MCMC MODULAR ENSEMBLE TRACKING

Thomas Penne[1,3], Christophe Tilmant[2],Thierry Chateau[2], and Vincent Barra[3]
[1]*Prynεl / TIMS, RD 974 Corpeau, 21190 MEURSAULT, France*
[2] *Institut Pascal, UMR 6602 CNRS, Blaise Pascal University, Campus des Cézeaux, 63170 AUBIERE, France*
[3]*LIMOS, UMR 6158 CNRS,Blaise Pascal University, Campus des Cézeaux, 63170 AUBIERE, France*
vincent.barra@univ-bpclermont.fr

Keywords: Object tracking, Classification, Adaboost, Ensemble Tracking, Feature spaces, Particle filtering.

Abstract: Object Tracking has become a recurrent problem in video-surveillance and is a important domain in computer vision. It was recently approached using classification techniques and still more recently using boosting methods. We propose here a new object tracking method, based on Ensemble Tracking and integrating two main improvements. The first one lies on the separation of the heterogeneous feature space into a set of homogenous subspaces (modules) and on the application of an Ensemble Tracking-based algorithm on each module. The second one deals with the new tracking problem induced by this separation by building a specific particle filter, weighting each module in order to estimate both position and dimensions of the tracked object and the linear combination of modular decisions leading to the most discriminative observation. Our method is tested on challenging sequences. We prove its performance and we compare its robustness with the state of the art.

1 INTRODUCTION

Numerous works identify object tracking as a critical issue in many applications (Hu et al., 2004). We herein define tracking as a two-step process which aims at estimating the trajectory of moving object from video sequences. The object is first detected, and potential candidates are identified in each frame. It is then tracked, and a specific candidate is tracked all along the frames. Depending on the constraints imposed, several algorithms are available (e.g. (Yilmaz et al., 2006) for a review). We impose in this article four constraints on the tracker: it has to be robust, real-time, usable from mobile cameras and able to track pedestrians. We are thus only interested in the following in points tracking and supervised learning based methods. Points tracking, in which the object is represented with a few points, brings together two methods widely used in the vision community: Kalman and particle filters. Particle filters are very efficient methods to track multiple objects, as they can cope with non-linearities and multi-modalities induced by occlusions and background clutter (Isard and Blake, 1998; Okuma et al., 2004). Supervised learning consists in inferring a function from supervised training data. The task of the supervised learner is to predict the class label of unknown data using only a given number of training samples. In the tracking community, the most popular supervised learning methods include the direct construction of an inter-classes frontier (e.g. SVM) or the combination of classifiers improving classification performance. In the context of pedestrian tracking, boosting, and especially Adaboost (Freund and Schapire, 1996), was proved to be very efficient (Grabner et al., 2006). We propose in this article to combine point tracking and supervised learning methods. More precisely, classifiers are trained with Adaboost on homogeneous feature spaces, and the classification decisions are used by a particle filter specially designed for the application. Some works are close to ours (Avidan, 2007; Tang et al., 2007; Nickel and Stiefelhagen, 2008), and we introduce here a modular version of ET (Ensemble Tracking) (Avidan, 2007) combined with a Markov chain Monte Carlo particle filter (MCMC). The key idea is to jointly track the object position/scale and the relevance of each observation module with a sequential Bayesian filter. In the following, we introduce our contributions, consisting first of a modular version of ET, and then on the introduction of a MCMC particle filter estimating both position and dimensions of the object to track, and weights of classifiers stemming from the modular ensemble tracking. We finally presents and analyzes results of our algorithm on synthetic and challenging video sequences.

2 MC^2-MET ALGORITHM

The ET algorithms is fully described in (Avidan, 2007), In this article tracking is performed on a heterogeneous feature space, and features used in can be reliable or not, and may be not discriminative enough and therefore may lead to a possible high global Bayesian error. In order to avoid these problems, we herein propose to work on several homogeneous feature spaces and to track the object using an ET-like algorithm on each of these spaces (called modules, one confidence map per space based on a consistent feature vector). Decisions are then combined into a unique one, managing their complementarity, reliability and their redundancy. Using one ET strong classifier per space allows an independent decision on each homogeneous feature space to be taken and therefore gives the possibility to handle undiscriminative data that may hinder the final decision stage. Splitting the feature space strongly modifies the objective of the tracking process: a tracking algorithm now has to estimate a hidden state composed on the one hand of the position and the dimensions of the object, and on the other hand of the linear weights of the module decisions, leading to the most discriminant observation. We therefore propose to use a specific particle filter jointly managing both the positions and dimensions of the object and the weights of the modules.

2.1 Synoptic view of MC^2-MET

The feature space is now composed of several homogeneous subsets (modules), composed of feature vectors representing pixel characteristics (e.g. colorimetric, texture-based, contour-based modules...) and the definition of relevant modules is application-based. For each of the modules, a strong classifier is built, using the ET algorithm. The set of resulting confidence maps allows several distinct object positions to be computed,that are combined into a single one using a specific particle filter. The synoptic diagram of the MCMC Modular Ensemble Tracking (MC^2-MET) is proposed in figure 1, and each step is detailed below.

The initialization step relies on the initialization of

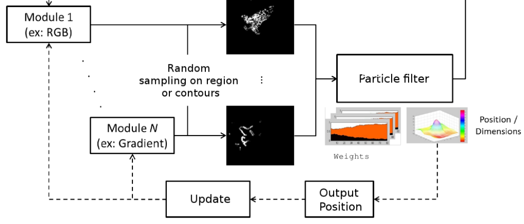


Figure 1: Synoptic view of the proposed algorithm.

ET, for each strong classifier of each module, but only a subset of the training examples is used. We indeed propose a sampling strategy of the training set to reduce the computational cost, adapted to each module: for a module that does not necessitate specific extraction rules (e.g. a colorimetric module), pixels are randomly chosen according to a Gaussian pdf, such as the number of pixels extracted from inside and ouside the region of interest are the same. The training zone is thus dynamically chosen and several background patterns can thus be managed. For modules with specific extraction rules (e.g. the histogram of oriented gradients module) , an adapted heuristic is generated.

2.2 Particle filter

A particle filter is a sequential Monte Carlo method used for Bayesian filtering. The particles are propagated through time by Monte Carlo simulation to obtain new particles and weights (usually as new information are received), hence forming a series of pdf approximations over time. Using the training sets of modules, a strong classifier is built for each of the modules. The set of strong classifiers is then used at the next iteration to build a confidence map for the object position. The particle filter aims at maintaining through time a set of particles jointly managing the position and dimensions of the object, and the weights to apply to the linear combination of the confidence maps in order to attain the best observability. The most popular particle filter algorithm is known as SIR algorithm (Isard and Blake, 1998). However, the number of required particles grows as an exponential of state-space dimension. Recent works proposed a MCMC space exploration strategy to overcome this limitation (Khan et al., 2005). We propose a similar algorithm to efficiently explore the space state in a realtime framework. The observation function is defined according to the confidence maps built from the current image. A particle i at time t is modeled as a specific rectangle centered in (x_i^t, y_i^t), with width w_i^t and height h_i^t, surrounding the object and to a set of weights $(w_{i,m}^t)$ for computing an unique confidence map from a linear combination of M ones. Since the score of the particle is computed from the confidence maps of the modules, and since the sampling imposed that these confidences are known for only a subset of pixels, we constrained the particles to only represent rectangles fully included in the image. We moreover imposed rectangle to have a "sufficient" size.

Propagation model: the particle based approximation of the state is achieved with a Markov Chain. At iteration i of the Chain at time t, we propose a marginal strategy to build the proposal sample from the particle $i-1$ of the Chain. A random choice allows to consider if either position and dimensions or the set of weights must be propagated. We then propose 3 type of random propagation for the position/dimensions information: an updating of position, an updating of dimensions or both. Each type is associated to a probability, and we empirically found that values 0.75, 0.2 and 0.05 gave good results.

Obsevation model: the observation model is defined as a likelihood function that gives a score c_i to any particle X_i^t. This score is then used in the Metropolis algorithm to infer if particle i will belong to the final Markov chain. For each particle X_i^t and each confidence map (i.e. each module m), two classification scores are computed: the mean classification score $S_m|_{\Omega_1}$ inside the rectangle Ω_1 surrounding the object and related to the current particle, and the $S_m|_{\Omega_2}$ outside this rectangle but inside a region of interest centered on (x_i^t, y_i^t) and three times larger. The global score of module m for the position/dimensions is then $s_{i,m} = S_m|_{\Omega_1} \times (1 - S_m|_{\Omega_2})$ and the score of the particle is finally computed as the weighted sum of $s_{i,m}$ with weights $w_{i,m}$. Since these scores are computed from the confidence maps c_m^t, we preprocessed these maps (Platt, 1999) in order both to suppress outliers and to transform the classification margins of the strong classifiers into calibrated probability values. More precisely, let Ω be the set of pixels for which a confidence value has been computed and $VCU_m^t(x,y)$ the confidence value computed by the strong classifier m at time t. The confidence value is given by:

$$c_m^t(x,y) = \frac{1}{1+\exp(A_m VCU_m^t(x,y) + B_m)}, \forall (x,y) \in \Omega$$

where A_m, B_m are computed by optimizing a cross-entropy function on the confidence map of m obtained on the first image of the sequence. The proposed particle is then accepted or rejected according to the Metropolis Hasting rule.

Module updating: once the particle filter has been applied, modules must be updated. We chose to apply the same updating process as in (Avidan, 2007) on each strong classifier. We only kept the best K strong classifiers at each iteration, based on the new position determined by the Mean Shift algorithm on the current image. We had to determine an unique updating position from the set of positions included in the different particles of the filter. Each pixel was first assigned a score equal to the number of particles for which the corresponding surrounding rectangle included this pixel. Pixels were then considered as object pixels if their score was greater than half the number of particles. In order to avoid the drifting effect, the current sample pixels were used together with sample pixels from the initial image. For each module and each updating step, two sets of labeled samples (initial and current) were available. From these sets, four sample groups (2 positives, 2 negatives) were randomly built and each pair (positive/negative) was used to either update the strong classifier or estimate A_m and B_m.

3 RESULTS

MC^2-MET was implemented in C++, on a PC equipped with Intel® Core 2 Duo E8500 3.16GHz and 4Go of RAM DDR2. Several challenging video sequences were used to demonstrate the efficiency of MC^2-MET algorithm, mainly extracted from the CAVIAR and the PETS2001 database. Simulated, homemade and available (Stalder et al., 2009) sequences were also used. Due to the 4 pages constraint, we only present comparisons with the state of the art.

3.1 MC^2-MET vs. Ensemble tracking

Since the basic principle or MC^2-MET relies on the ET algorithm, we first compared ET and our method on 6 CAVIAR sequences (Browse4, Fight OneManDown, TwoEnterShop2cor, OneStopMove-NoEnter2cor, with different tracking objectives). For both algorithms, RGB levels and HoG values were computed in a 5×5 neighborhood and rebinned in 8 classes. For ET, the feature vector was thus a 11D ; for MC^2-MET a 3D colorimetric feature vector and a 8D contour-based one were used. Table 1 presents a comparative study. For each sequence and each algorithm, the mean and standard deviation of Euclidean distances between the center of the computed rectangle and the ground truth are calculated, and the tracking status is reported (KO: target lost, OK: tracking completed). ET lost the target for all the CAVIAR videos showing a Shopping Center in Portugal (Seq.3 to 6). Quantitavive performances as well as target tracking were always worse using ET, since this algorithm supposes scale invariance: a change in object scale creates some opportunity for ET to find a better correspondance in other parts of the region of interest. An analysis of the first sequence reveals that ET can have results comparable to our algorithm when the conditions are adequate (no great deformation, no important change in scale, and no similar object near the object to be tracked).The scale invariance does not fully explain the the difference for sequences 3 to 6. Since MC^2-MET is modular, it allows a decision to be taken on each feature space. When combining module decision using the particle filter, MC^2-MET builds a final position + dimensions that can manage non relevant information stemming from modules.

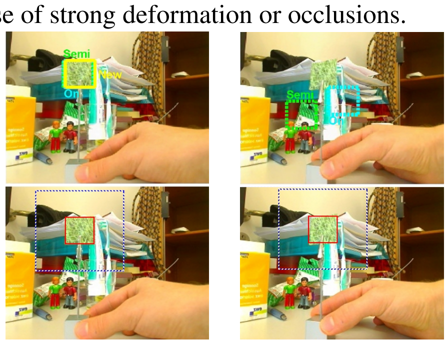
3.2 MC^2-MET vs. classical approaches

We compared MC^2-MET (using RGB and LBP modules) with classical tracking algorithms: online Boosting (*OB*, (Grabner et al., 2006)), semi-supervised online Boosting (*SSOB*, (Grabner et al., 2008)), and beyond semi-supervised Tracking (*BSST*, (Stalder et al.,

Table 1: Comparison of ET/MC^2-MET tracking results.

Caviar Seq.	Dist. ET/ MC^2-MET (pixels)	Status
1	5.92 ± 2.97 /5.98 ± 2.33	OK/OK
2	10.28 ± 4.59 /5.83 ± 2.43	OK/OK
3	34.40 ± 19.47/9.00 ± 5.23	KO/OK
4	92.09 ± 89.42/9.99 ± 4.30	KO/OK
5	29.83 ± 29.27/ 9.54 ± 4.73	KO/OK
6	16.64 ± 5.28/ 13.73 ± 6.85	KO/ OK

2009)), considered as references in the tracking community. The comparison was performed on a sequence proposed by the authors of the algorithms (Figure 2), and sheds light on several points. *OB* proposed an online version of Adaboost allowing classifiers to be constantly updated. *SSOB* proposes to handle both the drifting effect and the change of appearance of the target. For this, authors combined principles stemming from semi-supervised learning and adaptative online boosting for feature selection. *BSST* as for it dissociates detection, recognition and tracking tasks in distincts classifiers. Images (1) corresponds to the beginning of the sequence. Initialisations are performed for all algorithms on the rectangular texture object. Image (2) shows a deformation of the object. If MC^2-MET successfully tracks the object, the 3 other algorithms fail: *OB* and *SSOB* drifted to a more relevant candidate, and *BSST* considered that the object disappeared (no yellow rectangle). The method used to search objects in these algorithms is more global than in MC^2-MET, and object detection is much more restritive and can lead to target looses in case of strong deformation or occlusions.



(1) (2)

Figure 2: Tracking results for MC^2-MET, OB, SSOB, BSST. Line 1 gives the results of OB (cyan),SSOB (green) and BSST(yellow). Line 2 gives results of MC^2-MET(solid rectangle: object, dashed one: region of interest).

4 CONCLUSIONS

We presented in this article a modular version of Ensemble Tracking combined with a Markov Chain Monte Carlo particle filter (MCMC). The key idea is to jointly track the object position/scale and the relevance of each observation module with a sequential Bayesian filter. We proposed a special particle filter (MCMC) that maintains over time a set of particles corresponding to a hidden state composed of the position of the tracked object but also of all the weights to be applied to different sub-decisions in order to obtain compliance with this condition most discriminating. We finally presented and analyzed results of our algorithm on synthetic and challenging video sequences recorded on fix and mobile cameras. The comparaison versus other classical approches showed a better accuracy and better robustness compared to occlusions. Several extensions are now expected. We now plan to extend the number and type of modules, computing e.g. spatio-temporal or a priori modules (silhouette). Modules also have to be managed in real-time, so that relevant (resp. irrelevant) modules can be automatically selected (resp. discarded) at each time.

Bibliography

Avidan, S. (2007). Ensemble tracking. *IEEE Trans. on PAMI*, 29(2):261–271.

Freund, Y. and Schapire, R. (1996). Experiments with a new boosting algorithm. In *ICML'96*, pages 148–156.

Grabner, H., Grabner, M., and Bischof, H. (2006). Real-time tracking via on-line boosting. In *BMVC'06*, pages 47–56.

Grabner, H., Leistner, C., and Bischof, H. (2008). Semi-supervised on-line boosting for robust tracking. In *ECCV'08*, pages 234–247.

Hu, W., Tan, T., Wang, L., and Maybank, S. (2004). A survey on visual surveillance of object motion and behaviors. *IEEE Trans. on Sys., Man. Cyb.*, 34(3):334–352.

Isard, M. and Blake, A. (1998). Condensation - conditional density propagation for visual tracking. *Int. Jal.Comp. Vision*, 29(1):5–28.

Khan, Z., Balch, T., and Dellaert, F. (2005). MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Trans. PAMI*, 27:1805 – 1918.

Nickel, K. and Stiefelhagen, R. (2008). Dynamic integration of generalized cues for person tracking. In *ECCV'08*, pages 514–526.

Okuma, K., Taleghani, A., de Freitas, N., Little, J., and Lowe, D. (2004). A boosted particle filter: Multitarget detection and tracking. In *ECCV'04*, pages 28–39.

Platt, J. (1999). *Advances in Large Margin Classifiers*, chapter Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods, pages 61–74. MIT Press.

Stalder, S., Grabner, H., and Gool, L. V. (2009). Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In *OLCV'09*.

Tang, F., Brennan, S., Zhao, Q., and Tao, H. (2007). Co-tracking using semi-supervised support vector machines. In *ICCV'07*, pages 1–8.

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