Adaptive tracking via multiple appearance models and multiple linear searches

Tuan Nguyen

Tony Pridmore

Computer Vision Laboratory
School of Computer Science
University of Nottingham

Abstract

We introduce a unified tracker (FMCMC-MM) which adapts to changes in target appearance by combining two popular generative models: templates and histograms, maintaining multiple instances of each in an appearance pool, and enhances prediction by utilising multiple linear searches. These search directions are sparse estimates of motion direction derived from local features stored in a feature pool. Given only an initial template representation of the target, the proposed tracker can learn appearance changes in a supervised manner and generate appropriate target motions without knowing the target movement in advance. During tracking, it automatically switches between models in response to variations in target appearance, exploiting the strengths of each model component. New models are added, automatically, as necessary. The effectiveness of the approach is demonstrated using a variety of challenging video sequences. Results show that this framework outperforms existing appearance based tracking frameworks.

1 Introduction

Visual tracking is a time dependent problem. Its aim is to model target appearance and use it to estimate the state of a moving target, retrieve its trajectory, and maintain its identity through a video sequence. Two important components for a tracker are the search method and the appearance model matching approach. The tracking problem can be formulated as searching for the region with the highest probability of being generated from the appearance model. The search method could be a sliding window or sampling approach or use target motion to hypothesise where the target might be. The target appearance is typically constructed from the first frame by extracting features. These are then compared to measurements recovered from incoming frames at candidate target positions to estimate the most likely target state.

In real world scenarios, targets' appearance can, however, vary over time as a result of illumination changes, pose variations, target movement and/or camera movement, full or partial occlusions by other targets or by objects in the background, target deformation and complex background clutter. Also, their appearance might have the same appearance as their local background, which may attract the tracker. Adapting to these changes, however, exposes the tracker to *model drift*: localisation errors cause background information to be included in the appearance model, which gradually leads the tracker to lose its target. The

risk and degree of drift increases quickly if the tracked target is not well-located. The key to the model drift problem is to locate the target position precisely and recognise abnormal appearance changes before trying to update the target appearance.

We propose an online tracker capable of adapting to appearance changes without being too prone to drifting, and able to recover from drifting and partial or full occlusion. A number of questions should be considered during tracking: What appearances should be used for tracking? When should additional appearances be learnt? How can complex target movement be recovered precisely? In visual tracking, the best match at time t to appearance observed at time $t-1$ may not be the target, because of changes in visual properties. Thus, to reduce the risk of adaptation drift, additional constraints or supervision of the appearance model are needed. Figure 1 gives an overview of the proposed framework. The tracker contains two crucial components: the first learns target appearance changes during tracking and the second utilises features to enhance target prediction via multiple linear searches.

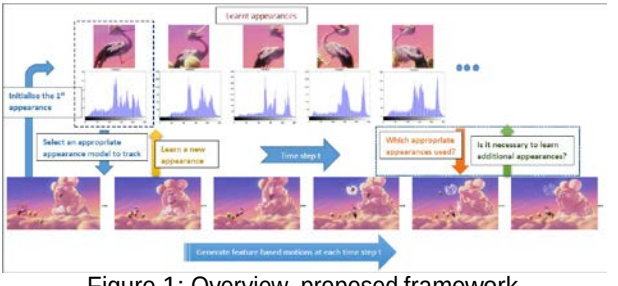


Figure 1: Overview proposed framework.

Our simple yet effective method builds appearance models which are a combination of two popular generative models: templates and histograms. When a suitable model is available, templates can provide stable matching and good localisation, due to the detailed spatial information they carry, and play a role as landmarks to reduce drift. Templates provide a solid anchor for target location when appearance is known. Templates are, however, very vulnerable to appearance changes. Histograms, in contrast, do not maintain spatial information and so are more robust to rotation and partial occlusion. Histograms can be thought of as a more abstract model; as many templates can produce a given histogram. The relative lack of precision of histogram based representations allows them to capture target appearance during changes in the spatial distribution of target features.

During tracking, especially in unconstrained environments, appearance changes are unpredictable. A fixed set of templates cannot be relied upon to capture the variations that might arise. Similarly, if only histograms are considered, there is no clear cue as to which histogram should be used, or when to construct a new histogram. However, with careful use, templates and histograms can complement each other. Templates allow the tracker to produce suitable histograms, which allow the tracker to estimate target location during appearance changes, which in turn allows new templates to be sought.

In the proposed method, each new appearance is learnt and maintained in a pool of appearance models. Storing multiple template-histogram pairs allows the tracker to handle variations by automatically switching among models, using template-matching to select a histogram which captures target appearance in the current frame. This reduces the risk of drifting, since it can check the similarity between the new and previous appearances before updating appearance model or adding a new appearance model to the pool. In the case of drifting or occlusion, the tracker can re-initialise the tracking process by selecting a new model from the pool.

To complete the tracker, by providing precise motion estimates during unexpected and abrupt target movement, we propose a mechanism utilising target features detected and matched between consecutive frames. Our method can predict target location without using a complex motion model or models, and select an appropriate model with which to search.

To robustly represent target motion and predict location, both bottom-up and top-down techniques are used. The top down component uses (simple) motion models to generate hypotheses (particles). The bottom up component extracts local motion estimates which inform the motion models, supporting top-down search. Local features of the target are identified, matched between adjacent frames and stored in a feature pool. While individual feature matches may be incorrect, the distribution of likely motion directions supplied by feature matching provides valuable information that can be used to guide search. Each feature match constitutes a hypothesis as to the direction of motion of the target. The distribution of motion directions provides an implicit representation of complex target movements which are difficult to model explicitly. In the proposed tracking algorithm, the search space is modelled as multiple potential directions and one-dimensional searches are performed in those directions to find the target, reducing and carefully targeting the search.

The proposed appearance and search mechanism are built into the Markov Chain Monte Carlo (MCMC) based particle filter [12]. We extend the proposal distribution of the standard MCMC to propose both the new location via motion direction sampling, and the appearance model that should be used. On completion of each Markov chain, each histogram is assigned a weight reflecting how frequently it was accepted during that chain. The new target location is estimated by identifying particles which have the highest weight and use the most common histogram. This strategy is adopted because, if the chain runs for long enough, the most suitable histogram will be used most.

2 Previous Work

Visual tracking is a longstanding problem in computer vision and a number of reviews exist [17, 29]. Many methods proposed aim to develop a richer appearance model, to help distinguish targets and make the tracker more robust. A fixed appearance model, as in [3, 11], cannot handle target appearance changes sufficiently. To achieve long term tracking, many researchers have tried to learn appearance models (e.g. [2, 4, 5, 8, 20, 24]). Regardless of approach, adaptive appearance-based trackers face a key problem: *model drift*.

Several methods have been proposed to deal with the drift problem (e.g. [18]). A fixed adaptation speed used in a simple linear update of the reference model [19] is suitable in some situations. [4] proposed to anchor the developing model on the original one, but the method could not react quickly enough to large variations. Multiple instance learning [2] has been proposed to handle location ambiguity in positive samples by using a positive bag and negative bag. This method may, however, select less informative features. Grabner et al. [9] proposed semi-supervised boosting to break the self learning loop in their Online Boosting method [8]. Despite its success in alleviating drift, this framework does not handle target changes well if the appearance becomes different from the prior.

Other frameworks have focussed on target motion, seeking to enhance prediction and reduce search space. For example, [22] used a random walk model, [21] described a proposal distribution mixing hypotheses generated by an AdaBoost detector and a standard autoregressive motion model, [10] combined two models, [23] used Kernel Mean Shift to control hypotheses generated by an annealed particle filter, [14] used a two stage dynamic model.

These methods all assumed target appearance to be (approximately) constant.

Fusion trackers (e.g. [15, 16]) have been proposed to combine multiple appearance and multiple motion models. Each tracker comprises a single appearance model and motion model to deal with a specific appearance change and different target motion. All trackers can operate in parallel and interact with each other. The challenge, however, raised by these works is how to ensure agreement among these trackers.

3 A Tracking Framework

Figure 2 shows the main steps in the proposed method, FMCMC-MM. This approach maintains an appearance pool containing appearance models learnt during tracking and a feature pool storing features detected in the previous and current frames. These features are used to support target motion modelling.

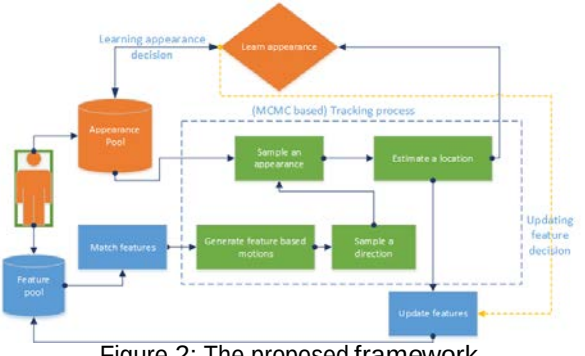


Figure 2: The proposed framework.

3.1 Appearance model

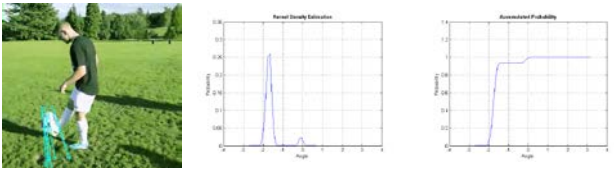
Targets are selected by manual annotation of the first image in the sequence. Once target location is specified, its template is extracted and added to the appearance pool. For each template, a simple generative model is constructed - an Epanechnikov kernel weighted colour histogram [5]. Colour is chosen here as a simple, but powerful and reliable feature widely used to model appearance when tracking objects against complex backgrounds. To compare the reference histogram p of the target with the candidate histogram q_t at state vector X_t, we use the Bhattacharyya distance. When comparing template and image data or pairs of templates, we use the Normalised Correlation Coefficient (NCC) to reduce the effect of illumination changes.

3.2 Motion Model

Target features are extracted by applying the method of Shi and Tomasi [26] within the target's bounding box. Shi and Tomasi proposed an affine model which proved adequate for region matching and provides the repeatable interest points needed to support robust tracking [25]. Features are defined as $f^i = (x^i, y^i, dx^i, dy^i)$ where f^i is the i^{th} feature, (x^i, y^i) is the location of the feature, and (dx^i, dy^i) gives its displacement relative to horizontal and vertical axes. The target maintains a feature pool $F_t = \{F^p_{t-1}, F^c_t\}$ at each time t which contains features detected in the previous tracked frame $F^p_{t-1} = \{f^i_{t-1}\}_{i=1..m}$ and features matched $F^c_t = \{f^i_t\}_{i=1..m}$ in the current frame, where as m is the number of features considered.

Each feature point extracted from the target is matched with features identified in the subsequent frame using a pyramidal implementation of the Kanade - Lucas – Tomasi tracker ([30]) forming a set of vectors $V_t = \{v_t^i\}_{i=1..m}$ linking matched features. This approach was selected for its ability to handle large movements. The Gaussian kernel density (KDE) is also used to estimate the motion direction distribution based on the local feature matches.

Rather than search the image in two dimensions, the proposed approach divides the search space into multiple linear segments corresponding to directions in which the target might move. To estimate target location, a motion direction is randomly selected from the distribution obtained by feature matching. Search in a given direction starts from the best state found in the previously selected (and searched) direction. X_t is the most likely state at time t of the target, X_t^I is the most likely state at time t of the target within a selected linear segment. Figure 3 shows an example of feature matching and its KDE.



(a) Features matched (b) KDE (c) Accumulated Prob.

Figure 3: Motion directions and their KDE at Frame #22 of the Football sequence.

3.3 Sampling Appearance & Motion Models

The motion and appearance model presented here are embedded into the MCMC method of [12]. MCMC methods define a Markov Chain over the state space X. A candidate particle X_t^I, sampled from the current sample X_t using a proposal $Q(X_t^I; X_t)$, is accepted if the acceptance ratio (in [12]) exceeds 1. A maximum a posterior (MAP) has typically been used to find a particle most likely the target over N samples at each time t. At each time step t, an appearance pool containing templates $T_t = \{T^j\}_{j=0..k}$ and equivalent generative models $G_t = \{G^j\}_{j=0..k}$ is given, where k is the current size of the pool.

Information from previous frames can be used to improve the accuracy of the prediction and reduce the search space; the target's previous location has been used in many trackers. In our approach, *three* pieces of information are used when predicting target location. *First*, the previous target location is used to decide the centre of the search area. The search area S is double the target size. *Second*, the confidence score matrix $C^j = NCC(T^j, I)$ is calculated by using NCC to compare each template T^j from T_t to each location $I(x, y)$ of image sequence I belonging to S. *Third*, features matched from the previous image are used to improve the initial location of an MCMC chain. Define $m^f = \sum_{i=1}^{m}(f^i \subset P)$ as the number of features in the current frame belonging to an image patch P defined by the target's bounding box.

Tracking begins with the initialisation of an MCMC chain. Starting position is determined where the confidence score at that location $C^J(x, y) \geq \theta_d$ and contains the maximum number of m^f. If no location satisfies these conditions because no templates learnt before produce a confidence score which is greater than θ_d, the starting position is determined using a second order auto regressive motion model. The initial appearance model is the histogram associated with the template that best matches the last recorded target location.

As the MCMC chain progresses, new states are proposed according to the proposal density $Q(X_t^I, X_t)$. The proposal comprises changes in position according to the motion model, from which a motion direction is randomly selected (Section 3.2), and an appearance model

(histogram) randomly selected from the appearance pool. Each appearance model has an associated weight, which records the number of times it was selected and accepted within the chain. Intuitively, the model that most improves the state hypothesis, and so can be assumed to best describe the target, will have the highest weight. Model selection takes this weight into account, better models are more likely to be selected as the chain develops. Each generated particle records its hypothesised target position, the weight associated with its appearance model, and the Bhatacharya distance between that model and the local image data. At the end of the MCMC process, the most highly weighted appearance model is identified. The particle generated using the model that has the best fit to the local image data provides the new estimate of target location. The motion direction sampling is then reapplied and templates matched to the estimated location to initialise processing into the next time frame. The tracking process is as described in Algorithm 1.

3.4 Updating Appearance Models

Updating an existing model: After the locating the target in a given frame, a new template is constructed from the local image data, compared to the current template and the NCC computed. If the correlation score is greater than a (high) threshold, the histogram model is updated; i.e. the histogram associated with the current template is replaced by the histogram of the new estimated target location. The effect is to update a generative model (the new histogram) while anchoring it with a related, earlier template. Use of the template to select the initial histogram in the MCMC chain allows the combined model to adapt without excessive risk of drift. The approach is conservative in two ways: the histogram is only updated if new data is a close match to the current best model, and the template remains fixed.

Adding a new model: When the new template differs from both the current selected model and the members of the current appearance pool a new model is created and added to the pool. Effective tracking with a single, fixed histogram is possible when target appearance is also (approximately) fixed. Adding more appearance models, however, allows the tracker to respond to future changes in target appearance.

Together, these mechanism extend the third strategy, Template Update with Drift Correction, of [18]. Existing models are kept unchanged, as they may support effective tracking in later frames, and the overall appearance model is updated implicitly by modifying its components. If a poor model is added, the tracker still has a chance to recover by selecting other, more correct appearance models. The proposed update method is different from those mentioned in Section 2, which contain and explicitly update a single appearance model.

3.5 Handling Occlusion & Re-detecting the Target

Occlusion is detected when both the NCC of the current template and location estimate, and the Bhattacharya distance between the current model histogram and the histogram computed around the location estimate, fall below a threshold. When this occurs a sliding window technique, commonly applied in tracking by detection and trackers no prediction mechanism, together with all pooled appearances is used to re-detect the target. The location with the best match is taken as the position of the re-appeared target. Note that an advanced occlusion detection, e.g. employing Semi Boosting [9], could be embedded into this framework. An occlusion detection step is necessary because the motion model is computed from local features; explicit detection of occlusion reduces the likelihood that features of the occluding object will over-rule features belonging to the true target.

3.6 Updating Motion Models

The target motion model depends on feature detection and matching. Features help the tracker handle motion variation and abrupt motion naturally by allowing the tracker develop a good sense of where the target might be. The features used should be updated as tracking progresses, as some will become invisible and others appear over time. Features are only updated if there is no occlusion.

A bounding box does not always provide a good fit to the target boundary, and some detected features may be outliers i.e. belong to the local background. The motion direction sampling method can deal with this problem, assuming that most of the features considered lie within the true target boundary.

3.7 Algorithm

Define M as the thinning interval before accepting one particle, a burn in period B, N_l is the number of particles used to search one line, L is the total number of lines considered. The tracking process is then as described in Algorithm 1.

Algorithm 1 Multiple appearance models and motion direction sampling (FMCMC-MM)

1. Detect and match features and compute the motion direction distribution as described in Section 3.2
2. Initialise the start state X_t for the target using features detected and templates in the pool as described in Section 3.3.
3. Initialise equal weight for each histogram model.
4. Repeat L times
   (a) Randomly select one direction from KDE of the target.
   (b) Calculate the slope s of the selected direction.
   (c) Propose a new state $Q(X_t^l; X_t)$.
   (d) Calculate the intercept b for the line using the slope s and new X_t^l.
   (e) Repeat $M * N_l$ times
      i. Generate $X"_t$ from X_t^l according to the s and b.
      ii. Propose a candidate appearance model for $X"_t$ according to the appearance weight.
      iii. Compute the acceptance ratio a.
      iv. If $a \geq 1$, then accept $X"_t$: Set the target in X_t^l to $X"_t$, increase the weight for the selected histogram and update the cached likelihood. Otherwise, accept with probability a. If rejected, leave X_t^l unchanged.
   (f) If the state X_t^l is better than X_t then move X_t to X_t^l.
5. The set of particles is obtained by storing N_l best particles at each direction.
6. The current posterior $P(X_t|Z_{1:t})$ is approximated by using MAP.
7. Check if the target is in occlusion as in Section 3.5.
8. Update the target model as in Section 3.4.
9. Re-detect features for the target (i.e. update motion model).

4 Experiments and Results

4.1 Data

Table 1 lists the video sequences used in experimental evaluation of the proposed method. Ground truth data was manually created, capturing the visible part of the target by a rectangle bounding box. It also takes into account scale changes.

4.2 Experimental Settings

We compared our new proposed method FMCMC-MM to other existing methods: conventional MCMC (our implementation), Online AdaBoost (OAB) [8], Semi Boosting (SB) [9], FragTrack (Frag) ([1]), IVT [24] and Visual Tracking Decomposition (VTD) [15].

Sequence	Challenge	Frames
Bouncing1(our collection)	Fast & unexpected movement, Deformation	654
Bouncing2(our collection)	Fast motion, Rotation	90
Bird2([28])	Deformation, Rotation, Occlusion	98
Table tennis(our collection)	Unexpected movement, Clutter	138
Emilio(synthesised by [27])	Fast & unexpected Motion, Scale changed, Occlusion	280
Animal(synthesised by [27])	Fast Motion, Clutter	71
Football(our collection)	Fast Motion, Clutter, Distractor	124
David2(synthesised by [27])	Illumination and Pose Variation, Distractor	537
Tiger1(synthesised by [27])	Fast motion, target rotates, occlusion, appearance deformed	354
Jogging(synthesised by [27])	Pose variation, Full occlusion, Deformation	307
Rolling Ball([13])	In-Plane rotation, Scale changed, Partial occlusion	601
Girl(synthesised by [27])	Scale changed, Face expression changed, rotation	500

Table 1: Testing video sequences and their challenges

OAB, SB, FragTrack and IVT rely heavily on rich appearance models to find the target. VTD was selected because it used sampling methods to sample appearance and motion models to construct trackers. Although their approach is different from ours, their sampling strategy is similar. We used 300 particles and an 8 bin histogram for each colour channel in FMCMC-MM and MCMC. The search areas of OAB and SB were set to twice the target size and of FragTrack and IVT were set 40x40 pixels (the maximum displacement of the centre of the target from one frame to the next). In OAB and SB, we used 100 feature selectors. Each selector maintained 10 features.

4.3 Result & Discussion

Table 2 summarises the results obtained. The numbers in the Table 2 give the centre location error (in pixels) averaged over all frames of each sequence, i.e the average distance of the predicted bounding box from the centre of the ground truth bounding box. The lower number is, the better the result, and the numbers in **{}** indicate the number (%) of successfully tracked frames (score>0.5), where the score is defined by the overlap ratio between the predicted bounding box B_p and the ground truth bounding box B_{gt} and calculated $score = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}$ ([6]). The higher a number is, the better the result. Each sequence was run three times with each tracking framework. The best result are marked in bold and the second best underlined. Table 2 shows that FMCMC-MM performed more accurately on **9 out of 12** sequences. Supplementary materials for this paper have been provided. In

Sequence	FMCMC-MM	MCMC	SB	Frag	IVT	VTD	OAB
Bouncing1	**3.00 {100}**	8.78 {94}	28.61 {86}	4.30 {96}	5.49 {99}	11.87{93}	28.07 {88}
Bouncing2	**2.32 {100}**	34.80 {76}	216.21 {21}	56.56 {46}	161.86 {1}	153.59 {1}	152.34 {1}
Bird2	15.78 {72}	22.54 {49}	174.02 {38}	29.03 {32}	164.07 {4}	111.83 {13}	**7.59 {98}**
Table tennis	**3.46** {99}	3.59 **{100}**	153.26 {6}	13.36 {74}	251.10 {14}	380.51 {6}	642.12 {6}
Emilio	**6.67 {87}**	8.99 {76}	226.87 {8}	206.40 {11}	68.46 {27}	20.30 {65}	235.37 {8}
Animal	11.12 **{100}**	272.66 {7}	48.50 {38}	62.13 {39}	**8.67 {100}**	208.14 {6}	366.73 {3}
Football	**5.73 {94}**	76.24 {18}	60.78 {6}	31.32 {41}	114.10 {7}	34.92 {45}	60.56 {14}
David2	**2.12 {100}**	6.17 {73}	14.96 {33}	57.78 {26}	67.67 {19}	3.52 {89}	6.28 {63}
Tiger1	**23.43 {53}**	24.52 {48}	122.26 {41}	63.17 {34}	280.84 {1}	109.22 {18}	63.25 {47}
Jogging	**5.08 {96}**	29.66 {67}	55.98 {71}	15.55 {75}	90.84 {25}	92.40 {25}	161.31 {25}
Rolling Ball	**5.66 {83}**	6.34 {81}	168.81 {17}	8.84 {72}	98.79 {11}	33.24 {50}	159.21 {16}
Girl	6.76 {78}	36.79 {51}	35.55 {40}	6.84 {75}	609.99 {13}	7.28 {64}	**3.48 {96}**

Table 2: Average center location error (in pixel) and (%) Overlap rate in **{}**.

Bouncing1, Bouncing2, Emilio and Animal sequence, most trackers (e.g. MCMC, VTD, FragTrack, OB, SB) suffered when the target moved in unexpected directions. With the use of feature based motion modelling, FMCMC-MM, however, predicted target locations

correctly. Though VTD contains multiple motion models, these motions can only capture smooth motions. VTD, for instant, lost its target at Frame #58 (the Emilio sequence) when the target starts to jump at Frame #57.

MCMC, FragTrack and SB were affected by distractors in the Football and David2 sequences. In the Football sequence, the football, socks and shorts of the player have similar appearance. SB and MCMC therefore locked onto the player's ankle (Frame #22). FMCMC-MM performed well on the Football sequence because the motion direction distribution (Figure 3(b)) allows most of the selections (around 90% from accumulated probability) will be angles in the range (-1.9;-1.5) radians. These point downwards, towards the ground beneath the ball, rather than towards the player's ankle. VTD could track the target at Frame #22 because its multiple motion models give it a better chance of locating the target. It, however, completely lost the target at Frame #57 because of the target's quick movement.

The target is occluded by a pillar in the Jogging sequence at Frame #69. SB and FMCMC-MM can re-detect the target using a sliding window technique. They, therefore, could start to track the target at Frame #79. SB, however, lost its target in several frames (e.g. Frame #95) because the target changed her pose.

OAB worked very well in the Girl sequence since it can learn and adapt to appearance changes if these changes stay inside the boundary specifying the target. The fixed target model MCMC lost the target at the Frame #84 while the target was turning around. With an adaptive appearance model, FMCMC-MM and VTD worked well in the Girl sequence, though VTD lost its target at Frame #299.

5 Discussion and Conclusion

We have proposed a single tracking algorithm (i.e. without fusing multiple trackers) applicable to both rigid and deformable targets. The appearance model combines two popular generative models, utilising their complimentary advantages to improve tracking performance. The tracker uses a pool of template-histogram pairs to provide the best fit appearance model, switching among them using a sampling mechanism. Appearance changes are automatically detected and new, corresponding templates are extracted. These templates are carefully checked for similarity to other templates maintained in the pool before adding them to it. The MCMC-based search uses the distribution of motion directions of local image features from the feature pool to enhance target prediction. These local motion directions are extracted directly from two consecutive frames. The algorithm can also handle variation in the motion of a target without using any prior knowledge of movement. Experiments showed the FMCMC-MM framework to have performance advantages over other trackers.

FMCMC-MM detects target appearance changes using the templates maintained in the appearance pool. Should the target change its appearance very often in a long video sequences, many models may be stored, some of which will become irrelevant. To cope with this problem, some learnt appearances should be removed from the pool. Care must, however be taken not to remove appearances which would be useful later. This will be the subject of future work. Note also that there is no motion learning mechanism in FMCMC-MM. The target motion is derived by detecting and matching sparse features. These matches could be used to enhance learning of target motion.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 798–805, 2006.

[2] B. Babenko, Ming-Hsuan Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(8):1619–1632, 2011. ISSN 0162-8828. doi: 10.1109/TPAMI.2010.226.

[3] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 232–237, Jun 1998.

[4] R.T. Collins, Yanxi Liu, and M. Leordeanu. Online selection of discriminative tracking features. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10): 1631 –1643, oct. 2005. ISSN 0162-8828. doi: 10.1109/TPAMI.2005.205.

[5] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564 – 577, may 2003. ISSN 0162-8828. doi: 10.1109/TPAMI.2003.1195991.

[6] Mark Everingham, Luc Gool, ChristopherK.I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. ISSN 0920-5691.

[7] J. Giebel, D.M. Gavrila, and C. SchnÃűrr. A bayesian framework for multi-cue 3d object tracking. In TomÃąs Pajdla and JiÅŹÃ■ Matas, editors, *Computer Vision - ECCV 2004*, volume 3024 of *Lecture Notes in Computer Science*, pages 241–252. Springer Berlin Heidelberg, 2004. ISBN 978-3-540-21981-1. doi: 10.1007/978-3-540-24673-2_20. URL http://dx.doi.org/10.1007/978-3-540-24673-2_20.

[8] H. Grabner and H. Bischof. On-line boosting and vision. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 260–267, 2006.

[9] Helmut Grabner, Christian Leistner, and Horst Bischof. Semi-supervised on-line boosting for robust tracking. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 234–247, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-88681-5.

[10] M. Isard and A. Blake. A mixed-state condensation tracker with automatic model-switching. In *Computer Vision, 1998. Sixth International Conference on*, pages 107–112, 1998.

[11] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *Proceedings of the 4th European Conference on Computer Vision-Volume I - Volume I*, ECCV '96, pages 343–356, London, UK, UK, 1996. Springer-Verlag. ISBN 3-540-61122-3.

[12] Zia Khan, T. Balch, and F. Dellaert. Mcmc-based particle filtering for tracking a variable number of interacting targets. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(11):1805 –1819, nov. 2005. ISSN 0162-8828.

[13] D.A. Klein, D. Schulz, S. Frintrop, and A.B. Cremers. Adaptive real-time video-tracking for arbitrary objects. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 772–777, Oct 2010. doi: 10.1109/IROS.2010.5650583.

[14] M. Kristan, S. Kovacic, A. Leonardis, and J. Pers. A two-stage dynamic model for visual tracking. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 40(6):1505–1520, 2010. ISSN 1083-4419.

[15] Junseok Kwon and Kyoung Mu Lee. Visual tracking decomposition. In *in CVPR*, 2010.

[16] Junseok Kwon and Kyoung Mu Lee. Tracking by sampling and integrating multiple trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99 (PrePrints):1, 2013. ISSN 0162-8828.

[17] Xi Li, Weiming Hu, Chunhua Shen, Zhongfei Zhang, Anthony Dick, and Anton Van Den Hengel. A survey of appearance models in visual object tracking. *ACM Trans. Intell. Syst. Technol.*, 4(4):58:1–58:48, October 2013. ISSN 2157-6904.

[18] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):810–815, 2004. ISSN 0162-8828. doi: 10.1109/TPAMI.2004.16.

[19] Katja Nummiaro, Esther Koller-Meier, and Luc Van Gool. An adaptive color-based particle filter. 2002.

[20] Katja Nummiaro, Esther Koller-Meier, and Luc Van Gool. An adaptive color-based particle filter. *Image and Vision Computing*, 21(1):99 – 110, 2003. ISSN 0262-8856.

[21] Kenji Okuma, Ali Taleghani, Nando De Freitas, O De Freitas, James J. Little, and David G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *In ECCV*, pages 28–39, 2004.

[22] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *In Proc. ECCV*, pages 661–675, 2002.

[23] T. P. Pridmore, A. Naeem, and S. Mills. Managing particle spread via hybrid particle filter/kernel mean shift tracking. In *Proc. BMVC*, pages 70.1–70.10, 2007. ISBN 1-901725-34-0.

[24] David A. Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking, 2008.

[25] D. Serby, E.K. Meier, and L. Van Gool. Probabilistic object tracking using multiple features. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 184–187 Vol.2, 2004.

[26] J. Shi and C. Tomasi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, pages 593–600, 1994. doi: 10.1109/CVPR.1994.323794.

[27] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[28] Fan Yang, Huchuan Lu, and Ming-Hsuan Yang. Robust superpixel tracking. *Image Processing, IEEE Transactions on*, 23(4):1639–1651, April 2014. ISSN 1057-7149.

[29] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey, 2006.

[30] Jean yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000.