both centers (Δc) can be neglected. More details on the localization can be found in (Bąk et al., 2008)

Adaptation is responsible for the correction of the size of the object bounding box and the adaptation of the neural network to handle changes in appearance. This procedure is based on the probability map obtained from the neural classifier and the motion information generated by the object classification module.

Firstly, we use the probability map to find the current object dimension. For this purpose, we define the class label map based on the estimated posterior probability. Then, the morphological operations are applied to remove noises. After noise removal, the map is used for size determination based on a calculation of continuous regions in width and height dimension. Secondly, the object classification generates mobiles which are compared with existing ones. If a mobile relates to an area where none target exists then a new one is initialized. Otherwise, the mobile is used to adapt the target located in this area. The information coming from motion segmentation is fused with the estimated posterior probability map generated by the neural classifier. The pseudo code of the information fusion at adaptation level is given in Listing 1.

Listing 1 : Pseudo code for information fusion.

```
Adaptation(frame_i)
begin
  if no mobile /* classification fails */
   do not adapt the NN and the target size
  else
   [c^t,s^t]_i - apply morphological operations
   [c^m,s^m]_i - mobile from the classification
  compare [c^t,s^t]_i with [c^t,s^t]_{i-1} and
          [c^m,s^m]_i with [c^m,s^m]_{i-1}
  if( Δs^t > 40% and Δs^m > 40%)
   do not adapt the NN and the target size
  else
 /*fusion of the NN and motion segmentation*/
   compare [c^t,s^t]_i with [c^m,s^m]_i
   if( Δs_i > 10% )
    compare [c^t,s^t]_i with the history of mobiles
    if( min( Δs ) > 20% )
        [c^t,s^t]_i := [c^m,s^m]_i
        update the history of mobiles
   adapt the NN using [c^t,s^t]_i
   [c^new,s^new]_i - apply morphological operations
   compare [c^new,s^new]_i with [c^t,s^t]_i
   if( Δs > 10% )/* the NN fails */
      retrain the NN
end.
```

Listing 2 : Occlusion events.

```
OcclusionEvent(Target A)
   Constraint{A.density < 10%}
   Alarm("Occlusion")
   State_Assignment{A.state = OCCLUDED}
DynamicOcclusionEvent(Target A, Target B)
   Constraint{Crossing.Area > 60% of A.Area}
   Alarm("Dynamic Occlusion")
   State_Assignment{
     if(A.density < B.density)
        A.state = DYNAMIC_OCCLUDED}
```

The algorithm starts by checking if there is a mobile in the related target area. We do not adapt the neural network and the target size in the case of absence of mobiles. If the mobile exists we try to fuse mobile information with output of the neural classifier. In order to achieve this we compare both information to make decision about the true dimension of the object. First, morphological operations are applied to estimate the current target size. Let us assume that the center and the size at frame i are represented by $[c^t, s^t]_i$ as the result of morphological operations and $[c^m, s^m]_i$ as the mobile. Next, both target and mobile size are compared with target and mobile size at previous frame, respectively. If the size change (Δs) in both cases is greater that 40% we do not allow to adapt the target size and also the parameters of the neural network are not modified. Here, we try to avoid abrupt changes obtained from the object classification which happens in a case of shadows, crowds or another disturbances coming from the motion information. Then, the fusion rules is used to decide whether it is necessary to correct the dimension and the localization of the target. We check the difference between the target and the mobile size (Δs_i) in current frame. If it is greater than 10% we compare the target size with the history of mobiles (max 5 previous mobiles). If the history suggest that the mobile size is invariant then we accept the new size and the new center. After establishing of the target dimension the neural network is trained using samples from the target and its local background region. The new probability map is computed and the new dimension of the target is determined. If the difference between new and previous size is greater than 10% we assume that the neural network fails. All neurons are removed and the network is retrained.

After the adaptation step, the tracker checks if there are any occlusions in the scene. Occlusion events are presented in Listing 2. We distinguish two types of occlusion. First, the static occlusion means that the moving target is occluded by static item. This decision is made only if the density value of the probability map decreases more than 10%. However, the dynamic occlusion is detected when the target crosses another target and the common area is greater than 60% of the area belonging to one of them. We assume that the first target is occluded if the density value is less than the density of second target.

3 EXPERIMENTAL RESULTS

We have tested many challenging video sequences to illustrate advantages and limitations of our tracker. The online learning phase enhances the ability to track under changing background and illumination conditions, changing in appearance and scale and improper initialization. First, the experiments were performed on the data obtained from the Gerhome laboratory which promotes research in the domain of activity monitoring and assisted living (Zouba et al., 2008). Next, we tested our approach on TREC Video Retrieval Evaluation (organized by NIST, TRECVID 2008) data obtained from Gatwick Airport surveillance system. Below we present two example sequences. More evaluations can be found in (Bąk et al., 2008).

Gerhome Video Sequence in Fig. 2 is presented. At first frame a target is initialized as a bounding box with a label which is obtained from the object classification module. During initialization for each new target the neural network is created and an identity is assigned. Next, this neural network is trained using features computed from the target and the local background region. At frame 67 we can observe important issues. The motion information for object initialization is not always correct which leads to track noise (as '2-PERSON') due to illumination change. Nevertheless if in subsequent frames the target is not confirmed by the motion information coming from the object classification, the target is assumed to be a noise and afterwards is removed. However a more important issue is that the tracked person is split into two targets ('0-PERSON' and '3-PERSON'), caused also by noise coming from motion information. We do not apply any merging algorithm for tracked targets because it is very difficult to decide whether few targets in fact represent one real object or several different objects. The neural network is not also helpful in that case because parts of a real object could have a completely different appearance model which prevents from merging such kind of targets. At frame 93 we can see '0-PERSON' marked as occluded target due to movement of object behind a cupboard door. At frame 133 we show that the neural tracker is able to capture true dimension of the object in following frames.

We also tested our approach on a long-term sequence. An elderly woman in her apartment (real world scene with occlusions and illumination changes) was tracked during 1 hour and 23 minutes (50.000 frames). A woman left for short periods of time the observed room (and came back) 7 times. The tracker was confused only 22 times (id switched).

TRECVID Video Sequence in Fig. 3 is presented. During this complex sequence many objects are crossing each other. It is shown that the neural tracker has the ability to manage with dynamic and static occlusions. The neural tracker is used to make decision which targets are occluded. For each target the probability map of the overlapping area is computed and the most probable one is chosen. The most probable target means the target which has the largest density value of the probability map. During dynamic occlusion localization stage does not use probability map to localize target but its history which is based on the target displacement. Also the adaptation process is suspended. The history contains the informa-

(a) Frame 1 (b) Frame 67 (c) Frame 93 (d) Frame 133

Figure 2: Tracking results from Gerhome laboratory. Person is split into two parts.

(a) Frame 1 (b) Frame 15 (c) Frame 52 (d) Frame 110

Figure 3: Tracking results for TREC Video Retrieval Evaluation data. Complex sequence.

tion about the velocity and the direction of the target motion represented by displacement vector. Using that information our tracker is able to approximate location of the occluded target. We resume using the neural network only if we confirm separated objects from the motion information. It is worthy noting that an appearance information coming from the neural network is not always enough to differentiate the object from the background accurately. Consequently, it leads us to fuse the neural classifier with the motion information.

At first frame, we can observe that two persons are grouped into single object (‘2-PERSON’ and ‘3-PERSON’). It is impossible to separate them due to close proximity and shadow. But, the effect of shadow was removed in subsequent frames using the posterior probability map from online learning tracker. We can also observe that the object ‘4-PERSON’ lost at frame 15 due to similarity with object ‘0-PERSON’ and occlusion with barrier. Later, the same object is reinitialized as a new object. From the result, we can observe that the proposed tracker is able to handle significant occlusion with multiple object and track them efficiently. The target ‘1-PERSON’ is crossing many other targets and its identity remains unchanged.

4 CONCLUSIONS

We have presented an online learning neural tracker to handle dynamic changes in object/background appearance, illumination and scale. The problem of tracking is treated as a binary classification problem, and online learning neural classifier is used to differentiate the object and the background region. The information from motion segmentation is fused with this neural classifier for robust object tracking. The results indicate that the tracker is robust under normal illumination variation, appearance and scale changes. Occlusions are also handled using the motion and the probability map information. Furthermore, consideration of different features like shape or silhouette might be beneficial. Additional research can also be carried out in order to handle separated parts of the same object and improve the initialization step.

REFERENCES

Avidan, S. (2007). Ensemble tracking. *IEEE Trans. Pattern Anal. Machine Intell.*, 29(2):261–271.

Bąk, S., Suresh, S., Bremond, F., and Thonnat, M. (2008). Fusion of motion segmentation and learning based tracker for visual surveillance. In *Internal report: INRIA Sophia Antipolis, http://www-sop.inria.fr/pulsar/personnel/Francois.Bremond/topicsText/neuralProject.html*, volume 1, pages 1–10.

Collins, R. T., Liu, Y., and Leordeanu, M. (2005). Online selection of discriminative tracking features. *IEEE Trans. Pattern Anal. Machine Intell.*, 27(10):1631–1643.

Cootes, T., Edwards, G., and Taylor, C. (2001). Active appearance models. *IEEE Trans. Pattern Anal. Machine Intell.*, 25(5):681–685.

Jepson, A. D., Fleet, D. J., and El-Maraghi, T. (2003). Robust online appearance model for visual tracking. *IEEE Trans. Pattern Anal. Machine Intell.*, 25(10):1296–1311.

Nummiaro, K., Koller-Meier, E., and Van Gool, L. (2003). An adaptive color-based particle filter. *Image vision computing*, 21(1):99–110.

Stauffer, C. and Brady, J. (2000). Learning patterns of activity using real-time tracking. *IEEE Trans. Pattern Anal. Machine Intell.*, 22(8):747–757.

Suresh, S., Sundararajan, N., and Saratchandran, P. (2008a). Risk sensitive loss functions for sparse multi-category classification problems. *Information Sciences*, 178(12):2621–2638.

Suresh, S., Sundararajan, N., and Saratchandran, P. (2008b). A sequential multi-category classifier using radial basis function networks. *Neurocomputing*, 71(7-9):1345–1358.

Williams, O., Blake, A., and Cipolla, R. (2005). Sparse bayesian learning for efficient visual tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1292–1304.

Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *ACM Computing Surveys*, 38(4):1–45.

Zouba, N., Boulay, B., Bremond, F., and Thonnat, M. (2008). Monitoring activities of daily living (adls) of elderly based on 3d key human postures. In *Proc. of International Cognitive Vision Workshop*, pages xx–xx, Greece.

Zúñiga, M., Brémond, F., and M.Thonnat (2006). Fast and reliable object classification in video based on a 3d generic model. In *Proceedings of the International Conference on Visual Information Engineering (VIE2006)*, pages 433–440, Bangalore, India.