International Scholarly Research Network
ISRN Signal Processing
Volume 2011, Article ID 173176, 15 pages
doi:10.5402/2011/173176

Research Article

Object Modelling and Tracking in Videos via Multidimensional Features

Zhuhan Jiang

School of Computing and Mathematics, University of Western Sydney, NSW 1797, Australia

Correspondence should be addressed to Zhuhan Jiang, z.jiang@scm.uws.edu.au

Received 30 November 2010; Accepted 5 January 2011

Academic Editor: C. S. Lin

We propose to model a tracked object in a video sequence by locating a list of object features that are ranked according to their ability to differentiate against the image background. The Bayesian inference is utilised to derive the probabilistic location of the object in the current frame, with the prior being approximated from the previous frame and the posterior achieved via the current pixel distribution of the object. Consideration has also been made to a number of relevant aspects of object tracking including multidimensional features and the mixture of colours, textures, and object motion. The experiment of the proposed method on the video sequences has been conducted and has shown its effectiveness in capturing the target in a moving background and with nonrigid object motion.

1. Introduction

Tracking an object within a video sequence is an important task in computer vision and has found applications in a variety of fields such as surveillance, machine intelligence, and even medical treatment. Object modelling is the capstone of tracking technologies, and different modelling methodologies typically lead to different scopes and capabilities. The two main application categories closely related to object modelling are object detection and object tracking. In object detection, the averaging method [1] is the most straightforward to extract the background for the frames, and thus the object, for a relatively static background scene, while other methods may resort to using a mixture of Gaussians to adaptively model each pixel [2], the kernel density [3] to statistically represent the background, or the local correlation maps [4] to exclude outliers in estimated motion vectors.

The traditional template matching [5–7] tries to match the whole object region directly with another in a new frame and locate the new object position there that achieves minimum and acceptable matching errors. Kalman filer was utilised [8] to make the template matching also adaptive to object occlusions. Since direct template matching is in general of high computational cost and has its own restrictions, contour approaches consequently gained considerable attention for their advantages on dealing with objects of deforming shapes and on lessening the computational complexity. In particular, the snake model first introduced in [9] has been extended into many variants [10–14] of the active contour model. They extract the object contour based on certain contour deforming criteria such as the minimisation of an energy function. More recently, kernel functions have been widely used [15–17] in estimating the likelihood of a given pixel to be on an object of interest or the probabilistic differences between the candidate and the target object regions, while multihypothesis methods and Bayesian inference [10, 18, 19] have been employed to propagate object contours or the like into the future frames. The condensation method in [11], for instance, propagates the contour of the tracked object in the framework of a posterior probability. In the harsh environment such as tracking a camouflaged object [20], motion features [10] may have to be additionally considered.

The template matching, though intuitive, has limitation on processing objects of deformable shapes even though certain deformable template methods have been proposed for the performance improvement. The contour approach,

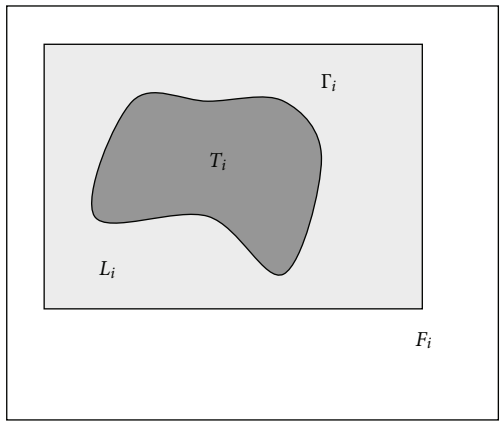


Figure 1: Boundary Γ_i, object T_i, and local window L_i in a frame F_i.

on the other end, has to rely heavily on the presence of strong object boundary. The kernel approach, as its own downside, often requires a heavier computational load. For all these different algorithms, there still need to be additional sanity assumptions in the form of such as rigid movement, limited illumination change, as well as the static background. Since significant features such as strong edges and distinctive textures are known to be indicative of the presence of the tracked object, it is anticipated that any feature that differentiates the object from the background will be good choice for the tracking purpose. Such features may be selected from a given pool [21] or as a result of enhancing the brighter or darker aspect of the object by the wavelet filters [22]. In most applications, however, essentially one feature [15–20, 22] is utilised at a time, and this may be largely due to the curse of multidimensionality. In this connection, our aim is to devise a framework that would allow us to incorporate several features at the same time in locating the object from the background, simplify the modelling representation, and then direct the tracking within the framework of Bayesian inference.

The main purpose of this work is to model the tracked object of nonrigid shape by one or several object features, such as colours and textures, so as to more accurately model the object and robustly resist the environmental disturbances and noises. This modelling will be based on multidimensional cubes of features, rather than one-dimensional bins of one feature, and will estimate the object location by the probabilities of its presence. Unlike the one dimensional case, the number of feature cubes could increase very rapidly in multidimensions. However, we found that nullifying "insignificant" cubes will still maintain the tracking logic while reducing the computational load. The location of the object will be represented in terms of probabilities and will be estimated in a Bayesian framework. In this regard, one could first approximate the local object and background densities in terms of the features such as colours and textures and then derive with the Bayesian inference the object probability that refers to the probability of a pixel belonging to the object.

This work is organised as the following. In Section 2, we establish the framework for the use multidimensional feature cubes and the Bayesian inference. Different ways of combining several features are also inspected there. We then look into a few different types of features in Section 3, including colours, textures, and the motion. A simplistic approach to better synchronise frame background is also explored. We then devise in Section 4 a scheme to select features of dominance to model the object. Section 5 then develops a method of shape consolidation and extraction so that an extracted object with background noises can be effectively enhanced. The experimental results are subsequently reported in Section 6 for a variety of video sequences. Finally Section 7 contains a short conclusion.

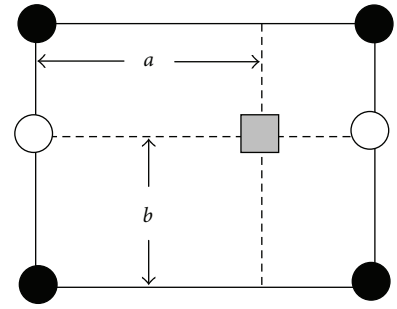


Figure 2: Interpolation via neighbour pixels.

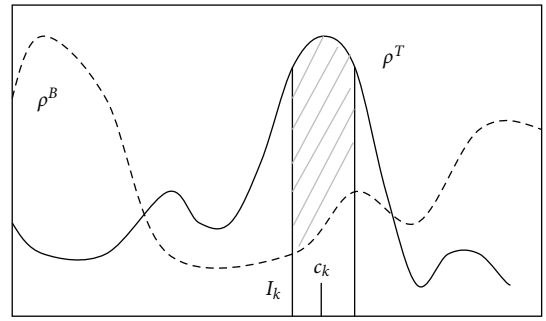


Figure 3: Density differences for the object T and the background B.

2. Multidimensional Feature Space

A video sequence $\{F_i\}_{i\geq 0}$ consists of a series of frames, and F_i thus represents a frame of $m \times n$ pixels. A general frame F_i typically contains a tracked object of interest, T_i, whose contour boundary is denoted by Γ_i, within a local window L_i, see Figure 1. For a given single feature, its pixel histogram $h : [a, b] \to \mathbb{N}$, where $\mathbb{N}$ is the set of natural numbers, $[a, b]$ is the range of the feature values, and $h(x)$ is the frequency of the value x, can be normalised to $h : I \to \mathbb{N}$ on the unit interval $I = [0, 1]$, or further normalised to a distribution density $\rho(x) = h(x)/\int_0^1 h(x)dx$. For a d-dimensional feature, such as a mixture of colours and textures, its pixel histogram will similarly take the form

$$h : D \longrightarrow \mathbb{N}, \quad D \equiv [a_1, b_1] \times \cdots \times [a_d, b_d], \qquad (1)$$

where $[a_k, b_k]$ represents the scope of the kth feature value.

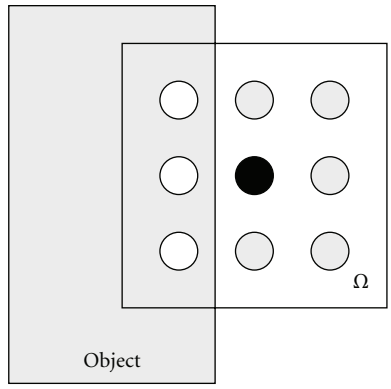


Figure 4: Border pixels in image enhancement.

If S and T are two image components, we denote by $S \subseteq T$ that each image pixel of image component S is also part of image component T, and by $T \setminus S$ the remaining part of T after S is taken from it. For a given $T \subseteq L \subseteq F$ with $B = L \setminus T$ like those in Figure 1, we can thus define the histogram h^T, h^B, and h^{B*} on them, respectively. Likewise, we can also define the densities $\rho^T(x)$ for the target object, $\rho^B(x)$ for the background within a local window L, and $\rho^{B*}(x)$ for the background of the whole frame F.

In order to minimise the modelling data and reduce the complexity, we first extend the one-dimensional notion of "bins" for the colours to multidimensional "cubes" for features. Basically, all the feature values will be put into certain cubes in such a way that (i) any two feature values belonging to the same cube are "close" in terms of the physical nature with which the feature is defined and (ii) all cubes together form a disjoint partition and are completely ordered. For this purpose, we partition each feature domain $[a_k, b_k]$ into the union of intervals of width $\omega_k > 0$, and the width of kth dimension of a typical d-dimensional cube is thus ω_k. The choice of the weight factor ω_k allows different feature values to impact modelling at different scales. For any two feature vectors $x, y \in \overline{D}$, the union of all the cubes, we define their distance via the infinity norm

$$||x - y|| = \max_{1 \le i \le d} \left(\frac{|x_i - y_i|}{\omega_i} \right). \tag{2}$$

We note that the common Euclidean metric $\|\cdot\|_2$ will not be able to allow us easily partition the feature space. If the kth cube has a centre feature vector $c = c_k$, then that cube is defined by

$$I_k \equiv \left\{ (x_1, \ldots, x_d) : -\frac{\omega_i}{2} \le x_i - c_i < \frac{\omega_i}{2},\ \forall 1 \le i \le d \right\}, \tag{3}$$

and obviously $\|x - c\| \le 1/2$ and $\cup_{1 \le k \le N} I_k \subseteq D \subseteq \overline{D}$, where N is the total number of the cubes. For any image component T, we can determine the histogram h^T and likewise determine h^B for the background. These histograms essentially represent the probabilities of a given pixel falling on the object T or the background B according to their feature values. If the features are chosen to be sufficiently discriminative, then they may be used to locate or track the given object in different video frames through the Bayesian inference which will be explained in the followings.

2.1. Bayesian Inference. The Bayesian inference is a standard statistical technique that involves collecting evidence E that is meant to be consistent or inconsistent with a given hypothesis H, and as evidences accumulate, the degree of belief of the hypothesis changes. Bayesian inference uses a numerical estimate of the degree of belief in a hypothesis before evidence has been observed and calculates a numerical estimate of the degree of belief in the hypothesis after evidence has been observed. Bayes' theorem thus states $p(H \mid E) = p(E \mid H)p(H)/p(E)$, where $p(H \mid E)$ is the posterior probability of H given E, an improvement on the originally estimated prior probability $p(H)$. In a more general case of having $n + 1$ mutually exclusive hypotheses H_i, the prior probabilities $p(H_i)$ can be improved into the posterior probability $p(H_o \mid E)$ based on a set of additionally observed evidence via

$$p(H_o \mid E) = \frac{p(E \mid H_o)p(H_o)}{\sum_{i=0}^{n} p(E \mid H_i)p(H_i)}, \tag{4}$$

where $p(E \mid H_i)$ is the likelihood of the hypothesis H_i under the observed evidence E. For our specific problem, (4) takes the form

$$p^t(T \mid x) = \frac{p^t(x \mid T)p^t(T)}{p^t(x \mid T)p^t(T) + p^t(x \mid B)p^t(B)}, \tag{5}$$

where $p^t(T)$ and $p^t(B)$ are the prior probabilities estimated prior to observing the actual pixel value, $p^t(x \mid B)$ denotes the probability of having the value x by a pixel on the background B, and $p^t(T \mid x)$ denotes the probability of being part of the object T for a pixel of value x. Since the total object area remains somewhat constant across the nearby frames, we will assume a constant $p^t(T)/p^t(B)$ and use the initial $p^0(T)$ and $p^0(B)$ for all the neighbouring frames. Since the object and background densities are very close across consecutive frames, $p^{t-1}(x \mid T)$ and $p^{t-1}(x \mid B)$ or the like can be made use of to approximate those at t in (5). Alternatively, if the feature values for the background are simple and distinctive, then one can estimate the probability of a pixel being on the background, as opposed to being on the object. Such a probability can be estimated similar to (5) via

$$p^t(B \mid x) = \frac{p^t(x \mid B)p^t(B)}{p^t(x \mid T)p^t(T) + p^t(x \mid B)p^t(B)}, \tag{6}$$

from which $p^t(T \mid x)$ can be easily estimated.

For the calculation of $p(x \mid T)$, for instance, it can be derived from the histograms of the pixels on the object T. In the case of independent feature variables $\{x_i\}$, the probability $p(x \mid T)$ becomes separable

$$p(x \mid T) = p(x_1 \mid T) \cdots p(x_d \mid T), \tag{7}$$

and the probabilities $p(x_i \mid T)$ can be calculated individually. Even if the feature variables are implicitly correlated to

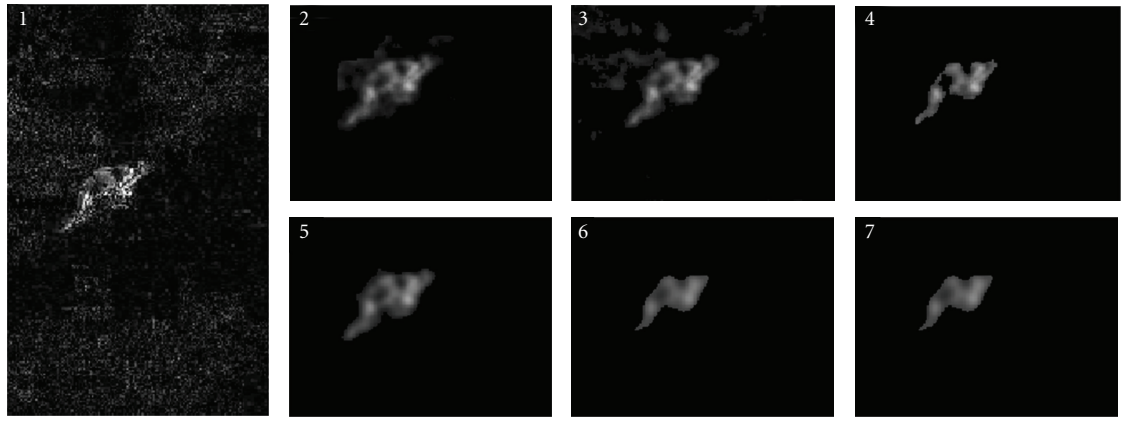


Figure 5: Effect of different enhancements.

Figure 6: Local partition of the 1st video frame.

certain extent, it is still possible to use the separable form (7) to approximate the $p(x \mid T)$, and this gives more flexibility in selecting the feature variables. In most cases, all the variables x_i correspond to the *affirming features* in that a larger value indicates a higher probability on the object T. However, if a particular x_k is in fact a *negating* feature as opposed to the affirming one, then $p(x_k \mid T)$ can be replaced by $\overline{p}(x_k \mid T) = 1 - p(x_k \mid T)$, or even estimated somewhat differently to adjust the significance of that feature variable.

2.2. Incorporation with Other Features. When an object is being tracked in a video sequence, it is natural to expect that the more features one monitors the better and more robust the tracking outcome, particularly when the prime features of colours and textures happen to be relatively weak in a particular shot. These additional features could be in the form of colours and textures, motion information, the physical restriction of a rigid body, or a domain mask on the object shape. These will obviously depend on the individual application scenarios.

In the case of extracting additional motion features, one can represent [10] the probability density function $p_D(d)$ of the observed interframe difference $D^t(s) = I^t(s) - I^{t-1}(s)$ in terms of a static background density $p_S(d)$ and a conditional mobile density $p_M(d)$, where S and M refer to the static and motion components, respectively, and d denotes the difference value in intensity. In other words, one has the mixed model $p_D(d) = P_S p_S(d) + P_M p_M(d)$, and the model parameters P_S, P_M and Θ can be estimated by maximising the joint likelihood, $\Pi_{d \in D} p_D(d \mid P, \Theta)$, through the maximum likelihood principle or the method of expectation maximization [23], where $d \in D$ means d is to go through $D^t(s)$ for all the pixel positions s there. We note that the motion frames D^t can be applied a formula similar to (5) for the estimation. For any pixel of intensity x, if the corresponding intensity difference in D^t is d, then the motion data can improve [10] the estimation to $\overline{p}^t(T \mid x)$ via

$$\overline{p}^t(T \mid x) = \frac{p^t(T \mid x) p^t(d \mid M) p^t(M)}{p^t(d \mid M) p^t(M) + p^t(d \mid S) p^t(S)}. \quad (8)$$

In a particular tracking application, additional features or geometric restrictions may also be utilised to refine the tracking accuracy. We note that although $\mathrm{abs}(D^t(s))$ does not fit mixed Gaussian models mentioned above, the absolute frame difference often works equally well directly.

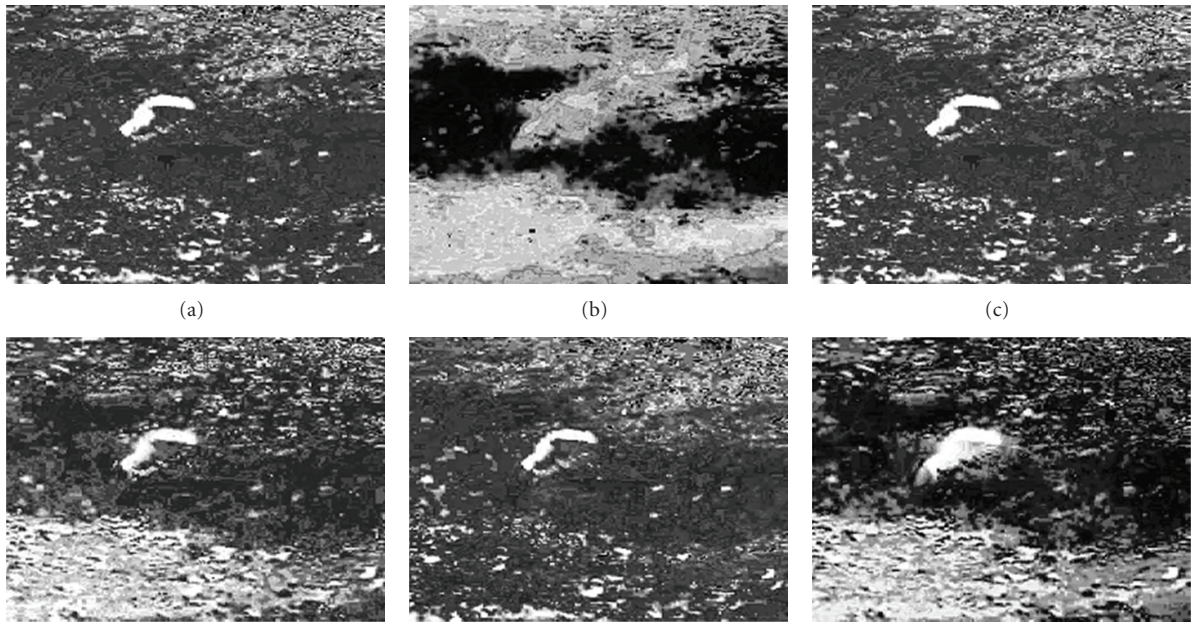
In general, when there are two separate probability maps, $p(x)$ and $q(x)$, derived for different features or from different methodologies, one can also combine the two maps in different ways. Suppose that after thresholding and enhancement, $p(x)$ and $q(x)$ lead to the object areas P and Q, respectively, we can also incorporate the relative physical distances to synthesise them via

$$r(s) = \frac{\alpha p(x)}{1 + d_a(s, Q)} + \frac{\beta q(x)}{1 + d_a(s, P)} + \frac{\gamma p(x) q(x)}{1 + d_a(s, P \cap Q)}, \quad (9)$$

where α, β, and γ are the coupling constants, $x = x(s)$ is the pixel value at position s, and $d_a(s, P) = \eta[\sum_{s' \in P} \|s - s'\|]/|P|$ for a nonempty set P and $d_a(s, \varnothing) = \infty$ for the empty set $\varnothing$. For calculational simplicity, one may sometimes replace $d_a(s, P)$, the average distance of s to P, simply by $\eta \|s - c(P)\|$, where $c(P)$ is the centre of P and η is a parameter that adjusts the coupling strength. We note that the usual spatial-independent combination $\alpha p(x) + \beta q(x)$, or $p(x)q(x)$, $[1 - (1 - p(x))(1 - q(x))]$, each corresponding to a different

(a) (b) (c)

Figure 7: Images in HSV colours.



(a) (b) (c)

(d) (e) (f)

Figure 8: Posterior probability maps for HSV and RGB colours for the 1st frame image.

approach to combine the probability maps, are all special cases of the extended $r(s)$ in the above.

3. Some Variety of Features

Colours and textures are obviously those most prominent local features one usually encounters when locating an object from a background environment. There are different colour spaces, offering a variety of possible colour features. For simplicity, we will consider only the RGB colour space and the HSV colour space, as the rest are very much similar. While most colour based approaches are template based and subimages are searched or matched blockwise [24], our proposed scheme is largely pixel based and probabilistic.

The literature on the use of texture in object tracking is quite scarce compared with that on the use of colours. One relatively recent work [25] presented an efficient approach that is based on the authors' customer-made texture of local binary patterns (LBP). Their work is typically applicable to video shots of a *static* camera, and the LBPs there served as the bases for the multimodal structure for the identification of the object motion.

At a given image pixel, the texture around the pixel is determined by the pixel values in the neighbourhood. There can be many different ways to define a local texture, ranging from mean, standard deviation, to LBP or

$$\Phi = \sqrt[p]{\sum_{1\le k\le m} r_k^q \sum_{x\in R_k} \frac{|x-\mu_k|^p}{|R_k|}}, \tag{10}$$

where $\cup_k R_k$ is a partition of the neighbourhood of pixel position, R_k is a concentric annulus, μ_k is the mean of the disk $\overline{R}_k \equiv \cup_{i\le k} R_i$, and r_k is the radius of the annulus R_k. In the simplest case of $p = 2$, $q = 0$ and $m = 1$, Φ is just the standard deviation.

To avoid the participation of the noises in the calculation of textures, we can use the *correlated texture* which is defined

Figure 9: Extracted object and the shape fittings.

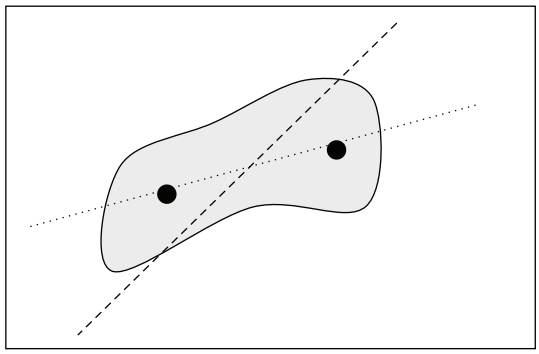


Figure 10: Fitting with a base shape.

through a set of correlated feature values such as a set of selected colour values. For instance, if $\{I_k\}$ is a set of ranges of leading colours for the object, then we calculate the texture property, such as the mean, the standard deviation and the skew, based on only these selected colours, correlated through the common object, by ignoring all the other colours. The determination of the feature values and their significance may vary slightly along with the size of the local window L_i. Ideally, L_i should be large enough to contain the object in the next frame, and a more representative local window can reduce noises for pixels at a distance from the object.

Although motion is generally considered a very strong feature, if present, the extraction of object motion is not easy when the camera is not steady, let alone when the camera is meant to move around. Hence, we limit ourselves here to extracting motion features only from frames that exhibit steady or very slow-varying background. Suppose that there are two frames of $m \times n$ pixels $F = \{f(s) : s \in \mathbb{Z}^2, s = (s_1, s_2),\ 0 \le s_1 \le m,\ 0 \le s_2 \le n\}$ and likewise $F' = \{f'(s)\}$, their difference will highlight the moving object if F and F' share the same background. By extracting the mixed models for the difference frame, an additional feature of motion can be incorporated as in (8). It is also possible to make use of the difference frame directly, at the expense of involving with physical pixel locations explicitly. This time it serves more like a probabilistic mask for the object.

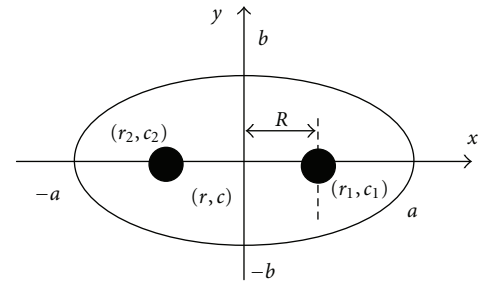


Figure 11: Centres in an ellipse.

In the case of an unsteady camera, one can improve the quality of the difference frame by shifting pixel positions slightly. By shifting the row and column position of the frame F by a and b, respectively, becoming frame $F'' = \{f(s + \Delta)\}$ for $\Delta = (a, b)$, the difference frame between F' and F'' will minimise locally $[\sum_{s \in D^o} |f(s + \Delta) - f'(s)|]/[(m - |a|)(n - |b|)]$, where $|a| \ll m$, $|b| \ll n$ and D^o denotes the overlapped area of F' and F''. When the shifts a and b are not all integers, then an interpolation on the neighbouring pixel values is needed; see Figure 2. for the case of $|a| \le 1$ and $|b| \le 1$, where the pixel value at the point $(r + a, c + b)$ marked by a small square is to be interpolated from its four neighbour pixels drawn in solid disks. The interpolation is done via the two pixels marked by empty circles which are in turn derived from the interpolation of their respective two neighbours in solid disks, and the formula reads

$$\begin{aligned} f(r + a, c + b) = {} & |ab| f(r + \operatorname{sgn}(a), c + \operatorname{sgn}(b)) \\ & + |b|(1 - |a|) f(r, c + \operatorname{sgn}(b)) \\ & + |a|(1 - |b|) f(r + \operatorname{sgn}(a), c) \\ & + (1 - |a|)(1 - |b|) f(x, y), \end{aligned} \tag{11}$$

where the sign function is defined by $\operatorname{sgn}(x) = 1$ if $x > 0$, $= -1$ if $x < 0$, and $=0$ if $x = 0$. When the background is well preserved locally even though it changes across frames due to camera panning, it is often possible to generically synchronise the consecutive frames so that the difference of the synchronised frames can be used to extract the motion components. Given two frame images $F = \{f(s)\}$ and $F' = \{f'(s)\}$, the initial synchronisation displacement (r, c), and an initial navigation step size χ, we compare the synchronisation error $[\sum_{s \in D^o} |f(s) - f'(s + \Delta s)|]/(\text{intersection area of } F \text{ and } F')$ for $\Delta s = (r, c) + (\Delta r, \Delta s)$ with Δr and Δs being $\pm\chi$ or 0. The displacement (r, c) then updates to $(r^*, c^*) \equiv (r, c) + \Delta s$ for the Δs that minimises the synchronisation error. If (r^*, c^*) differs from (r, c), then the process repeats again. Otherwise, reduce χ by a half and also repeat the process again. The algorithm terminates when the synchronisation displacement reaches the required precision. Once the difference of the synchronised frames is derived, it becomes also possible to use mixed model to extract the motion distribution or simply use the difference frame directly. Just as in the case of meerkat sequence Figure 24, we expect that pixel displacement within a single pixel

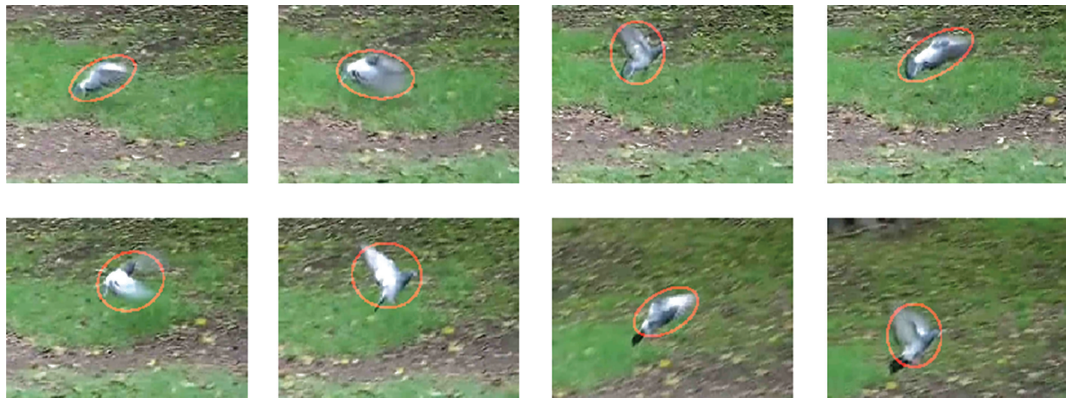


Figure 12: Bird object tracked in the base shape.

Figure 13: Bird object tracked in free form.

is in general not so significant in extracting the motion component. Even though the above simplistic algorithm can stop at a local minima, it is surprising that it works for a great majority of frames even for bird video Figure 12 (red colour selected), where the camera panning shifts the background in 10 s of pixels.

4. Choice of Dominant Features

Suppose that $\{I_i\}_{i\in I}$ is a partition of the feature space, with c_i as its centre. Then, an image component S is said to contain feature vector $c_i \in I_i$ if for a threshold $\tau_{\min} > 0$ one has

$$\int_{I_i} \rho^S(x)dx \geq \|I_i\| \cdot \tau_{\min}. \tag{12}$$

For a multidimensional feature space, there are typically a great number of feature cubes I_i to consider, resulting in potential computational complexity. While for the one-dimensional feature space, we can just directly use the whole set of cubes, or bins in this case, to conduct the object tracking, it is desirable to significantly reduce the number of cubes from the inspection horizon.

How can we dynamically select the right feature cubes so that an object of interest can be more effectively and efficiently tracked within a video sequence? Suppose that a general pixel has a d-dimensional feature vector $x = (x^{(1)}, x^{(2)}, \ldots, x^{(d)})$, then for each single feature element $x^{(j)}$, we locate the feature cubes that best differentiate the tracked object T against the local background $B = L \setminus T$. More precisely, we could locate the feature cubes I_k such that the following regularity conditions:

$$\gamma\Delta_k^{(j)} \geq |I_i|\eta, \quad \Delta_k^{(j)} \equiv \int_{I_k} \left(\rho^T(x) - \rho^B(x)\right)dx, \tag{13a}$$

$$\frac{\int_{I_k} \rho^T(x)dx}{\int_{I_k} \rho^{B*}(x)dx} \geq \tau > 0, \tag{13b}$$

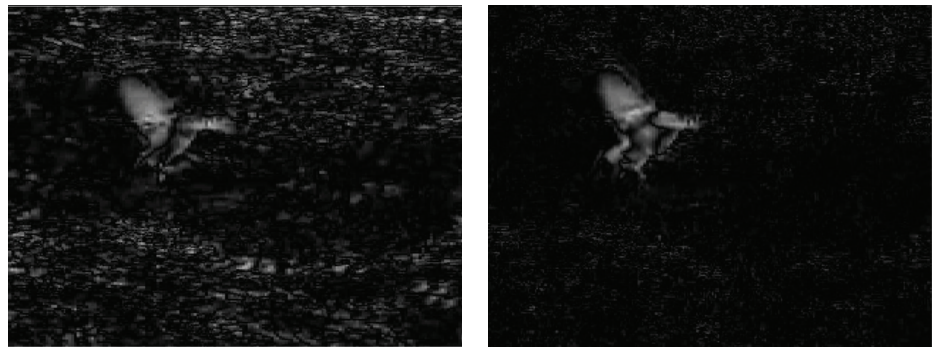
$$\frac{\Delta_k^{(j)}}{\int_{I_k} \rho^T(x)dx} \geq \tau > 0 \quad \text{for } \gamma = 1, \tag{13c}$$

$$\frac{\Delta_k^{(j)}}{\int_{I_k} \rho^B(x)dx} \geq \tau > 0 \quad \text{for } \gamma = -1, \tag{13d}$$

$$\gamma\left(\rho^T(x) - \rho^B(x)\right) \geq 0, \quad \forall x \in I_k \tag{13e}$$

hold for a constant $\gamma = \pm 1$, see Figure 3. Intuitively, (13a) requires that the chosen features are sufficiently discriminative for the object and the background, (13b) requires that the object area is not diminishing, and (13c)–(13e) require that the feature difference over an interval of interest should be sufficiently observable with respect to the object or to the background. We note that the threshold τ in (13b)–(13d) is to ensure that the feature cube I_k does not get overwhelmed by the frame background. For the general case of feature vectors, we can apply this scheme to each feature component $x^{(j)}$ so as to collect a set of cubes of differentiating features $\{I^{(j)}\}$ with $I^{(j)} = \cup_{k\in K^{(j)}} I_k^{(j)}$, or

$$\left\{I_k^{(j)}\right\}_{k\in K^{(j)}}, \quad j = 1, \ldots, d, \tag{14}$$



(a) (b)

Figure 14: Difference frames.

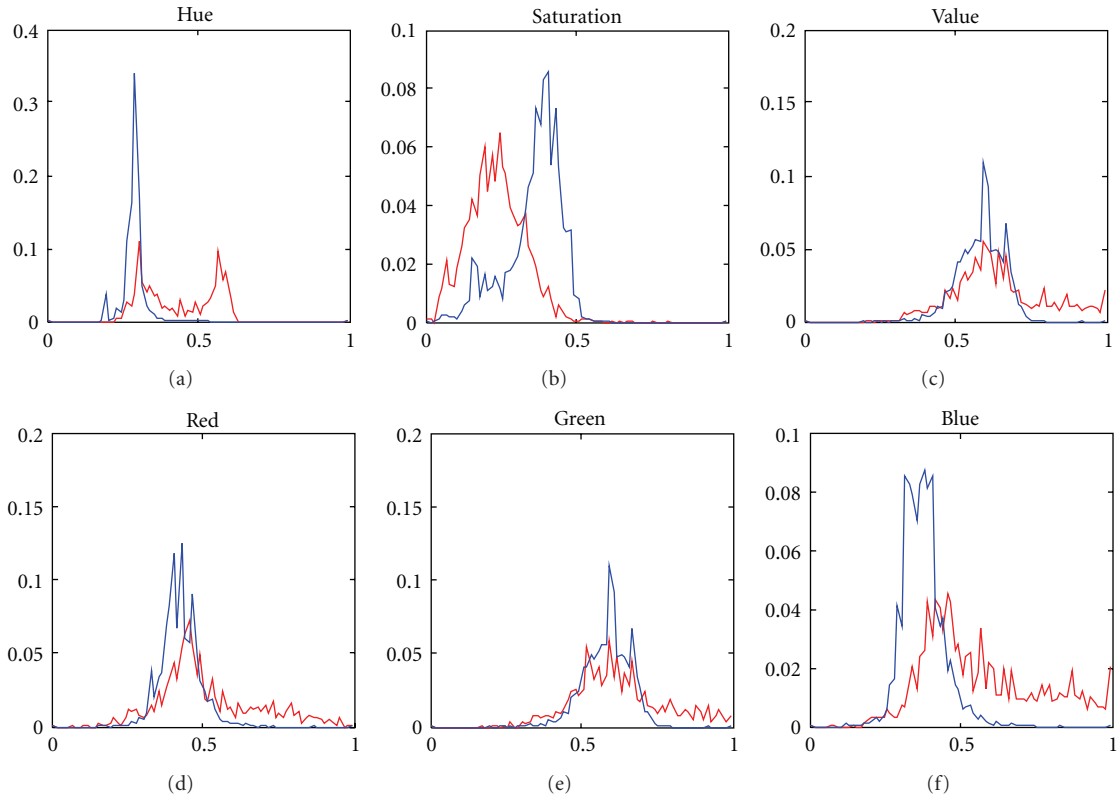


Figure 15: Histograms for object (red) and background (blue) in HSV and RGB colours.

where $K^{(j)}$ contains the indices for the chosen cubes. For convenience, we will refer to such a collection $I^{(j)}$ of cubes as the *discriminating band* (DB) for the (j)th feature component and use negative index k to denote the case of dominant background feature for $\gamma = -1$. The main reason that we explore all available features instead of just one is that one feature component, such as red in RGB, may not suffice to tell apart of parts of the tracked object T. However, a direct composition of all the feature components may lead to unnecessarily duplicated cubes in terms of search effect and to missing some of those less differentiating features which are nonetheless responsible for telling apart other areas of the object from the background. Moreover, different feature space may exhibit different power of distinguishing the object and the background.

Let S be an image component, we denote by

$$d(s, S) = \min\{||s - s'|| : s' \in S\}, \tag{15}$$

the physical distance of the pixel position s to the component S. Let M be a pixel mask of same size as S, then we denote by $S \otimes M$ the set of those masked pixels in S. That is, for any $S_{i,j} \in S$, $S_{i,j} \in (S \otimes M)$ if $M_{i,j} \neq 0$ and $S_{i,j} \notin (S \otimes M)$ if $M_{i,j} = 0$. Then, we can use the following procedure to obtain those cubes of tracking features.

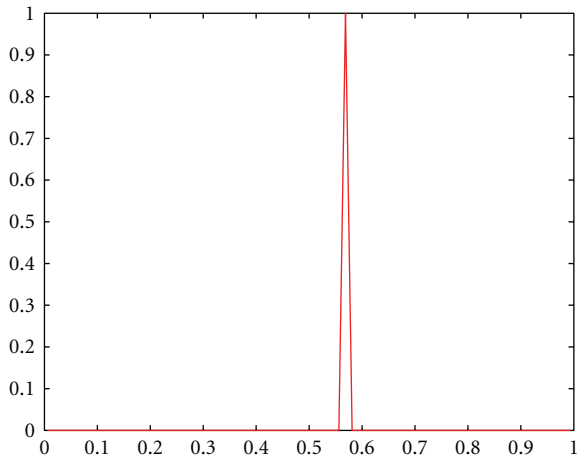


Figure 16: Single bin histogram in hue.

(1) For a given frame F and an object $T \subset F$, select a local window L such that $T \subset L$ and $\{s \in F : d(s,T) \leq \delta\} \subset F$ for a constant distance $\delta > 0$. Initialise a mask M for F.

(2) Choose the next unprocessed feature component, the (j)th feature component $x^{(j)}$. For the masked image $S \otimes M$, calculate the normalised histograms and locate the corresponding discriminating band $I^{(j)}$.

(3) Repeat step (2) if current feature component does not induce a good DB $I^{(j)}$, or repeat (2) after converting the image into another feature space. Otherwise, go to step (4).

(4) Stop, if all feature components have been considered or if current collection of selected feature cubes already well covered the object T. Otherwise, go to step (5).

(5) Update the mask M by marking off those pixels contributing to the features in the DB $I^{(j)}$, and then go back to step (2).

We note that larger distance between the density peaks of the object T and background B, and larger value in such as $|\Delta_k^{(j)}|$ in (13a)–(13e) typically correspond to a better tracking performance. We also note that the masking in the above can be made optional for simplicity if needed. By annihilating all the features other than those selected in $\{I_k^{(j)}\}$, we can construct a histogram based on those in $\{I_k^{(j)}\}$ for $k \geq 0$ and then normalise it to the probability density function $p^T(x)$. Likewise, we can also construct the probability density function $p^B(x)$ for the local background based on $\{I_k^{(j)}\}$ for $k < 0$ via (6).

5. Shape Consolidation and Extraction

When an object is extracted from a new frame, it may be in the form of a haze of pixels, particularly when they are estimated probabilistically. Even though it may not matter that much if one is to estimate its rough area with a method like the ellipse fitting algorithm we would use mostly for our experiments later on, it helps if one can enhance the object image or probability maps further on. Given any pixel position $s \in \mathbb{Z}^2$, the total weight of its neighbourhood $\Omega \subset \mathbb{Z}^2$ centred or pivoted at s,

$$w(s) = \frac{(\sum_{s' \in \Omega} x(s+s'))}{|\Omega|}, \tag{16}$$

where $|\Omega|$ denotes the cardinality of set Ω, can be used to determine how the pixel value $x(s)$ at position s should be enhanced to $x^*(s)$ according to

$$x^*(s) = \frac{(\sum_{s' \in \Omega^*} \alpha(s')x(s+s'))}{|\Omega^*|}, \quad \text{if } w(s) \geq \tau, \tag{17}$$

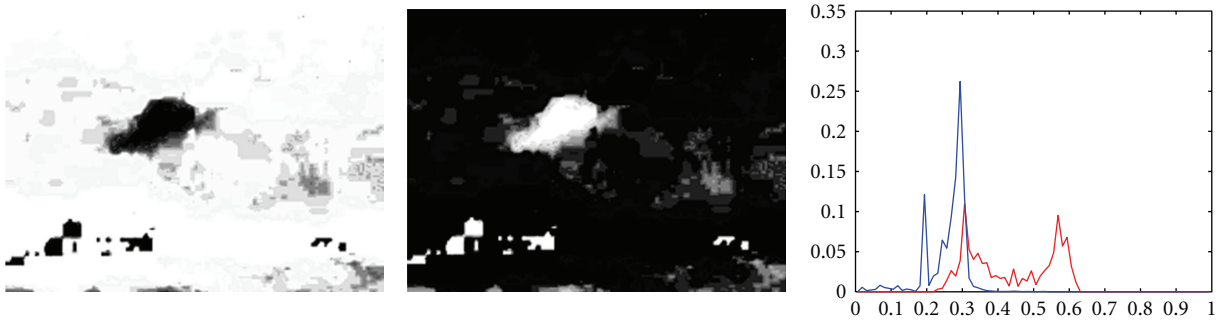
and $x^*(s) = 0$ if otherwise, where $\tau > 0$ is a controlling threshold, the α's are weight constants, and $\Omega^* \subset \mathbb{Z}^2$ is another neighbourhood of s which may or may not be the same as Ω. A simple such Ω can be taken as $\Omega = \{(0,0), (\pm 1, 0), (0, \pm 1), (\pm 1, \pm 1)\}$ which can be represented by the 3×3 grid of 1's, and a simple choice of the weight constants $\alpha(s)$ can be made as $\alpha(s) = \beta/(\|s\| + 1)$ for Euclidean distance $\|\cdot\|$ and a given constant β. This is essentially an iterative process we first considered in [20]. In general, we choose $\beta = \gamma/[\sum_{s \in \Omega^*} 1/(\|s\|+1)]$ with $\gamma \approx 1$, and slightly larger, so that the enhanced pixel value x^* is at about the same scale as the original pixel values, and slightly better. As the iteration proceeds, the total number of nonzero pixels will initially decrease due to the annihilation of isolated small patches of pixels and may, however, eventually turn around and increase instead because certain choices of β may make the enhanced solid body slighter expansive.

The choice of τ controls how far or fast the enhancement shrinks or expands the image. If the average intensity μ of the object is estimated by the average of the nonzero pixels $w(s)$, then τ should be of the scale $\mu/2$. In particular, if Ω is a $(2k+1) \times (2k+1)$ grid of pixels as the dotted box in Figure 4, then the intensity $w(s)$ at the centre point $\approx \mu k/(2k+1) \approx \mu/2$ will leave the pixel at 0 intensity if $\tau = \mu/2$. For the extraction of the object borderline, one can use essentially the known erosion technique [26]. One can first convert all nonzero pixels $x(s)$ of the image into the binary 1, then calculate the averaged image $w(s)$ via (16) for the Ω of 3×3 grid and again convert the nonzero $w(s)$ into 1, and finally derive the borderline as the nonzero pixels of the subtraction of these two images.

When noises are somewhat extensive in an image or map to be enhanced, a more effective way of removing the noises is to set the pixel threshold according to the percentage of the "target" pixels in its neighbourhood. For a given image to be enhanced, assumed to be gray scaled for simplicity without loss of generality, the intensity of the target object is expected to fall within a certain domain Ψ. For instance, if the difference image of two consecutive frames is to be enhanced, the brighter the pixel intensity, the more likely it is a pixel of a moving object. The domain Ψ in such a case could be chosen as for instance the top 1% of the nonzero pixels. Let such a domain $\Psi(\xi)$ be controlled by a parameter ξ. Then, for any pixel x at location s, if less than ζ percent

(a) (b)

Figure 17: Object tracked in 1st and last frame.



(a) (b) (c)

Figure 18: Posterior probability map via (6).

of pixels in its neighbourhood Ω' belong to Ψ, we increase the threshold τ in (17) by a factor τ'' to increase significantly the threshold because that pixel position under consideration is deemed far away from the target object. In other words, the τ in (17) is to be dynamically determined by the $\tau(s)$ below:

$$\tau(s) = \begin{cases} \tau', & \text{if } \dfrac{|\omega|}{|\Omega'|} \geq \zeta, \\ \tau'\tau'', & \text{if } \dfrac{|\omega|}{|\Omega'|} < \zeta, \end{cases} \tag{18}$$

where $\omega = \{s' : x(s') \in \Psi,\ s' \in \Omega'_s\}$ represents those relatively sure pixels on the target in the neighbourhood Ω' of s. Figure 5 shows the effectiveness of this method of discriminative threshold for a feature frame from the meerkat video sequence in Figure 24. The original (half frame) is depicted in subimage 1, the two consecutive enhancements via (17) and (18) are shown as subimages 2 and 5, the two consecutive enhancements via (17) without using (18) are displayed as subimages 3 and 6, subimage 4 shows an single step of enhancement without (18), and finally subimage 7 illustrates the outcome of applying four consecutive steps of enhancements without (18). We note that subimages 2–7 contain only cropped middle area of the enhanced subimage 1. We, thus, observe that subimage 5 of discriminative threshold retains better the object body than subimages 4, 6, and 7.

6. Experiments

We first illustrate for a video sequence that many a single feature may be utilised on its own to track certain type of objects if the feature is distinctive enough, and different features may be able to trace or highlight different aspects of the tracked objects. We then show how we can improve the tracking accuracy by combining several features together at the same time, that is, by utilising a multidimensional feature vector.

6.1. Single Feature in Full Range. First, we observe the 1st frame in Figure 6 has the following images in HSV colours respectively in Figure 7. If we pick hue as the base feature, then the posterior probability of the object for HSV and RGB are sequentially in the 6 image plots in Figure 8. If we use hue colour as the feature for tracking, then a threshold will easily extract the object from the probabilistic map in the above figure. When the probability map is somewhat "foggy", then certain enhancement techniques may need to be utilised. This could be in the form of incremental threshold followed by incremental region growth, or in the form of (16) and (17). The increment property is to ensure a good trade off between removing noises and keeping the dominant part of interest. Such regional shrinkage and growth are typically controlled by the intensity at a given pixel and the average in the local neighbourhood. Another technique is to use region masking, and this is based on the idea that if one knows the rough whereabout of the object, then one

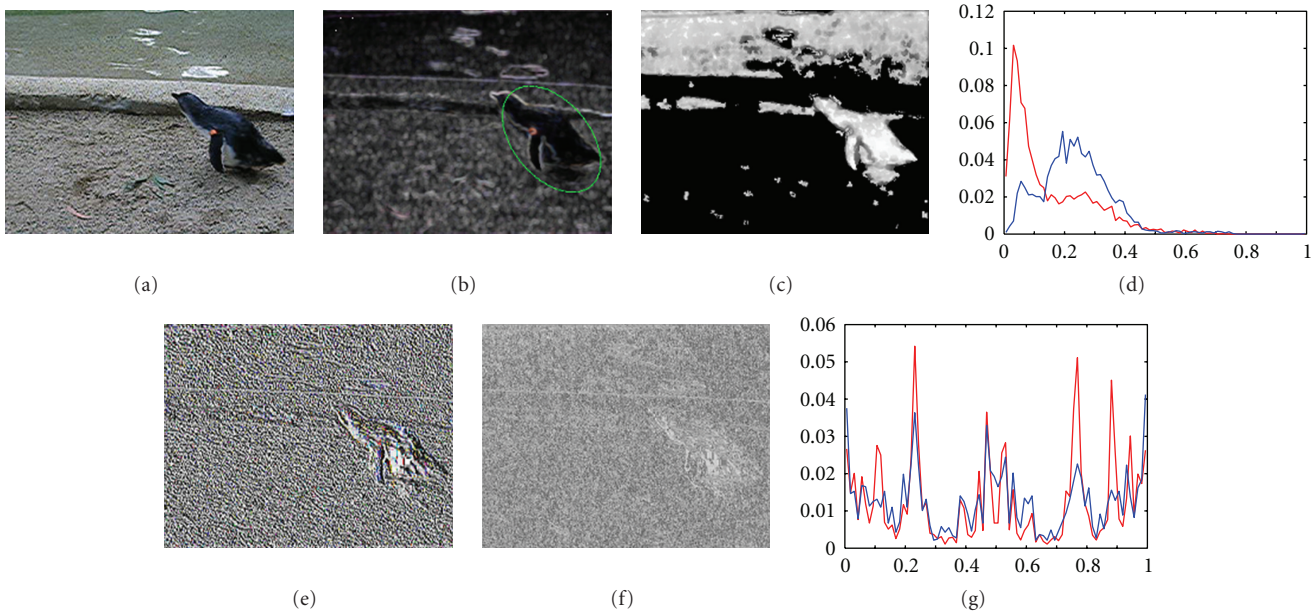


Figure 19: Top (a–d): for standard deviation. Bottom (e–g): for local binary patterns.

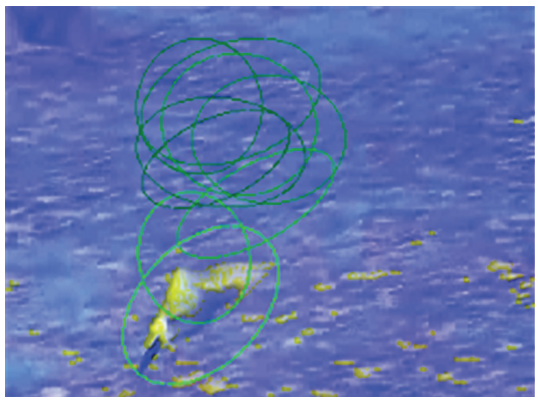


Figure 20: Tracking via LBP texture.

simply nullify directly the probability of those "far away" pixels. The procedure is as follows: (i) select a large radius for the object; (ii) choose the centre of the object in the previous frame as the approximate centre of the object in the current frame; (iii) nullify directly the probabilities of those exceeding the mask radius from the approximate centre; (iv) update the centre and repeat the above steps for several times if necessary. The probability map then indicates where the object is, as in Figure 9. We note that one can also make use of the method of dynamical threshold, as in (18), to determine if a pixel is far away from the object or not.

For an object of free form like a bird, tracking is largely about locating, where the object is rather than where precisely the object border locates. In this regard, one can use a preselected base shape to fit the object. Such a base shape could be a rectangle or an ellipse for instance. For this purpose, we first locate the centre of the object and draw a straight line through the centre. Then, locate the centres of two separate halves and draw a straight line linking them; see Figure 10. The fitting is done when these two straight lines become perpendicular, which resembles the diagram in Figure 11, where the two heavy dots denote the centres of the corresponding halves. Since the area of the right half of the ellipse in Figure 11 is $\pi ab/2$ and the moment of that area with respect to the y axis is $(2/3)a^2b$, we have $R \times \pi ab/2 = 2a^2b/2$ and thus $a = 3\pi R/4$. By evening out the potential imbalance between the left and right halves through an average, the radius on the axis connecting the two centres of the halves can be calculated based on the ellipse shape and thus reads

$$a = \left(\frac{3\pi}{8}\right)\left(\sqrt{(r_1 - r)^2 + (c_1 - c)^2} + \sqrt{(r_2 - r)^2 + (c_2 - c)^2}\right), \tag{19}$$

where (r_1, c_1) and (r_2, c_2) are the row and column of the two centres, respectively. Swapping the roles of the centre lines one can then directly calculate the radius b on the other axis. For clarity, we may also enlarge such a and b uniformly by a few pixels, resulting in the outer ellipse in Figure 9 for instance.

As a result of all these employed methodology and techniques, the typical tracked objects are exhibited in Figure 12 in which the tracked parts are circled by the red ellipses. If one wishes to follow the steps in Section 5, one can further improve the object shapes from the ellipses in Figure 12 to their closer form; see the first few frames in Figure 13.

We note that if one wishes to select the motion feature for this tracking, synchronisation of the frame background is needed because the background view shifts in 10 s of pixels across each frame. We applied the background synchronisation described late in Section 3 to the 20 consecutive frames, on mainly the red colour space. The synchronisation all succeeded apart from frames 11, 14, and 17, which in turn succeeded in hue for frame 11 and in green for frames 14

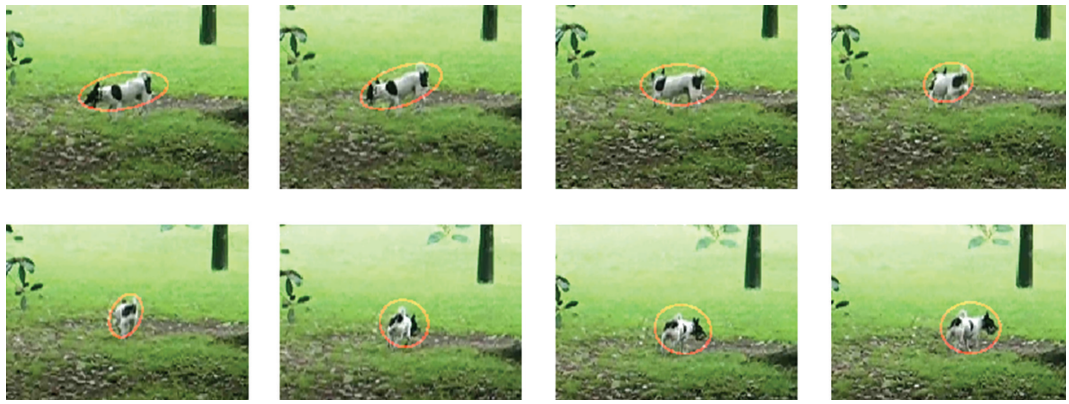


Figure 21: Dog tracked with hue and saturation.

Figure 22: Meerkat scene.

and 17. This shows the motion component can also serve well to track the bird in this sequence. In Figure 14, we depict a typical improvement on the right from the direct frame difference on the left by the background synchronisation.

6.2. Single Feature of Selected Values. If we plot the histograms for both the object in red and the background in blue, as for HSV and RGB in the following Figure 15, we observe that some colours differentiate better the object and the background.

For the hue histograms, we observe that the object histogram (in red) is most discriminative at 0.57 which corresponds to the 46th bin for the 80 bin histogram, and the background peaks at 0.29 corresponding to the 24th bin. In this experiment, we show that by selecting just *several* discriminative feature values and nullifying the histogram for the other values, the tracking is often still possible. This shows that one may merely opt for several feature values to simplify the histograms and consequently the whole calculation. In this example, we push this idea to the extreme and nullify the whole histogram apart from the 46th bin for the hue colour. The histogram becomes trivial as in Figure 16, but the tracking has not been much affected; see Figure 17.

We can also select the dominant values for the background to serve the purpose, and we just need to make use of (6) instead of (5). Figure 18(a) displays the posterior probability map for the background, with its reverse displayed in (b) for better visibility, while (c) shows the corresponding histograms. We note that we enlarged a little the size of the local window to make it more representative of the background.

6.3. Feature Varieties. Since a distinctive feature could be a particular colour or texture, or any contrived mathematical quantity, we illustrate below a few different type of features for our tracking purpose. First, we examine the use of the local standard deviation. For each pixel point of a chosen image space, calculate the standard deviation of the neighbouring, say 9 or 21, pixel values. For the video sequence in Figure 19(a) below, for instance, its red colour image in standard deviation of 21 neighbours is shown in (b) while its posterior probability image is depicted in (c). The corresponding histograms are plotted in (d), while the tracked object base shape is outlined in green in (b).

We also applied the LBP as the feature value to this video. The LBP image is shown in Figure 19(e), its posterior probability map is depicted in (f), and the corresponding histograms are plotted in (g). The LBP seems to somewhat randomise the original image, as shown in the histograms, and fails to serve as an acceptable feature for the object tracking. For the texture of standard deviation, it served fine for a few frames until similar textures become abundant in the neighbourhood of the object. It is overall less robust than the colour features.

However, if we utilise the correlated texture as the feature for tracking, then the outcome is much improved. In this connection, we experimented on the video sequence in Figure 6 with the textures defined by LBP, while the correlated blue colour ranges are chosen as 0.55 to 1. The tracking is fine, and the result is shown in Figure 20 in which brighter ellipses indicate later object traces and the slight overall colour change is due to the neglect of those unselected blue colours as well as the LBP application to those colours.

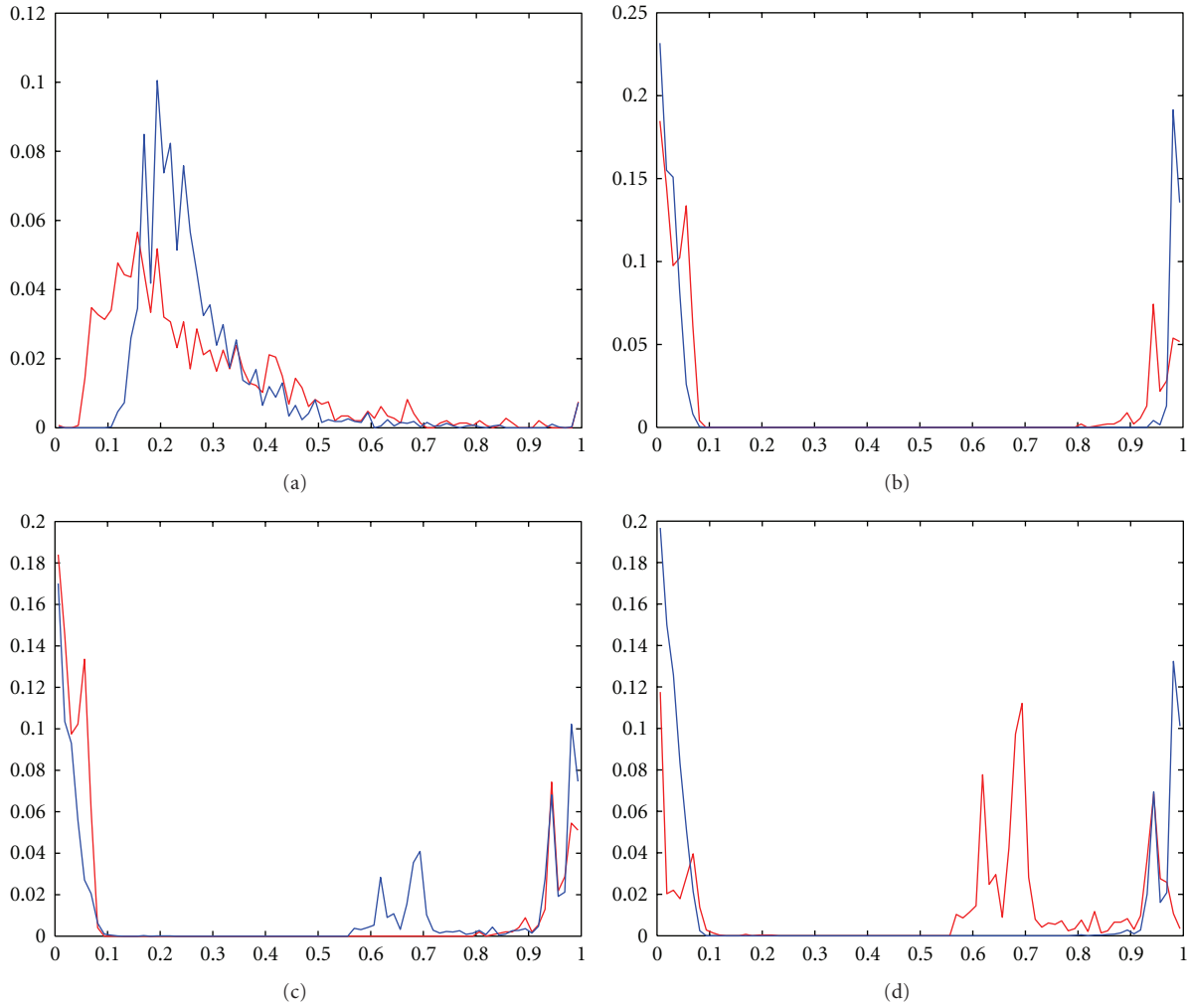
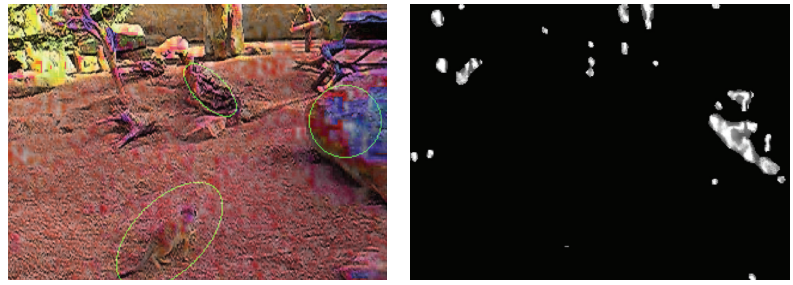


Figure 23: Top (a, b): histograms for saturation and hue. Bottom (c, d): histograms for hue.

Figure 24: Traces of object in 12 frames.

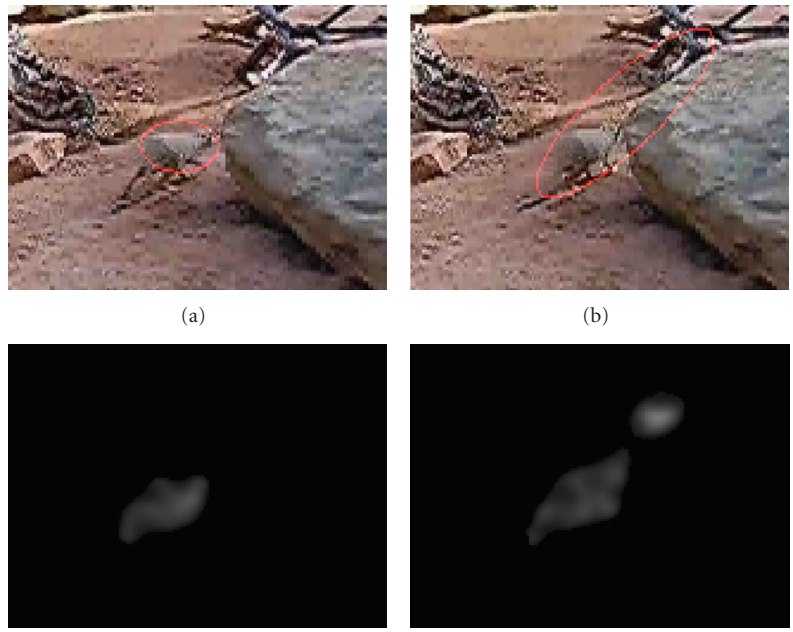
6.4. Combination of Several Features. There are two different ways to utilise multiple features for the tracking purpose. They differ according to at which stages these different features are combined. One method is to track simultaneously and separately with each individual feature and then unify their separate results together using some kind of confidence voting system, thus leading to an improved tracking performance. The walking dog in the following sequence in Figure 21, displayed every 4 frames, is tracked this way in terms of the use of hue and saturation colours.

The other is to track the combined features. In the next example of camouflaged meerkat, if we use the red-circled parts in Figure 22 as object and background, the tracking via the saturation colour is possible during the initial steps of "clean" environment, as is indicated in the histogram Figure 23(a). However, it is less stable when the meerkat is close to other camouflaging objects like the tree stumps and the rock. If we add these (within green circles) to the background samples, then they represent the blue spikes in the middle of Figure 23(c) for the hue colour in comparison with the corresponding flat part in (b) when these are not added. This indicates that the use of an additional feature, the hue colour, will differentiate the background better and make the tracking more robust. In fact, the hue colour of the rock is quite dominant as is shown also in Figure 23(d) which plots the case of the rock (within green circle) against



(a) (b)

Figure 25: Track with texture on saturation.



(a) (b)

(c) (d)

Figure 26: Combine colour and motion features.

the background of tree stumps (within green ellipse) and the meerkat (within the larger red ellipse). To improve the robustness, we adopt saturation colour as the affirming feature with range selection 0.05 to 0.15 and adopt hue colour as the negating feature with ranges .55 to .75 and 0 to .1 The ω_i in (2) are set to equal to each other and to 1/160, and a combination form similar to (7) is also utilised. The tracking results are shown in Figure 24.

Alternatively, we can select the rock's feature to consolidate its background status via (6) or simply treat it as a background object. Figure 25(a) shows the rock is sampled in a circle while nonrock environment is sampled in two ellipses. The rock feature here is exemplified with the use of texture LBP on the saturation and then mean of the 21 neighbourhood cells. Figure 25(b) shows the rock probabilities which can be incorporated with the normal tracking in Figure 22.

Since the meerkat is moving in this video sequence, the motion feature can stand out the object from the cluttered background. If we use the frame differences to extract the motion feature according to Sections 2.2 and 3, choosing in this case simply $\alpha = \beta = \eta = 0$ in (9) and with no extra frame synchronisation as the frame background is already static enough, then we can combine just the features for the saturation colour and for the motion. Figure 26 illustrates a clear separation of the object meerkat from the camouflaging rocks. Figures 26(c) and 26(d) display the enhanced motion feature for frames 11 and 12, while Figures 26(a) and 26(b) show the consequent tracking effect. We note that the motion feature also detects the tiny head of a second meerkat coming into the scene which was not detected in the previous schemes.

Further experiments can be done with other type of features such as other type of textures. It is also possible to

apply to several objects at the same time, with all but one such object treated as essentially the background. However, these are beyond our current scope.

7. Conclusion

We proposed to represent the object in terms of dominant feature vectors of colours and textures, and possibly motion, in the local environment and use them to track the object in the video frames. Such effective feature elements can be extracted dynamically for the object modelling. The feature elements are determined by their collective power to distinguish the object from its background. It is also noted that the impact of multidimensionality can be significantly reduced if insignificant feature cubes are directly nullified.

Acknowledgments

The author thanks Zhuan Qing Huang for making available some useful Matlab-coded functions and video clips of her own and Xiling Guo for some programming assistance.

References

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processsing*, Prentice-Hall, New York, NY, USA, 2nd edition, 2002.

[2] D. S. Lee, "Effective Gaussian mixture learning for video background subtraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 827–832, 2005.

[3] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1151–1162, 2002.

[4] M. Miyoshi, J. K. Tan, and S. Ishikawa, "Extracting moving objects from a video by sequential background detection employing a local correlation map," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '08)*, pp. 3365–3369, October 2008.

[5] A. K. Jain, Y. Zhong, and S. Lakshmanan, "Object matching using deformable templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 3, pp. 267–278, 1996.

[6] F. Tombari, S. Mattoccia, L. Di Stefano, F. Regoli, and R. Viti, "A template analysis methodology to improve the efficiency of fast matching algorithms," in *Proceedings of the 11th International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS '09)*, vol. 5807 of *Lecture Notes in Computer Science*, pp. 100–108, Bordeaux, France, September-October 2009.

[7] T. Chateau and J.T. Lapreste, "Realtime kernel based machine learning template matching (KMLT)," *Electronic Letters on Computer Vision and Image Analysis*, vol. 8, no. 1, pp. 27–43, 2009.

[8] H. T. Nguyen and A. W. M. Smeulders, "Fast occluded object tracking by a robust appearance filter," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 1099–1104, 2004.

[9] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1987.

[10] N. Paragios and R. Deriche, "Geodesic active contours and level sets for the detection and tracking of moving objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 3, pp. 266–280, 2000.

[11] M. Isard and A. Blake, "CONDENSATION—conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[12] C. Zimmer and J. C. Olivo-Marin, "Coupled parametric active contours," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1838–1842, 2005.

[13] L. Pi, J. Fan, and C. Shen, "Color image segmentation for objects of interest with modified geodesic active contour method," *Journal of Mathematical Imaging and Vision*, vol. 27, no. 1, pp. 51–57, 2007.

[14] Z. Ying, L. Guangyao, S. Xiehua, and Z. Xinmin, "Geometric active contours without re-initialization for image segmentation," *Pattern Recognition*, vol. 42, no. 9, pp. 1970–1976, 2009.

[15] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[16] A. Elgammal, R. Duraiswami, and L. S. Davis, "Efficient kernel density estimation using the fast Gauss transform with applications to color modeling and tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 11, pp. 1499–1504, 2003.

[17] Y. Zha, D. Bi, and Y. Yang, "Learning complex background by multi-scale discriminative model," *Pattern Recognition Letters*, vol. 30, no. 11, pp. 1003–1014, 2009.

[18] T. J. Cham and J. M. Rehg, "Multiple hypothesis approach to figure tracking," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99)*, pp. 239–245, Ft. Collins, Colo, USA, June 1999.

[19] Y. Sheikh and M. Shah, "Bayesian modeling of dynamic scenes for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1778–1792, 2005.

[20] Z. Q. Huang and Z. Jiang, "Tracking camouflaged objects with weighted region consolidation," in *Proceedings of the Digital Imaging Computing: Techniques and Applications (DICTA '05)*, B. C. Lovell et al., Ed., pp. 161–168, IEEE CS, Cairns, Queensland, December 2005.

[21] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–1643, 2005.

[22] X. P. Zhang and M. D. Desai, "Segmentation of bright targets using wavelets and adaptive thresholding," *IEEE Transactions on Image Processing*, vol. 10, no. 7, pp. 1020–1030, 2001.

[23] J. Bilmes, "A gentle tutorial of the em algorithm and its application to parameter estimation for Gaussian mixture and hidden markov models," Tech. Rep. TR-97-021, ICSI, 1997, http://citeseerx.ist.psu.edu/~viewdoc/summary?doi=10.1.1.28.613.

[24] A. Mason and Z. Duric, "Using histograms to detect and track objects in color video," in *Proceedings of Applied Imagery Pattern Recognition Workshop*, pp. 154–159, 2001.

[25] M. Heikkilä and M. Pietikäinen, "A texture-based method for modeling the background and detecting moving objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 657–662, 2006.

[26] A. Amer, "New binary morphological operations for effective low-cost boundary detection," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 17, no. 2, pp. 201–213, 2003.