Confidence-Based Dynamic Classifier Combination For Mean-Shift Tracking

I.S. Topkaya*, H. Erdogan

Vision and Pattern Analysis Laboratory
Sabanci University - Faculty of Engineering and Natural Sciences
Tuzla, 34956, Istanbul, Turkey

Abstract

We introduce a novel tracking technique which uses dynamic confidence-based fusion of two different information sources for robust and efficient tracking of visual objects. Mean-shift tracking is a popular and well known method used in object tracking problems. Originally, the algorithm uses a similarity measure which is optimized by shifting a search area to the center of a generated “weight image” to track objects. Recent improvements on the original mean-shift algorithm involves using a classifier that differentiates the object from its surroundings. We adopt this classifier-based approach and propose an application of a classifier fusion technique within this classifier-based context in this work. We use two different classifiers, where one comes from a background modeling method, to generate the weight image and we calculate contributions of the classifiers dynamically using their confidences to generate a final weight image to be used in tracking. The contributions of the classifiers are calculated by using correlations between histograms of their weight images and histogram of a defined ideal weight image in the previous frame. We show with experiments that our dynamic combination scheme selects good contributions for classifiers for different cases and improves tracking accuracy significantly.

Keywords:
Mean-Shift, Object Tracking, Classifier Combination, Background Modeling.

1. Introduction

Object tracking in image sequences is an important problem in computer vision applications. Mean-shift tracking is a popular technique used for object tracking which models the color histogram of the tracked object in a frame and tries to shift the tracking window to a neighborhood area in the next frame, histogram of which is most similar to the modeled one. Although the original method relies on an iterative optimization of a similarity measure, the interpretation of the optimization target -called the weight image- has allowed extended techniques to be developed. Classifier based mean-shift tracking [1] is such an extension, where the weight image is generated by a classifier system which aims to perform binary classification for “object” and “outside” pixels in a local search window.

Another interesting problem considering moving objects is the background modeling problem. One of the popular techniques that is used to discriminate moving objects from a constant background in a long sequence of images is Stauffer-Grimson method [2] which relies on modeling the background pixels with mixtures of Gaussians. After a feasible time, generated background models for each pixel can be used to find the similarity of the pixel to the background at any time. One of the advantages of this method is its quick modeling of the background (usually ten frames) which allows even non-fixed cameras to benefit from this method during when they are temporarily fixed.

Handling the tracking problem within a classifier-based framework enables principled fusion of classifiers to be applied to the problem. In this work we combine two kinds of classifier approaches to generate a classifier combination system that is used to track a desired object over time. First kind of classifier is a binary classifier that differentiates the
tracked object from its surroundings. This kind of classifier uses the same logic with any classifier used in conventional classifier based mean-shift trackers which are actually binary classifiers that are trained on pixels that belong to the tracked object and its surrounding area in the local search window in the previous frame or frames. This classifier generates posterior probabilities of belonging to the tracked object for each pixel in the current frame.

The second kind of classifier is derived from the background model of the Stauffer-Grimson method, where we derive posterior probability of belonging to any foreground object for each pixel. Then with a novel proposed scheme, we dynamically calculate confidence values (i.e. weights) for each classifier and find a weighted combination of these classifiers to use in mean-shift tracking.

As mentioned above and reviewed on following sections, there exist previous works that rely on binary classifiers to generate weight images [1] or background models [2] used for tracking. However, in our work, we investigate disadvantages of using separate approaches and give examples of situations that individual methods fail. So we propose our novel approach, which combines both approaches and benefits from advantages of both. We propose a dynamic scheme that gives weights to individual classifiers and generates weighted combinations of them. Our dynamic weighting scheme can select proper weights for any situation since it uses a well-defined metric for classifier confidence. This combination scheme is a novel approach that leads to superior tracking performance.

This paper is organized as follows; in Section 2 we give a brief literature review of classifier based mean-shift tracking approach and in Section 3 we similarly review Stauffer-Grimson method and how it can be used to generate posterior probabilities for foreground objects. Section 4 contains information about our proposed combination scheme and finally in Section 5 results are presented as well with discussions and comments.

2. Classifier Based Mean-Shift Tracking

Mean-shift has been originally proposed for estimating the gradient of a density function [3] and has been used for feature space analysis [4]. After being introduced [5][6] into computer vision literature as an object tracking technique, mean-shift tracking has been a well known and referenced method for tracking non-rigid objects.

The original mean-shift tracking relies on modeling the grayscale histogram of the tracked object, which is called the target model \( q(b) : b \in \{1, ..., B\} \). Then on the next frame, the histogram of the search window (target candidate; \( p(b) \)) is generated and a similarity measure (Bhattacharyya coefficient) between the target model and candidate is defined:

\[
BC(p, q) = \sum_b \sqrt{q(b)p(b)}. \tag{1}
\]

To track the object, the search window should be iteratively shifted towards the direction where this similarity is at maximum. The direction and magnitude of the shift vector is calculated by optimizing the similarity measure which yields a weight value for each pixel which is calculated as:

\[
W_i = \sqrt{\frac{q(b(X_i))}{p(b(X_i))}}. \tag{2}
\]

Here \( b(X_i) \) gives the bin \( b \in \{1, ..., B\} \) for the feature vector (which is simply the grayscale value of the pixel in the original work) \( X_i \) of pixel \( i \). Equation (2) can be interpreted as a weight such that higher values are assigned to grayscale levels that are more frequent in the target model but less frequent in the target candidate. The set of \( W_i \)'s form the so-called weight image. After obtaining weight values of each pixel, the center of the search window is shifted to the center of weight of the weight image. This procedure is repeated after the first shift until there is no shift an example of which can be seen in Figure 2. Since tracking process shifts the search window towards the center of weight, the more the object is separated from the surrounding with higher values, the better the weight image is. A sample weight image can be seen in Figure 1.

The above interpretation of weight values has given rise to different approaches for generating the
weight image. One such approach \[7\] uses Red-Green-Blue (RGB) channels and their 49 different linear combinations for the frame. Discrimination of classes (i.e. object and outside) of pixels in the search window is calculated for each combination and top 5 of them are selected for the final tracking decision. The discrimination for each feature in the work is calculated as a measure of variance ratio of log-likelihoods of the class histograms which yields higher values for features where the two classes are better separated in the histogram such that; within class variances are low and overall variance is high. This interpretation is very similar to Fisher discriminating used for linear discriminant analysis.

Another similar work \[8\] again uses the better separating linear combinations of RGB channels where the calculation of separation relies on a measure of Bayes error of the class histograms. Bayes error exposes discriminative power better in multimodal distributions whereas variance ratio relies on separation of modes of distributions. This error rate yields lower values where two classes share less pixels, which means a separation of classes in the histogram. Derivation of Bayes error from the Bayesian decision theory relates the work to Bayesian classifiers.

Since the approaches rely on mappings that takes pixel features (either directly grayscale or linear combinations of RGB channels) as inputs and outputs weight values, a good classifier that discriminates two classes of object and outside can be used for generating these output weight values. A detailed analysis about classifier based mean-shift tracking is presented in the work \[1\] where weight values are generated by classifier ensembles. In this work, an ensemble of weak classifiers that distinguish object pixels from outer pixels are trained at each step. During training, each pixel contributes as a pair \(\{X_i, C_i\}\), where \(X_i\) is the feature vector (such as appearance features like grayscale intensity or RGB color values, texture features like Gabor coefficients, gradient histograms or local binary patterns) for pixel \(i\) and belongs to \(d\)-dimensional input feature space \(\mathbb{R}^d\). \(C_i\) denotes the label of the pixel as \(C_i \in \{-1, +1\}\). The weak classifiers \(h(X)\) map input pixels to labels such that:

\[
h(X) : \mathbb{R}^d \rightarrow \{-1, +1\}.\tag{3}
\]

\(h(X)\) classifiers try to separate the two classes in \(\mathbb{R}^d\). Most classifiers can output a score (posterior probability) \(c(X) \in [0, 1]\) where it takes higher values if the pixel is more likely to belong to the tracked object. The score values of each pixel is then used like a weight image \(W_t^1 = c(X_i)\) and mean-shift tracking is applied. In \[1\], not only color values of pixels but also texture values calculated from local gradients are used as features for classifiers. Although we only use color values in this
work, the explicit modeling of the object/outer classifier based weight image generation is the baseline for first kind of classifiers that we use in our classifier combination system. In this work we define the outer area such that it surrounds the object area, and together they form the larger search window, so bounds of the search window in the following tracking process are the bounds of the outer area as shown in Figure 3. We train our binary classifier using the pixels in the object area by assigning them class label +1 and the pixels in the outer area by assigning them class label −1. We train the classifier using the pixels of the frame $t$ (after tracking for that frame is completed) and apply them on the pixels within the same search window to get posterior probability values for each pixel at frame $t + 1$. However, the center of weight calculation is still done in the smaller object window which is shifted during mean-shift iterations to obtain its new location in frame $t + 1$. In contrast with the original mean-shift algorithm, the weight image is not updated after each mean-shift iteration since there is no histogram calculation, instead the weight image calculated in the larger search window is used directly.

The normalized values of the score values in $[0, 1]$ range allows us to interpret them as posterior probability values. Let us denote $C$ as a discrete random variable taking values $\{-1, +1\}$ such that $C_i = +1$ if the pixel $i$ belongs to the object and $C_i = -1$ if it belongs to the outer area. Then classification score of a pixel can be seen as the posterior probability of belonging to a moving object for that pixel:

$$p(C_i = +1|X_i) = c(X_i),$$

and for the same pixel the probability of belonging to outer area is then given as the complement of that:

$$p(C_i = -1|X_i) = 1 - c(X_i).$$

Using classifiers to generate the weight image helps to handle the tracking problem in a classifier based framework. As mentioned in the introduction this allows to apply classifier specific techniques to the problem like classifier fusion as proposed in this work. Although training a classifier may give the impression to add an extra computational load to the process, the advantage of using classifiers is that a classifier is trained and used to generate the pixel weight values only once for a frame, whereas on the original mean shift tracking weight values are recalculated at each iteration. Also using classifiers avoids using multi-dimensional feature histograms as shown in Equations (1) and (2) and easily allows using higher dimensional features.

In the original mean-shift algorithm, kernels can be used to emphasize the effect of central pixels in the tracking window by increased weighting of their contribution in the object histogram. This idea can also be used during classifier training by taking multiple samples from central pixels or by weighting central pixels more using classification algorithms that can work with weighted training data. However in this paper, we do not perform any kernel weighting since it did not result in improved results.

3. Background Modeling and Object Classifier

The background modeling proposed by Stauffer-Grimson [2], handles each pixel as a three dimensional (RGB) random vector $X$. The vectors are supposed to be generated by a mixture of Gaussians where basically each pixel’s RGB values are generated by an individual mixture. Gaussians allow to capture the deviations for a background object and since a pixel may contain different background objects more than one Gaussian is used. For example in a windy scene where a tree is observed, the leaves of the tree may move and in some pixels both the leave and the background sky is observed over time. Although the observed value of the pixel changes between two group of values over time, the values
are generated by fixed objects (i.e. tree and sky). This leads to the necessity of using more than one Gaussian to model the background of the pixel. The Gaussian mixture model (GMM) defines the likelihood of the random vector $X$ using a weighted sum of $K$ number of Gaussians:

$$p(X) = \sum_{j=1}^{K} w_j \mathcal{N}(X, \mu_j, \Sigma_j). \quad (6)$$

Parameters of the distributions ($\mu_j$ and $\Sigma_j$) are initialized with fixed values and updated dynamically [2] using the observations over time. Similarly weight of each Gaussian component ($w_j$) is also updated over time, such that weights for frequently observed mixtures are set to higher values. In any time, there exists a Gaussian mixture model for each pixel which can be used to infer the likelihood of belonging to the background for the current observation. This likelihood can be calculated using Equation (6), however not always all of the Gaussians in the mixtures is used, because some of the Gaussians in the mixtures may be generated by temporary foreground objects. The distinction between foreground and background Gaussians is determined by their weights such that weights are ordered from highest to lowest and first $B$ Gaussians cumulative sum of which are lower than a fixed threshold are taken as belonging to the background.

This distinction may be used for a binary classification as well as calculating the likelihood of belonging to background for the current observation of the pixel. The likelihood that the pixel $i$ is generated by the background ($BG$) is equal to the likelihood of the feature vector of the pixel under the mixture that contains only the background Gaussians:

$$p(X_i| i \in BG) = \sum_{j=1}^{B} \left( \frac{w_j}{\sum_{k=1}^{B} w_k} \right) \mathcal{N}(X, \mu_j, \Sigma_j). \quad (7)$$

Since this method models the background over time, it can adapt to changes and update the background model with respect to new observations. This is different from and superior to training separate models for background and foreground, since a pixel that is classified as foreground with this model may blend into background in time.

Equation (7) can be used to calculate probability that the observed pixel $i$ and its feature vector $X_i$ belongs to the background using the Bayes rule [3]:

$$p(C_i = -1|X_i) = \frac{p(X_i|C_i = -1)p(C_i = -1)}{p(X_i, C_i = -1) + p(X_i, C_i = +1)} \quad (8)$$

Here, $C_i$ is the same random variable, showing the object/background belonging of the pixel $i$, which is introduced in Equations (4) and (5). $p(C_i = -1)$ is the prior probability of belonging to background for the pixel at any time and $p(C_i = +1)$ is its complement; $1 - p(C_i = -1)$. Since $p(C_i = -1|X_i)$ is the probability that observed pixel value belongs to background, its complement gives the probability that the observed pixel value belongs to non-background—thus any moving object:

$$p(C_i = +1|X_i) = 1 - p(C_i = -1|X_i). \quad (9)$$

This probability value in Equation (9) is higher for the pixel values that are less likely to be generated by the past background structure of that pixel which we use as the second type of classifier that we use in our classifier combination system. Although class label $+1$ and probability value in Equation (9) in this context refers to any moving object rather than only tracked object, it is appropriate to use it since it gives a good discrimination between fixed background and tracked object—which is usually not fixed. Since we are interested in the pixels that are within the search window which is defined in Section 2 we use probability values only for those pixels within the search window. However parameters of the relevant background model for each pixel are updated on the whole image. Our idea is to use these posterior probabilities as another weight image $W^2_i = p(C_i = +1|X_i)$ to aid in our mean-shift tracking algorithm.

4. Classifier Combination for Mean-Shift Tracking

Combination of inputs of several sensors or combinations of outputs of different processings of the same sensor can be used to improve tracking accuracy. Like [7] which calculates a sum of different mean-shift vectors calculated from different features there exists methods in the literature like; feature selection by AdaBoost [10], combination of classifier outputs under a linear programming framework [11], combination of linear support vector machines [12], random forest classifiers [13] and
combination of thermo-visual and regular camera images to perform tracking.

As mentioned in Section 2, a classifier that performs binary (object and outer) classification by learning from samples of both classes can be trained and used to calculate probability of belonging to tracked object for each pixel in the search window. The classifier mentioned in Section 3 learns how to model the background using past data and infers the probability of a pixel belonging to foreground object. In our opinion distinct and heterogeneous nature of these two classifiers have complementary properties and using both of them is the most suitable approach.

When the tracked object passes near a fixed object that has similar appearance, the classifier that tries to separate the object and the outer area may fail to discriminate those two objects because of their similar color profile. However background classifier will continue to classify the fixed object as background and generate low probability of belonging to a moving object and will help to discriminate the tracked object and the fixed one. In contrast, when two moving objects come near, background classifier will generate high probability of belonging to a moving object for both. However this time the classifier that tries to separate the object and the outer area will discriminate two objects as long as their color profiles are different.

The quest at this point is how to combine the outputs of these two classifiers. Obviously a simple averaging is not feasible since in some situations either classifier may produce very unintended results. We come up with a dynamic scheme that calculates weights (λ) for the classifiers at each frame and uses these weights on the next frame to calculate their combined output.

\[ W_i = \lambda W_i^1 + (1 - \lambda) W_i^2. \]  

(10)

We handle the outputs of the classifiers as separate weight values for the weight images that can potentially be used for mean-shift tracking. Naturally their weighted combination calculated with Equation (10) is the final weight image that we use for our classifier combination based mean-shift tracking. The calculation of λ values rely on the idea that these weights are actually a measure of how confident each classifier is. Although it is not indicated explicitly in Equation (10), the combination parameter λ dynamically varies from frame to frame. To be able to regard weight images as grayscale images, we scale the weight values in the range [0, 1] to integral values in the range [0, 255].

To define the measure of confidence, we define what a good classifier means; an ideal classifier outputs posterior probability of one for the pixels that really belong to the tracked object and zero to others. On top of this definition we define two binned histograms that belong to weight images generated by these ideal classifiers such that one is concentrated around 255 and other around 0. The histograms can be seen in Figure 4 and Figure 5, where \( H_{\text{ideal}}^{\text{obj}} \) in Figure 4 belongs to object pixels in the ideal weight image and \( H_{\text{ideal}}^{\text{out}} \) in Figure 5 belongs to outer pixels in the ideal weight image. Grayscale values in these histograms are collected in eight bins, so each bin represents 256/8 = 32 consecutive grayscale levels.

After ideal histograms are defined, we now turn our attention to relate them to the output image generated by any classifier. Let a classifier c gener-
ate a weight image for a frame at time $t$ and $H_{obj}^c(t)$ be the histogram of the part of the weight image that belongs to the tracked object and $H_{out}^c(t)$ be the histogram of the outside pixels in the search window. The confidence of the classifier for the next frame at time $t+1$ may be inferred from the similarity of these two generated histograms to the ideal histograms. We can interpret histograms as discrete signals so we employ signal correlation coefficient which produces normalized values. The signal correlation coefficient between any two signals $g$ and $h$ is defined as:

$$
\rho(g, h) = \frac{\sum_b (g(b) - \mu_g)(h(b) - \mu_h)}{\sigma_g \sigma_h},
$$

where $b$ denotes bins of the histograms (of the weight images) and $\mu$ and $\sigma$ are mean and standard deviation of the relevant histogram. $\rho$ changes between $-1$ and $+1$ and higher values mean that two histograms are similar. We can use $\rho$, to measure the similarity of classifier generated histograms to ideal histograms and confidence $G$ of a classifier can be defined as the sum of correlations between two generated histograms (for object and outside) and two ideal histograms (again for object and outside):

$$
G_c(t+1) = \rho(H_{obj}^c(t), H_{obj}^{ideal}) + \rho(H_{out}^c(t), H_{out}^{ideal}).
$$

Since we use two classifiers ($c = 1, 2$), $\lambda$ in Equation (12) can be calculated as the ratio of the confidence of the two classifiers:

$$
\lambda = G_1 / (G_1 + G_2).
$$

In summary, after tracking is finished on a frame at time $t$, $\lambda$ value for the next frame (at time $t+1$) is calculated using Equation (12) and Equation (13). On the next frame, outputs of classifiers are combined using this $\lambda$ value as shown in Equation (10). An overall summary of the steps of the proposed classifier combination system is shown in Figure 6.

5. Experiments and Results

To demonstrate the proposed approach, we have selected some objects from the PETS 2001 database and performed tracking on them. At every frame we have trained or updated an AdaBoost classifier using pixels that belong to the object and outside pixels. Also we have trained a background classifier as mentioned in Section 3.

At each frame we have generated weight image values using outputs of the classifiers. We have applied classifier combination and found a final weight image which is the weighted combination of the previous ones. Contribution of each weight image to the final one is calculated using the proposed approach in Section 3.

For the mean-shift tracking we have used the Camshift extension, which adaptively resizes the search window at each frame. The amount of shift and resize of the search window is determined from the generated final weight image.

To compare results, we have performed tracking using outputs of both classifiers independently and their combined output as well. This way we have been able to see scenarios where independent classifiers fail and combined approach succeeds. As argued in Section 3, AdaBoost classifier failed in situations where tracked object passes near a fixed object with similar color value, an example of which can be seen in Figure 7. In that figure it can be clearly seen that the weight image generated by the AdaBoost classifier assigns high values to pixels belonging to the fixed object since its color is very similar to the tracked object. However background classifier can perform separation perfectly, since the pixels belonging to the fixed object currently have color values that are most likely to be generated by the background model.

In contrast, Figure 8 shows another situation where two moving objects come together. This time the background classifier fails to separate objects, because the pixels of both objects are assigned high values in the weight image since their current color values are not likely to be generated by the background model. However this time AdaBoost classifier can achieve much better separation, since color values of both objects are different and classifier

![Figure 6: Overall summary of the proposed tracking system; matching colors indicate related steps](image-url)
assigns low values to the pixels of the un-tracked object.

It can be seen that generated final weight images are very suitable for mean shift tracking in both of the situations. Since these final weight images are generated by weighted combinations of the other two, where their contributions are dynamically calculated, it can be said that the proposed approach succeeds to establish a proper classifier combination scheme.

To present the results of tracking for different objects, a measure \[16\] of tracking performance has been used. This measure defines the success ratio of the tracking at a single frame by:

\[
SR = \frac{A_r \cap A_t}{A_r \cup A_t},
\]

(14)

where \(A_r\) is the real area of the tracked object (the handmarked ground truth) and \(A_t\) is the area of the object found by the tracker. This ratio changes between 0 and 1 where 1 means that the tracker performed a perfect job and tracked the object by including all pixels of it and nothing more. On the contrary 0 means, the tracker failed to track the object altogether.

In Table 1 we present tracking success ratios of trackers for different objects, where we also show situations that tracker has failed to track the object until the end with a * mark. The trackers are initialized using ground truth rectangles of the objects after they are fully visible in the frame. Also, videos that show tracking process using separate classifiers and their fusion can be found on http://students.sabanciuniv.edu/~isaygint/tracking11.

In Table \[1\] it can be seen that in all of the situations where individual classifiers fail to track the object until the end, the tracker obtained with our fusion approach can track the object successfully. In addition, even in the cases where both classifiers can succeed individually, the measure presented in Equation (14) has always higher values with the tracker obtained with our fusion approach.
Table 1: Tracking success ratios for different object and trackers

| Object       | AdaBoost Only | Backgr. Only | Classifier Comb. |
|--------------|---------------|--------------|------------------|
| Red Coat     | 9.62*         | 42.05*       | 52.85            |
| Female       |               |              |                  |
| Blue Car     | 64.34         | 58.94        | 75.91            |
| Peugeot      | 65.79         | 46.04        | 74.72            |
| White Van    |               |              |                  |
| Left Entry   | 23.62*        | 33.51        | 47.60            |
| Male         |               |              |                  |
| Female Blue  | 1.93*         | 36.66        | 52.29            |
| Skirt        |               |              |                  |

6. Conclusion

As can be seen from numeric results, weight images generated using the classifier combination scheme give the best tracking results and even succeed in situations where individual classifiers fail when they are used alone. The reasons why individual classifiers may fail and how the dynamic combination scheme can overcome these can also be seen on the supplied sample figures and in videos. Dynamic and non-parametric calculation of classifier contributions and its positive effect on tracking emphasizes the robustness of the proposed scheme.

In this work we have employed only RGB color values of the pixels, however more complex but better classifiers may be achieved by employing other scene information such as local texture. Additionally the search window has been taken as a simple rectangular area in this work, however a search window that is adaptive to the actual shape of the object may also increase the recognition accuracy.

References

[1] S. Avidan, Ensemble tracking, in: Computer Vision and Pattern Recognition, 2005. IEEE Computer Society Conference on, 2005, pp. 494–501.
[2] S. C., G. W.E.L., Adaptive background mixture models for real-time tracking, in: Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on, 2002.
[3] K. Fukunaga, L. Hostetler, The estimation of the gradient of a density function, with applications in pattern recognition, Information Theory, IEEE Transactions on 21 (1) (1975) 32 – 40.
[4] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Trans. Pattern Anal. Mach. Intell. 17 (1995) 790–799.
[5] D. Comaniciu, V. Ramesh, P. Meer, Real-time tracking of non-rigid objects using mean shift, 2000, pp. 142–149.
[6] G. R. Bradski, Computer Vision Face Tracking For Use in a Perceptual User Interface, Intel Technology Journal (Q2).
[7] R. Collins, Y. Liu, M. Leordeanu, On-line selection of discriminative tracking features, IEEE Transaction on Pattern Analysis and Machine Intelligence 27 (1) (2005) 1631 – 1643.
[8] D. Liang, Q. Huang, W. Gao, H. Yao, Online selection of discriminative features using bayes error rate for visual tracking, in: PCM’06, 2006, pp. 547–555.
[9] D. Turdu, H. Erdogan, Improving gaussian mixture model based adaptive background modeling using hysteresis thresholding, in: Signal Processing and Communications Applications, 2007. SIU 2007. IEEE 15th, 2007, pp. 1 –4.
[10] Y.-J. Yeh, C.-T. Hsu, Online selection of tracking features using adaboost, IEEE Trans. Cir. and Sys. for Video Technol. 19 (2009) 442–446.
[11] M. Grabner, C. Zach, H. Bischof, Efficient tracking as linear program on weak binary classifiers, in: Proceedings of the 30th DAGM symposium on Pattern Recognition, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 102–111.
[12] M. Tian, W. Zhang, F. Liu, On-line ensemble svm for robust object tracking, in: Proceedings of the 8th Asian conference on Computer vision - Volume Part I, ACCV’07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 355–364.
[13] C. Leistner, M. Godec, A. Saffari, H. Bischof, Online multi-view forests for tracking, in: Proceedings of the 32nd DAGM conference on Pattern recognition, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 493–502.
[14] C. Conaire, N. E. OConnor, A. Smeaton, Themovisual feature fusion for object tracking using multiple spatiogram trackers, Mach. Vision Appl. 19 (2008) 483–494.
[15] http://www.cvg.cs.rdg.ac.uk/pets2001/, Second IEEE International Workshop on Performance Evaluation of Tracking and Surveillance.
[16] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Kozhova, J. Zhang, Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2009) 319–336.