Adaptive Texture and Color Segmentation for Tracking Moving Objects

Ercan Ozyildiz, Nils Krahnstöver, Rajeev Sharma [1]

Department of Computer Science and Engineering, Pennsylvania State University, 220 Pond Lab, University Park, PA 16802, Phone: (814) 865-9505, Fax:(814)865-3176, {*ozyildiz,krahnsto,rsharma*}*@cse.psu.edu*

Abstract

Color segmentation is a very popular technique for real-time object tracking. However, even with adaptive color segmentation schemes, under varying environmental conditions in video sequences, the tracking tends to be unreliable. To overcome this problem, many multiple cue fusion techniques have been suggested. One of the cues that complements color nicely, is texture. However, texture segmentation has not been used for object tracking mainly because of the computational complexity of texture segmentation. This paper presents a formulation for fusing texture and color in a manner that makes the segmentation reliable while keeping the computational cost low, with the goal of real-time target tracking. An autobinomial Gibbs Markov Random Field (GMRF) is used for modeling the texture and a 2D Gaussian distribution is used for modeling the color. This allows a probabilistic fusion of the texture and color cues and for adapting both the texture and color over time for target tracking. Experiments with both static images and dynamic image sequences establish the feasibility of the proposed approach.

Key words: Visual Tracking, Color Segmentation, Texture Segmentation, Cue Fusion

[1] Corresponding author.

1 Introduction

Some of the most popular methods for real-time visual tracking of moving objects are based on color segmentation [1, 2, 3]. The main reason for choosing color-based segmentation is that the color cue is relatively invariant to scale, illumination and viewing direction while being computationally efficient.

Although the different color-based tracking approaches reported in the literature demonstrate a certain degree of adaptation to object color changes, for complex backgrounds and changing environments, the reliance on one cue can lead to poor tracking performance. This suggests the use of multiple cues for tracking.

Texture provides a way of characterizing the spatial structure of an object and can complement the use of color for reliable object tracking. In the presence of uncertainty, pixel neighborhood information can be exploited for more robust segmentation. Texture segmentation uses neighborhood statistical information since it is a non-local process unlike pixel-based color segmentation. However, texture segmentation is computationally demanding. Hence a combination of a color segmentation scheme with texture segmentation can be used advantageously to achieve real-time robust object tracking.

There is a large body of literature that addresses different aspects of the texture segmentation problem. However, texture segmentation has not been considered in the context of real-time object tracking mainly because of two reasons. First, texture parameter estimation and segmentation is computationally inefficient, making real-time implementation difficult. Second, unlike color, obtaining scale and rotation invariant texture information in a dynamic environment is very difficult.

In this work a new approach for tracking a moving object using a combination of adaptive texture and color segmentation is proposed. A Gibbs Markov Random Field (GMRF) is used for modeling the texture and a 2D Gaussian distribution is used for modeling the color. This allows a probabilistic framework for fusing the texture and color cues at region level and for adapting both the texture and color segmentation over time for target tracking. The fusion of texture and color makes the segmentation reliable while keeping the computation efficient with the goal of real-time target tracking. A Kalman filter based motion estimation and prediction further helps in improving performance. Experiments with both static images and dynamic image sequences are used for establishing the feasibility of the proposed approach. The experiments were conducted for both indoor and outdoor scenes. Scenes with complex backgrounds and mixtures of similar object

colors and textures were particularly chosen to test the algorithm. The experiments show that the probabilistic fusion of texture and color information at the region level improves the robustness of the tracking system under difficult visual conditions.

The rest of paper is organized as follows. Section 2 discusses the background and related work on color and texture segmentation in the context of target tracking. Section 3 explains characterization of textured images by using Gibbs Markov Random Fields(GMRF), the autobinomial model and the parameter estimation method. Section 4 gives a formulation of color segmentation using a Gaussian distribution that is appropriate for adaptive segmentation. Section 5 presents a formulation for fusing color and texture. Section 6 proposes an approach for adapting texture and color segmentation during the tracking process. Section 7 presents the experimental results and performance analysis of color and texture segmentation on both static images and dynamic image sequences. Section 8 concludes with a discussion of the issues involved in improving color and texture segmentation for more efficient and reliable target tracking.

2 Background and Related Work

2.1 Color Segmentation

In recent years, the analysis of color images has been playing an important role in computer vision because color can provide an efficient cue for focus of attention, object tracking and recognition allowing real time performance to be obtained using only modest hardware. Color segmentation is also computationally efficient and relatively robust to changes in illumination, in viewing direction, and in scale. Robustness is achieved if the color components are efficiently separated from luminance in the original image and color distribution is represented by a suitable mathematical model for thresholding [4].

Sharbek et al. [5] reviewed color segmentation algorithms and categorized them as pixel, area, edge and physics based segmentation according to their attributes. There is no winner among color segmentation algorithms. The effectiveness of an algorithm changes with application. Several researchers have compared different color spaces for the application of skin detection [4][6]. For example, [7] and [8] present comparisons of unsupervised and supervised color segmentation algorithms respectively.

For online applications, two color segmentation methods are mostly used. The first method uses Gaussian mixture models to characterize color distribution of an object [1] [3] [4]. The second method employs a histogram model [9][10]. The histogram-based methods are basically non-parametric forms of density estimation in color space.

Although color is an efficient cue for computer vision applications, a number of viewing factors, such as light sources, background colors and luminance levels, have a great impact on the change in color appearance. Most color-based systems are sensitive to these changes. There are two major approaches for handling environmental changes. The first approach finds the effects that change the color and use them as inverse filter to obtain the real color [11][12][13]. The second approach, adaptive color segmentation, adapts the previously developed color model to the changing environment [1][3]. This approach is more suitable to compensate for changes in the natural environment. However, just adapting color may not achieve the robustness desired, motivating multiple cue approaches.

2.2 Texture Segmentation

Texture has been widely accepted as a very important feature in image processing and computer vision since it provides unique information about the physical characteristics of surfaces, objects, and scenes [14]. Numerous texture segmentation methods have been proposed in the past. Reed et al.[15] categorized texture feature extraction methods as feature-based, model-based and structural. Especially stochastic image models based on the Gibbs distribution have received a lot of attention and have been applied in ecology, sociology, statistical mechanics and statistical image modeling and analysis [16, 17].

Gibbs Markov Random Field (GMRF) theory provides a foundation for the characterization of contextual constraints and the derivation of the probability distribution of interacting features [18]. A comprehensive discussion about the use of MRF in computer vision and the statistical aspects of images is given in [18] [19]. GRFs were applied to textured images for modeling and segmentation by Derin et al. [20]. Cross et al. [21] explored the use of GMRFs as texture models. They used the binomial model where each point in the texture has a binomial distribution with a parameter controlled by its neighbors and ”number of trials” equal to the number of gray levels. Schroder et al. [17] recently used the autobinomial GMRF model as a powerful, robust descriptor of spatial information in typical remote-sensing image data.

Panjwani et al. presented a model that extracted texture information from the interaction within and between color bands [22].The disadvantage of this method is its computational complexity. Dubuisson et al. combined texture and color separately by using maximum likelihood estimation [23]. Because of this separation they decreased the interaction of cues.

2.3 Tracking with Multiple Cue Fusion

Recent developments in computing devices, video processing hardware, and sensors enable the construction of more reliable and faster visual tracking systems. Despite these advances, most visual tracking systems are brittle. In particular, the systems which rely on a single cue or methodology for locating their targets are easily confused in commonly occurring visual situations. Some color based tracking techniques try to increase the reliability by updating the color cues over time [1, 2, 3]. Other approaches similarly employ multiple cues to get more reliable information [14, 24, 25, 26].

Yang et al. introduced a tracking algorithm which constructs a self-adapting model from the detected moving object, using features such as shape, texture, color, and edgedness[25]. In a work by Murrieta et al. color and texture information was used to characterize and track specific landmarks [24]. Paschos et al. described a visual monitoring system that performs scene segmentation based on color and texture information [14]. Color information was combined with texture to detect and measure changes in a given space or environment over a period of time. Deng et al. used a combination of color, texture and motion to analyze and retrieve video objects [26]. However, none of these reported works utilize a combination of texture and color for tracking moving objects.

3 Formulation of Texture Segmentation

In this work, Gibbs Random Fields are used to model the texture information. This section first briefly reviewes GRF theory. Furthermore, the auto-binomial GRF, is discussed, followed by a description of the linear parameter estimation process.

The image is assumed to be the realization of a random field x_s of pixel sites. This random field is called Markovian, if the probability density function of the pixel x_s is completely determined by the values of the pixels in the local neighborhood $\mathrm{N}(x_s)$. Figure 1 shows a particular neighborhood

$\mathrm{N}(x_s) = \{x_i, \hat{x}_i\}$. The index labels the two pixel cliques consisting of two facing neighbors x_i and $\hat{x}_i$.

Such a Markov field can be described as a Gibbs field with a local energy function $H(x_s, \mathrm{N}(x_s); \theta)$ with θ being a vector of scalar parameters reflecting the influence of the different cliques $\{x_i, \hat{x}_i\}$. For a single pixel site x_s, this approach results in the conditioned probability distribution

$$p(x_s|\mathrm{N}(x_s)) = \frac{1}{Z} e^{-H(x_s, \mathrm{N}(x_s); \theta)} \tag{1}$$

The variable Z, the partition sum, normalizes the distribution. The parameter vector θ weights the contributions of the different neighborhood pixels and is a parameterization of the image content.

The form of the energy function should be chosen depending on the image model. In this work, the autobinomial model, which was recently used by Schroder et al. [17] with a different scaling of parameters, is used for modeling the image. Using this model the parameter estimation can be done efficiently while being able to characterize real images adequately. In the following, the notation and formalism is adapted from the work presented in [17].

In the autobinomial model, the energy function is defined as

$$H(x_s, \mathrm{N}(x_s); \theta) = -\ln \begin{pmatrix} G \\ x_s \end{pmatrix} - x_s \eta(\theta, x_s) \tag{2}$$

where the G denotes the maximum gray value and $\begin{pmatrix} n \\ m \end{pmatrix}$ denotes the binomial coefficients. The influence of the neighboring pixels is represented by the linear equation

$$\eta(\theta, x_s) = \theta_0 + \sum_i \theta_i \frac{x_{si} + \hat{x}_{si}}{G}. \tag{3}$$

The coefficients

$$\theta = [\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6 ...]^T \tag{4}$$

determine interaction between the pixels and hence the statistical properties of the random field x_s. A typical realization of the random field corresponds to a typical appearance of the image or in our context to the spatial properties of the texture.

For the autobinomial model, the partition sum Z is calculated as

$$Z = \sum_{x=0}^{G} e^{-H(x,\eta)} = (1 + e^{\eta})^{G} \tag{5}$$

Therefore, the PDF of the autobinomial can be written as

$$p(x_s|\mathrm{N}(x_s), \theta)(x_s)) = \begin{pmatrix} G \\ \\ x_s \end{pmatrix} \varrho^{x_s}(1 - \varrho)^{G - x_s} \tag{6}$$

with $\varrho = 1/(1 + e^{-\eta})$.

The mean and variances are found to be

$$\mathrm{E}[x] = \frac{G}{1 + e^{-\eta}} \tag{7}$$

and

$$\mathrm{Var}[x] = \mathrm{E}[x^2] - \mathrm{E}[x]^2 = G\frac{e^{-\eta}}{(1 + e^{-\eta})^2} \tag{8}$$

For $\eta = 0$, the mean and variance are obtained as $\mathrm{E}[x] = G/2$ and $\mathrm{Var}[x] = G/4$. Before the parameter estimation, data is normalized to these values. Therefore, the estimated parameters depend on spatial information more than intensity radiometric properties.

The parameter vector θ of the Model parameterizes the spatial information of the image. Reliable and robust MRF parameter estimation can in general be quite complex [18], but for the autobinomial model an efficient closed form conditional least square (CLS) solution exists and were recently applied to remote sensing data retrieval [17].

The CLS estimator is defined as

$$\hat{\theta} = \arg\min_{\theta} \sum_{s} (x_s - \mathrm{E}[x_s])^2 \tag{9}$$

By using the expectation (7), equation (9) can be written as

$$\hat{\theta} = \arg\min_{\theta} \sum_{s} (x_s - \frac{G}{1 + e^{-\eta}})^2. \tag{10}$$

In [17] it was shown how that, using a mean field approximation, η can be approximated as

$$\eta \approx -\log\left(\frac{G}{x_s} - 1\right) + N(0, \delta) \tag{11}$$

where $N(.,.)$ is a Gaussian noise term of zero mean and some small variance δ. With the definition of η, equation (3), the parameter estimation process is reduced to the overdefined set of linear equations

$$\mathbf{G}\theta = \mathbf{d} + \mathbf{n}, \tag{12}$$

where θ is the unknown parameter vector,

$$\mathbf{d}_s = -\log\left(\frac{G}{x_s} - 1\right), \tag{13}$$

and **n** a Gaussian noise vector. **G** is a matrix containing the neighboring pixel values

$$\mathbf{G}_{s,t} = \begin{cases} 1 & \text{if } t = 0 \\ \frac{x_{st} + \hat{x}_{st}}{G} & \text{otherwise.} \end{cases} \tag{14}$$

The values x_{st} and $\hat{x}_{st}$ denote the two t-th neighbors of the pixel x_s according to the definition in Figure 1. With the assumption that the noise term **n** is Gaussian, the maximum likelihood estimate of θ is given by

$$\hat{\theta} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{d} \tag{15}$$

Thus, this provides a closed form solution for the texture parameters allowing to efficiently map texture image information to a low dimensional parameter space, making it appropriate for target tracking.

4 Formulation of a color space and model

The selection of the color space is one of the key factors for efficient color information extraction. A number of color space comparisons are presented in the literature as stated in section 2.1. After experimentally observing the effect of different color spaces on the segmentation results, the YES space was selected as the most appropriate color space. For details, the readers can consult [27].

Luminance-chrominance is known as the YES space, where "Y" represents the luminance channel and "E" and "S" represent the chrominance components. The YES space is defined by a linear transformation of the SMPTE (Society of Motion Picture and Television Engineers) RGB coordinates, given by

$$\begin{bmatrix} Y \\ E \\ S \end{bmatrix} = \begin{bmatrix} 0.253 & 0.684 & 0.063 \\ 0.500 & -0.500 & 0.000 \\ 0.250 & 0.250 & -0.500 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{16}$$

The second important element of color segmentation is the choice of the model for the object color distribution. Color histograms and the Gaussian models have been successfully used for real-time color segmentation systems. Color histograms [28] are a simple and non-parametric method for modeling color. But they need sufficiently large datasets in order to work reliably. A second drawback is the difficulty of adapting the model over time. A more effective approach is to model the color distribution with a 2D Gaussian. The representation of color distribution is possible using relatively little data with the Gaussian model. It is also suitable for adaptive color segmentation.

Based on the chrominance components E and S, a bivariate(2D) Gaussian distribution $N(\mu_c, \Sigma_c^2)$ with mean μ_c and covariance Σ_c is used to represent the distribution of object color. The 2D Gaussian probability density function is

$$p(\mathbf{z}_c|\text{objectcolor}) = \frac{1}{2\pi|\Sigma_c|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}[\mathbf{z}_c - \mu_c]^T \Sigma_c^{-1} [\mathbf{z}_c - \mu_c]\right), \tag{17}$$

where

$$\mathbf{z}_c = \begin{bmatrix} E \\ S \end{bmatrix}, \mu_c = \begin{bmatrix} \mu_E \\ \mu_S \end{bmatrix}, \Sigma_c = \begin{bmatrix} \sigma_E^2 & \sigma_{ES} \\ \sigma_{SE} & \sigma_S^2 \end{bmatrix}.$$

The distribution $p(\mathbf{z}_c|\mathrm{objectcolor})$ is simplified to the Mahalanobis distance ($[\mathbf{z}_c - \mu_c]^T \Sigma_c^{-1} [\mathbf{z}_c - \mu_c]$) by taking the natural logarithm. Color segmentation is performed by calculating the Mahalanobis distance for the pixels in a small region and comparing its values to a threshold τ_c.

5 Fusion of Texture and Color

In this work, color and texture segmentation is integrated by estimating their joint probability distribution function (PDF). Using a joint probability function of cues enables one cue to support the other one if it becomes unreliable due to environmental changes. Texture and color PDFs are combined at the region level because reliable texture information extraction is only possible if performed with sufficient sample sizes. The region $\mathrm{R}(x_s)$ is defined as a 3×3 sub-window in the 5×5 texture region window centered around the pixel site x_s as shown in Figure 2. Each window element itself contains $M \times M$ pixels for typical values $M = 5 \ldots 15$ depending on the texture scale.

Because color and texture features are conditionally independent, the probability of the region belonging to class ω can be expressed as:

$$p(\mathbf{z}_{t,s}, \mathbf{z}_{c,s}|\omega) = p(\mathbf{z}_{c,s}|\omega) p(\mathbf{z}_{t,s}|\omega) \tag{18}$$

where $\mathbf{z}_{c,s} = [ES]^T$ denotes the average chrominance vector and $\mathbf{z}_{t,s}$ denotes the average texture feature vector in the neighborhood $\mathrm{R}(x_s)$ around the pixel site x_s. With the above definition of $\mathrm{R}(x_s)$, this means the average value of $9M^2$ pixel samples. To keep the notation in the remainder of this section simple, the pixel site index s is dropped.

As described in section 4, the color distribution is modeled by a Gaussian:

$$p(\mathbf{z}_c|\omega_i) = \frac{1}{(2\pi)|\Sigma_{c,i}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\lambda_{c,i}\right), \tag{19}$$

where

$$\lambda_{c,i} = [\mathbf{z}_c - \mu_{c,i}]^T (\Sigma_{c,i})^{-1} [\mathbf{z}_c - \mu_{c,i}]. \tag{20}$$

$\mu_{c,i}$ and $\Sigma_{c,i}$ denote the mean vector and covariance matrix of $\mathbf{z}_c$, respectively, for class ω_i.

Let θ_n denote the texture parameter vector estimated from the n^{th} neighbor window of the region centered around x_s. As for the color distribution, this estimate is based on $9M^2$ pixel samples. The texture feature distribution is modeled by a Gaussian probability distribution. The probability of the n^{th} neighbor window region belonging to class ω_i can be written as

$$P_{ni} = p(\theta_n|\omega_i) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_{t,i}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\lambda_{t,ni}\right), \tag{21}$$

where

$$\lambda_{t,ni} = [\theta_n - \mu_{t,i}]^T (\Sigma_{t,i})^{-1} [\theta_n - \mu_{t,i}] \tag{22}$$

and $\mu_{t,i}$ and $\Sigma_{t,i}$ denote the mean vector and covariance matrix of θ, respectively, for class ω_i.

The texture parameters are estimated for each of $N = 9$ neighbor windows as shown Figure 2. Therefore, $p(\mathbf{z}^t|\omega)$ can be expressed as a function of N neighbor probabilities:

$$p(\mathbf{z}_t|\omega_i) = \mathrm{F}(P_{1i}, P_{2i}, P_{3i}, ...P_{Ni}) \tag{23}$$

The experimental results show that the distance of the region texture vector $\mathbf{z}_t$ to the class ω_i, $\lambda_{t,i}$, can be expressed as the average of the N neighbor distances $\lambda_{t,ni}$

$$p(\mathbf{z}_t|\omega_i) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_{t,i}|^{\frac{1}{2}}} \exp\left(\frac{-\sum_{n=1}^{N} \lambda_{t,ni}}{2N}\right) \tag{24}$$

$$= \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_{t,i}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\lambda_{t,i}\right) \tag{25}$$

With assumption of conditional independence of the color and texture cues, the PDFs $p(\mathbf{z}_c, \mathbf{z}_t|\omega_i)$ can be approximated as

$$p(\mathbf{z}_c, \mathbf{z}_t|\omega_i) \propto \exp\left(-\frac{1}{2}(\lambda_{c,i} + \lambda_{t,i})\right) \tag{26}$$

Because of the assumption of dynamic and complex background, the representation of the background is very difficult. Therefore, the region classification proceeds with the following hypothesis test:

$$x \in \begin{cases} \omega_{\text{target}} & \text{if } \left(\frac{\lambda_{c,\text{target}}}{\tau_c} + \frac{\lambda_{t,\text{target}}}{\tau_t}\right) < \tau_{\text{fusion}} \\ \omega_{\text{background}} & \text{otherwise.} \end{cases} \tag{27}$$

where τ_c and τ_t are respectively color and texture segmentation thresholds which are specified by the user or computed by the system using the ROC curve analysis at the initialization level as described in [29][30]. When the desired segmentation is accomplished for both cues, the value of the combined distances becomes smaller than 2. The segmentation is obtained for τ_{fusion} values in the range $[1.5, 2.5]$.

6 Adaptation of Texture and Color for the Target Tracking

The appearance of the target object varies over time. If the tracking system does not compensate these changes, it may lose the object. Under the assumption that the appearance changes gradually, a statistical adaptation scheme is proposed. Basically, the adaptation scheme updates the mean and the covariance of the texture and color feature vectors over time. The process keeps L previous mean and covariance estimates and the number of sample points for each mean and covariance. The new estimates at time $(l+1)$ are calculated as follows:

$$\hat{\mu}_c^{l+1} = \frac{\mu_c^0 N_c^0 \sum_{s=l-L+1}^{l-1} \hat{\mu}_c^s N_c^s + \mu_c^l N_c^l}{N_c^0 + \sum_{s=l-L+1}^{l} N_c^s} \tag{28}$$

$$\hat{\Sigma}_c^{l+1} = \frac{\Sigma_c^0 M_c^0 + \sum_{s=l-L+1}^{l-1} \hat{\Sigma}_c^s M_c^s + \Sigma_c^l M_c^l}{M_c^0 + \sum_{s=l-L+1}^{l-1} M_c^s} \tag{29}$$

where N_c^l and M_c^l denote the number of sample points for calculating the color feature vector mean and covariance at time l, respectively. The texture variance and mean are calculated in the same manner

$$\hat{\mu}_t^{l+1} = \frac{\mu_t^0 + \sum_{s=l-L+1}^{l-1} \hat{\mu}_t^s N_t^s + \mu_t^l N_t^l}{N_t^0 + \sum_{s=l-L+1}^{l-1} N_t^s} \tag{30}$$

$$\hat{\Sigma}_t^{l+1} = \frac{\Sigma_t^0 M_t^0 + \sum_{s=l-L+1}^{l-1} \hat{\Sigma}_t^s M_t^s + \Sigma_t^l M_t^l}{M_t^0 + \sum_{s=l-L+1}^{l-1} M_t^s} \tag{31}$$

with N_t^l and M_t^l denoting the number of sample points for calculating the texture parameter vector mean and covariance at time l, respectively. Protecting the system from adapting to a wrong target is the most important problem for a self-adapting system. In the tracking system presented here are several levels of protection. After passing another test with threshold $\tau_{c/t}/2$, color mean samples are collected at the pixel level and the texture mean samples are collected at the region level. Samples of the covariances are collected after passing the fusion threshold. The sum of means at time $l = 0$ are added to the above estimates to make the system memorize the initial values. This way, the resulting estimates are always in close proximity to the initial values.

6.1 Algorithm

This section describes the overall system (Fig. 3). First a region of the color image, which is bounded by a tracking window, is sub-sampled and color segmented with the method described in section 4. The biggest connected component is selected as the region of interest. Within this region, the color image is segmented by fusing texture and color as described in section 5 and the biggest connected component again taken as the final target ROI. This two-step process effectively reduces the size of the region at which a texture analysis has to be performed. Suitable samples for the color and texture adaptation process are obtained from within the target ROI. The adaptation process updates the color and the texture model before the next frame. Using predicted and measured locations, a Kalman filter is used to estimate the next location of the target. A window centered at the estimated location is used as the initial region for the next frame. The window size is allowed to changed with the size of the target.

7 Experimental Evaluation

This section describes the experiments for evaluating the performance of the proposed adaptive texture and color segmentation method. Two test databases where created for this purpose. The first database contains static images with various objects each having different textures and colors, and was used to fine tune and develop the basic texture and color segmentation algorithm. The experimental results and observations are described in section 7.1. The second database consists of a set of dynamic image sequences and was used for testing the adaptation of color textures for moving target tracking. The results of the experiments are shown in section 7.2.

7.1 Experiments with Static Images

This section presents experimental results and observations for the segmentation of static images using color and texture fusion. The main purpose of these experiments is to show how the fusion of cues enhances the reliability of the segmentation. The proposed algorithm is tested on real color texture images. The images where chosen to have objects with colors and textures similar to the background.

Three real image examples were selected to show the importance and the effectiveness of color and

texture fusion. The segmentation result for an image containing three different wooden textures is shown in Figure 4. The system was initialized by taking a sample from the upper left texture in Figure 4(a). Figure 4(b) shows that the color segmentation itself cannot separate the left textured wooden surface from the right one. The segmentation that is achieved by combining texture and color is shown in Figure 4(c). The result shows that the texture cue clearly helps to discriminate the object from the background.

Figure 5 shows a scene containing a tree and a patch of lawn. The goal of this experiment is to separate the tree from the background of the image. An appropriate sample from the tree is taken to initialize the procedure. Figure 5(b) shows that the color segmentation of the tree object is very noisy. No clear discrimination between the tree and the background can be seen because both have very similar color distributions. In Figure 5(c) the image is thresholded using texture segmentation alone. This gives a better result compared to the color segmentation because the tree and the lawn differ more in texture than in color. However, small false positive components can still be seen. The image in Figure 5(d) shows the result of fusing texture and color cues. The tree is now clearly separated from the background. The frame was taken from a compressed movie file therefore the compression algorithm produces a noise pattern on the image which can be seen in Figure 8 (b).

The image in Figure 6 serves to demonstrate how the proposed algorithm increases the separation of a target object from the background. An image sample was taken from the center of the ball. The goal of this experiment is to investigate for each location (x, y) in the image, the similarity $z = 1/d(x, y)$ between that location and the ball (i.e., the image sample that was taken from the ball). In particular, this experiment shows the advantage of performing a texture parameter averaging over the nine neighbor windows for each image location. As the similarity measure, the inverse Mahalanobis distance $1/\lambda_{t,\text{ball}}$ according to equation (25) was used. Figure 7 shows the similarity measurement when we look along the y-axis. Figure 7(a) shows the result of similarity map obtained by using texture segmentation without performing parameter averaging (i.e., the texture parameter is calculated estimated only from the sub-centered region $N(x_s)$). Even though a separation between the objects can be seen in the graph, it is not enough to discriminate the ball from the background. Figure 7(b) shows the result for using the approach of averaging the nine texture distances obtained from neighboring region windows. It can be seen, that the separation is increased and the block variation is decreased. In Figure 7(c), the similarity map is obtained by using the proposed model for integration of color and texture. As seen in Figure 7(c), the color integration enhances the separation of the ball image quite well.

This section presents experimental results and observations for the proposed tracking system applied to dynamic image sequences. The main purpose of these experiments is to show how the tracking system performs in dynamic environments. The tracking system is tested on various synthetic and natural image sequences. Each experiment starts with an initialization process where a small area is selected for estimating the initial parameters of the color and texture models. After the parameter estimation, the tracking process begins with the segmentation step inside the selected area.

The first type of image sequences is captured from natural outdoor scenes. Several scenes that included trees and grass were selected for two reasons. Firstly, trees and grass have distinct textures while often being very similar in color. Secondly, due to the self-similarity at different scales, trees provide good textures at any viewing distance. Motion is introduced by changing the camera location in the scene. Variations in color and texture are obtained from the change in viewing direction and from the digitization process. The algorithm was operated on decompressed image data which produces a noise pattern, that changes with the motion of the camera. As an example, the difference between sample images taken from the same location of the tree object can be seen in Figure 8(a) for the stationary camera and in the Figure 8(b) for the moving camera.

Among the natural sequences, one image sequence is selected to demonstrate the result of the cue integration, the adaptation of model parameters and the tracking performance. Figure 8(c) shows the result of the tracking algorithm with the tree as the target. Figures 9(a) and (b) show the adaptation of the first component of the mean color vector and the first parameter of the texture model vector respectively. Figure 10 shows the result of tracking the tree with the tracking window (incorrectly) centered on the grass. Figures 11(a) and (b) again show the adaptation of the first components of the color and the texture vectors. It can be seen how the parameter vectors are adapted continuously over time during the sequence. In Figure 10 the tracker initially can't find the target object and does not perform any adaptation until frame number 75 when it finds the tree object and begins the adaptation. The adaptation is made especially challenging due to vertical camera motion between frames 85 and 180 which leads to fluctuations in the color model. The tracker needs some time to zero in on the color model of the tree until it finally stabilizes after frame 140. This experiment also shows the independence of the texture and color cues. While at certain times the color parameter has to be adapted to account for changes in the color properties of the target object, the texture parameters remain stable and vice versa.

To obtain a more controlled situation, the second type of sequences are synthetically combined images of natural textures. Figures 12 and 14 demonstrate how the integration makes the segmentation more reliable. The images consist of background and foreground textures which are real texture images obtained from Columbia University and Utrecht University reflectance and texture database [31]. A background texture moves along two paths from the upper side of the image to the bottom left of the image. On its path, it is occluded by several foreground textures. The foreground textures are selected according to their color and texture similarity. The parameter vectors are initialized with samples from the right upper corner of the background texture. The velocity vector is initially set to a non-zero value towards the left bottom corner of the image. Figures 12 and 14 show the same experiment with different paths and tracking window sizes. Figures 13(a) and (b) show the adaptation of the first parameter of the mean color vector and the first parameter of the texture model vector respectively. Figures 12 and 14 show that the target texture gets partially occluded by the foreground textures. However, the algorithm manages to continue tracking the target, without adapting to the color and texture of the foreground. Thus, this shows, how the combination of cue fusion, adaptation and prediction makes the tracking system reliable, even in the presence of occlusion.

8 Discussion

The experiments show that the combination of color and texture cues can provide more information than any of the cues alone. The main reason is their independence from each other. While color is processed based on the chromacity information, the texture parameters are calculated from the luminance. In addition the cues are extracted at different scales. Color is defined on a per pixel basis and is processed at this level. Texture however, is extracted at a region level, which allows to obtain more information about the spatial relationship of the underlying structure.

The local information supported by the neighborhood information results in a high entropy extraction per pixel. However, the different scales of the information makes the probabilistic combination difficult. In this work color and texture are both combined at a regional level. Because texture information is calculated over a set of regions, it gives more reliable results than when calculated based on a single region.

Choosing the right window element size M used for the texture parameter calculation is very important, especially for getting good representation of the parameter covariance matrix. Larger

values yield better texture parameter estimates but increase the computational cost. In general, the window element size depends on the texture scale of the target object. After the experiments and cost calculation, $M = 5$ was selected as the most typical window element size for the system.

In this work, the average of the Mahalanobis distances calculated from the parameter vectors is used. Other forms of the function (23) are possible and where investigated, but the average of distances gives the best result both in terms of performance and cost efficiency.

Although the adaptation to environmental changes makes the tracking system more flexible, it has important drawbacks. There is a risk of adapting the model parameters to wrong targets or the background which can result in a complete loss of target. However, the independence of the cues prevent to some extend the adaptation to a wrong target because changes in the environment are less likely to cause changes in both cues at the same time (Fig. 9 and 11). For example, scaling in general changes the texture of an object but leaves the object color unchanged. Furthermore, the use of texture and color cues can also aid when tracking multiple targets because the chances of targets being similar in both color and texture is less than similarity in a single cue alone. In this work additional counter measures where proposed to prevent adapting to the wrong target after the color and texture segmentation, as explained in section 6. When the tracking target and the background both show very little actual texture, the system reduces to an adaptive color tracker. Nevertheless, even the absence of a texture is modelled by the system and our two cue approach aids in situations where non-textured targets are moving in front of a textured background.

Further experiments will have to show, to what extend the adaptation of the covariance matrices are necessary. First tests indicate that the covariance matrices change little over time and that the tracking algorithm performs well, even if the covariances are not adapted. Assuming constant covariances would be a great advantage because the recalculation of the covariance matrices is computationally very expensive.

One of the difficulties faced in this work is the quantitative performance evaluation of the results because establishing the ground truth for this work is very difficult. For the segmentation results, the error and the correct percentages where not computed because the main purpose of the system is to detect the presence of the target object. A quantitative measures of the segmentation does not express the performance of the proposed tracking scheme.

It is important to note that our work so far has only been aimed at investigating the feasibility of fusing texture and color for target tracking. Depending on the parameters, especially the window

element size M, the algorithm runs at five frames per second on a 200 MHz SGI O_2. A careful implementation of the algorithm will be able to run at near real-time speed on the hardware described above. To achieve real-time performance, additional strategies can be implemented. For example, a compromise between robustness and performance could be achieved by performing a texture analysis on every second frame while doing the color segmentation for every frame.

The experiments have shown that the texture segmentation gives very high error distances near the boundaries. The cause of this might be unbalanced textures at these points. This observation could in the future be used for controlling the boundary location of the target object. The integration method can be improved by changing the thresholds τ_c and τ_t adaptively over time, for example by using threshold histograms. The Kalman filter can also be used for updating the parameter vectors but this would require modeling the color and texture vector changes parametrically.

9 Conclusion

This paper presents a novel technique for combining texture and color segmentation in a manner that makes the segmentation robust under varying environmental conditions while keeping the computation efficient for real-time target tracking. A probabilistic basis is used for combining texture and color cues using the Gibbs Markov Random Field for the texture and the 2D Gaussian Distribution for the color segmentation. Both the color and texture segmentation is adapted over time in a manner that increases the probability of correct segmentation while not "drifting" with the changing background. Extensive experiments with both static and dynamic image sequences were used for establishing the feasibility of the proposed approach. The work demonstrates the advantages of fusing multiple cues within a stochastic formulation while providing a scheme for practical implementation of target tracking applications.

Acknowledgments

This work was supported in part by the following grants: National Science Foundation CAREER Grant IIS-97-33644, NSF Grant IIS-0081935, and U. S. Army Research Laboratory Cooperative Agreement No. DAAL01-96-2-0003.

References

[1] S.J McKenna, S. Gong, Y. Raja, Modelling facial colour and identity with Gaussian mixtures, Pattern Recognition, 31(12)(1998), pp. 1883–1892.

[2] R. Schuster, Color object tracking with adaptive modeling. Proc. IEEE Symposium on Visual Languages, 1994, pp. 91–96.

[3] J. Yang, W. Lu, A. Waibel, Skin-color modeling and adaptation. Proc. Third Asian Conference on Computer Vision, 1998, pp. 687–694.

[4] J.C. Terrillon, M. David, S. Akamatsu, Automatic detection of human faces in natural scene images by use of a skin color model and invariant moments. Proc. International Conference on Automatic Face and Gesture Recognition, 1998, pp. 112–117.

[5] W. Sharbek, A. Koschan, Color segmentation survey, Technical Report, Univ. of Berlin, 1994.

[6] S.E. Umbaugh, R.H. Moss, W.V Stoecker, A.G. Hence, Automatic color segmentation algorithms, IEEE Engineering in Medicine and Biology, 12(3)(1993), pp. 75–82.

[7] P.W. Power, R.S. Clist, Comparison of supervised learning techniques applied to color segmentation of fruit images. Proc. of SPIE, 1996, pp. 370–381.

[8] A.G. Hence, S.E. Umbaugh, R.H. Moss, W.V Stoecker, Unsupervised color image segmentation, IEEE Engineering in Medicine and Biology, 15(1)(1996), pp. 104–111.

[9] M.J. Jones, J.M. Rehg, Statistical color models with applications to skin detection, Technical report, Cambridge Research Labaratory, 1998.

[10] R. Kieldsen, J. Kender, Finding skin in color images. Proc. International Conference on Automatic Face and Gesture Recognition, 1996, pp. 312–317.

[11] D.A. Forsyth, A novel approach to color constancy, International Journal of Computer Vision, 5(1)(1990), pp. 5–36.

[12] G. Healey, D. Slater, Global color constancy: recognition of objects by use of illumination-invariant properties of color distributions, Journal Optical Society of America A, 11(11)(1994), pp. 3003–3010.

[13] W.C. Huang, C.H. Wu, Adaptive color image processing and recognition for varying backgrounds and illumination conditions, IEEE Transactions on Industrial Electronics, 45(2)(1998), pp. 351–357.

[14] G. Paschos, K.P. Valavanis, A color texture based visual monitoring system for automated surveillance, IEEE Trans. on System, Man, and Cybernetics, 29(1)(1999), pp. 298–307.

[15] T.R Reed, J.M Hans Du Buf. A review of recent texture segmentation and feature extraction techniques, CVGIP: Image Understanding, 57(3)(1993), pp. 359–372.

[16] J. Goutsias, Mutually compatible Gibbs images: Properties, simulation and identification, IEEE Transaction on Information Theory, 35(6)(1989), pp. 1233–1249.

[17] M. Schroder, H. Rehrauer, K. Seidel, M. Datcu, Spatial information retrieval from remote-sensing images - part 2: Gibbs-Markov random fields, IEEE Transaction on Geoscience and Remote Sensing, 36(5)(1998), pp. 1446–1455.

[18] S.Z. Li, Markov Random Field Modeling in Computer Vision, Springer, first edition, 1995.

[19] K.V. Mardia, G.K. Kanji, Statistics and images, Abingdon, first edition, 1993.

[20] H. Derin, H. Elliott, Modeling and segmentation of noisy and textured images using Gibbs random fields, IEEE Transactions on Pattern Analysis and Machine Intelligence, 9(1)(1987), pp. 39–55.

[21] G.R. Cross, A.K. Jain, Markov random field texture models, IEEE Transactions on Pattern Analysis and Machine Intelligence, 5(1)(1983), pp. 25–39.

[22] D.K. Kumar, G. Healey, Markov random field models for unsupervised segmentation of textured color images, IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(10)(1995), pp. 939–954.

[23] M.P.D. Jolly, A. Gupta, Color and texture fusion: application to aerial image segmentation and GIS updating. Proc. Third IEEE Workshop on Applications of Computer Vision, 1996, pp. 2–7.

[24] R. Murrieta-Cid, M. Briot, N. Vandapel, Landmark idenfication and tracking in natural environment, IEEE/RSJ International Conference on Intelligent Robots and Systems, 1998, pp. 738–740.

[25] D.S. Jang, H.I. Choi, Moving object tracking by optimizing models. Proc. International Conference of Pattern Recognition, 1998, pp. 738–740.

[26] Y.N. Deng, B.S. Manjunath, Netra-V: Toward an object-based video representation. IEEE Trans. Circuits and Systems for Video Technology, 8(5)(1998), pp. 616–627.

[27] E. Ozyildiz, Adaptive texture and color segmentation for tracking moving objects, Master's thesis, Pennsylvania State University, 1999.

[28] M.J. Swain, D.H. Ballard, Color indexing, International Journal of Computer Vision, 7(1)(1991), pp. 11–32.

[29] E. Saber, A.M. Tekalp, Integration of color,shape, and texture for image annotation and retrieval. Proc. International Conference on Image Processing, 1996, pp. 851–854.

[30] T. Pavlidis, Y.T. Liow, Integrating region growing and edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(3)(1990), pp. 225–233.

[31] S.K. Nayar, K.J. Dana, B.V. Ginneken, J.J. Koenderink, Columbia-Utrecht reflectance and

texture database, http://www.cs.columbia.edu/CAVE/curet, 1999.

Figure 1

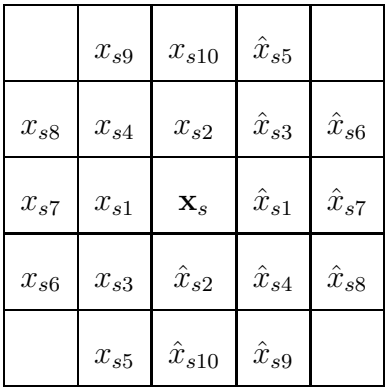


Fig. 1. Definition of the local neighborhood around the pixel site x_s.

Figure 2

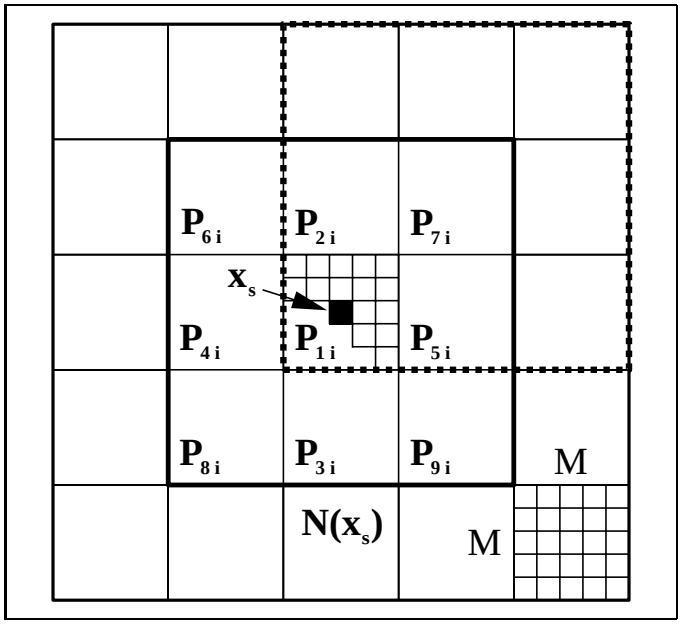


Fig. 2. The sub center window $N(x_s)$ with $N = 9$ neighbor windows. Each window element has size $M \times M$ pixels. The color parameter vector is estimated based on the region $N(x_s)$. For the texture analysis, a parameter vectors are estimated for each of the neighbor windows and subsequently averaged.

Figure 3

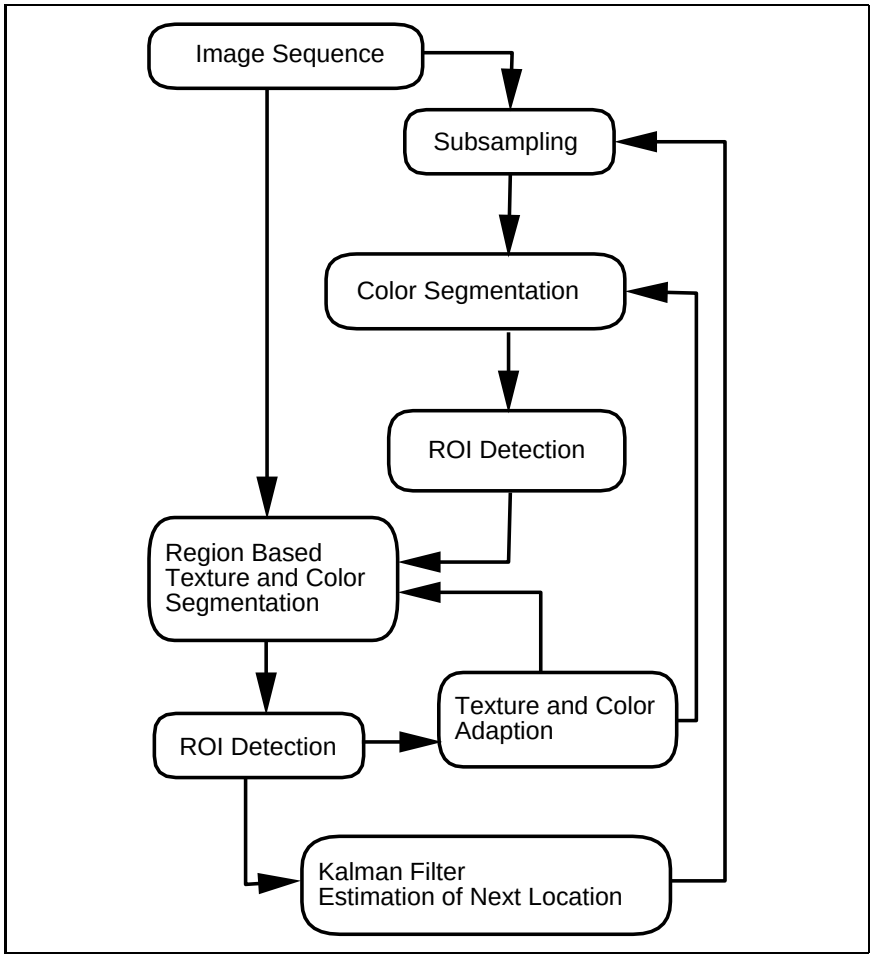
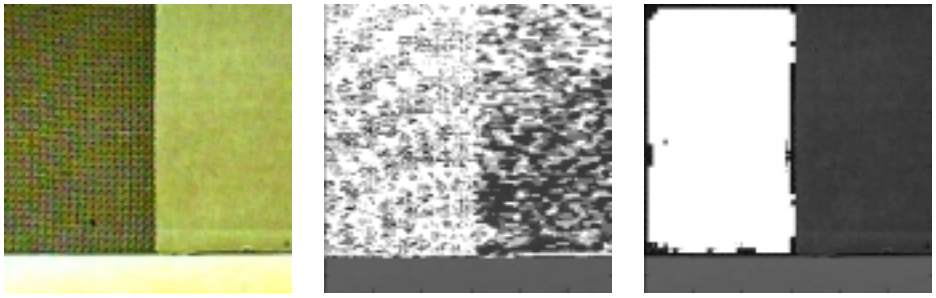


Fig. 3. Schematic overview of the system details for tracking moving targets with adaptive color and texture segmentation.

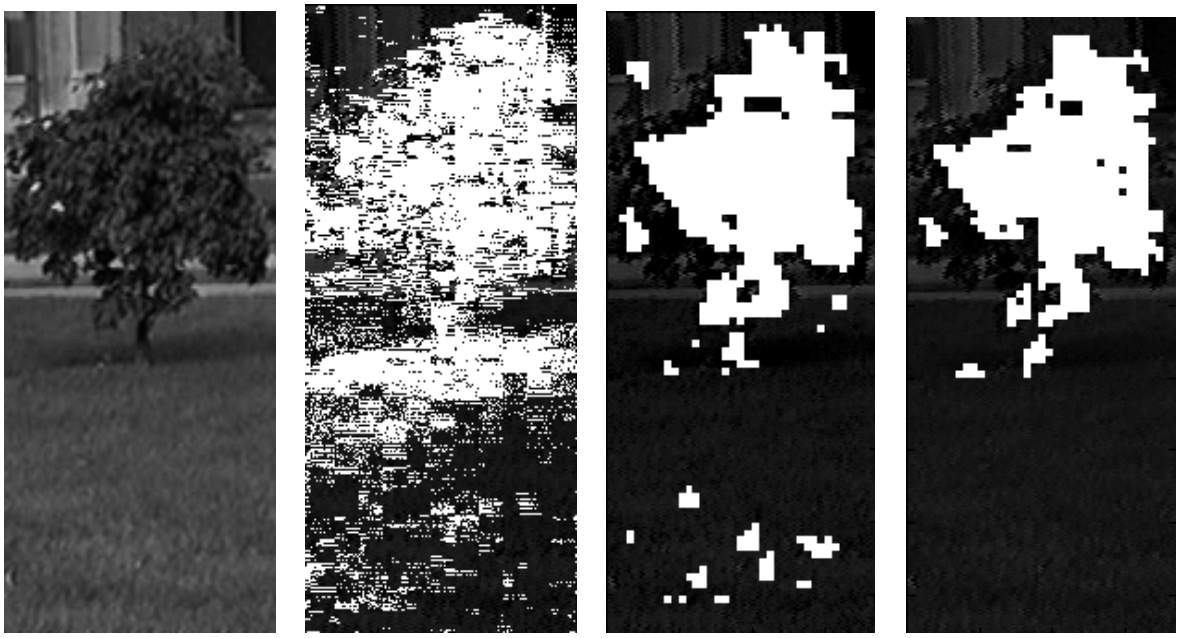
Figure 4



(a) (b) (c)

Fig. 4. Experiment performed on a natural image containing three wooden textures (a). The result of performing a color segmentation can be seen in image (b). (c) shows the segmentation that is obtained by fusing texture and color.

Figure 5



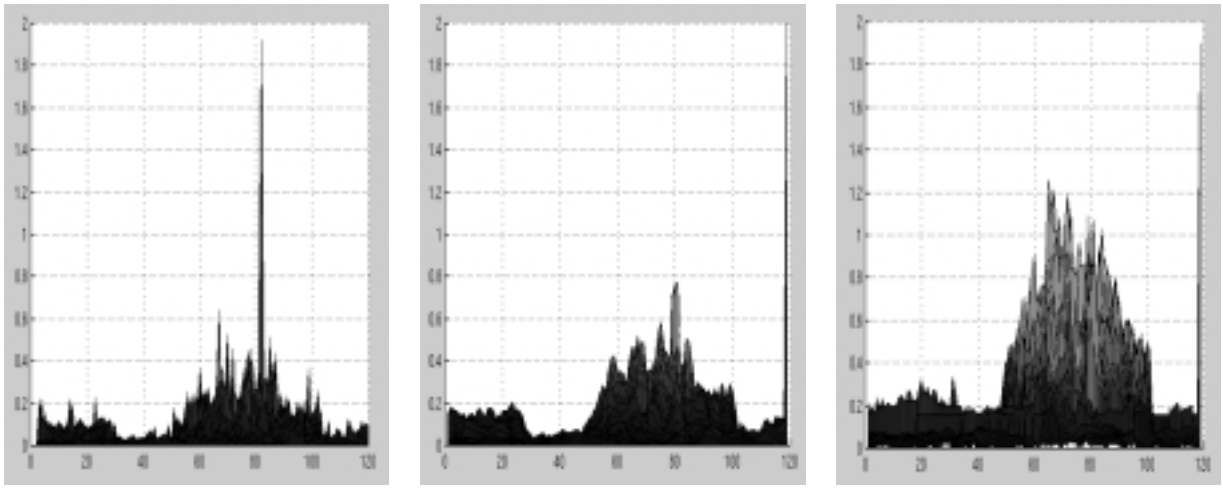
(a) (b) (c) (d)

Fig. 5. Experiment performed on a natural image containing natural textures (a). The result of performing a color segmentation can be seen in image (b). (c) shows the segmentation that is obtained by using texture alone. The fusion of texture and color results in image (d).

Figure 6

Fig. 6. Image of a football that is used for examining the discrimination power of the texture and color fusion approach. The football has a spatially varying texture and color due to its shape and the lightning condition.

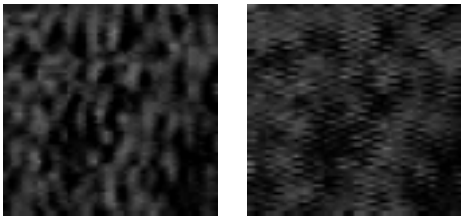
Figure 7



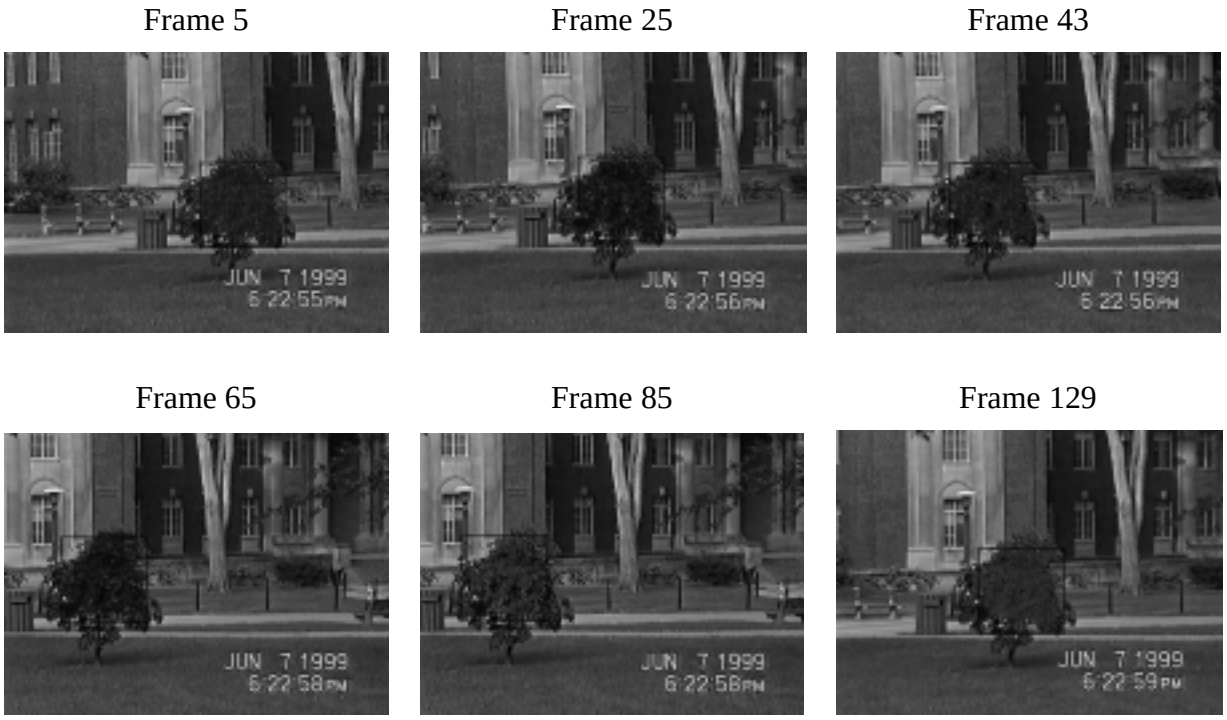
(a) (b) (c)

Fig. 7. Similarity measures after segmentation based on (a) one texture sample region obtained from the sub-centered window, (b) the average of the texture estimates based on the nine neighbor windows, (c) the combination of color and texture.

Figure 8



(a) (b)



(c)

Fig. 8. Image showing a section of the tree for a stationary camera (a) and a moving camera (b). The full tree tracking sequence can be seen in (c).

Figure 9

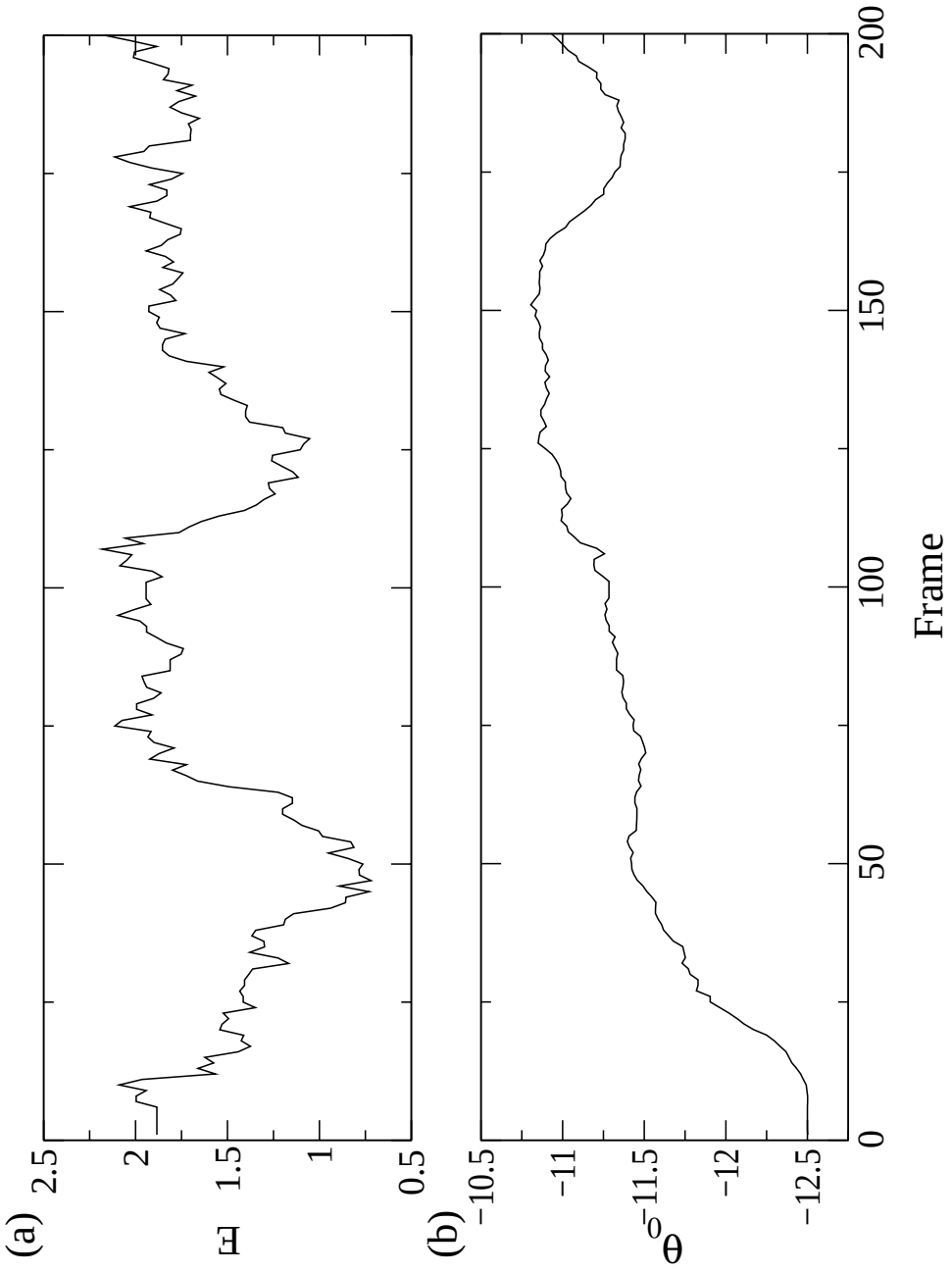


Fig. 9. The adaptation of the E component of the color parameter vector (a) and the first component of the texture parameter vector (b) during the tracking of the tree in Figure 8.

Figure 10

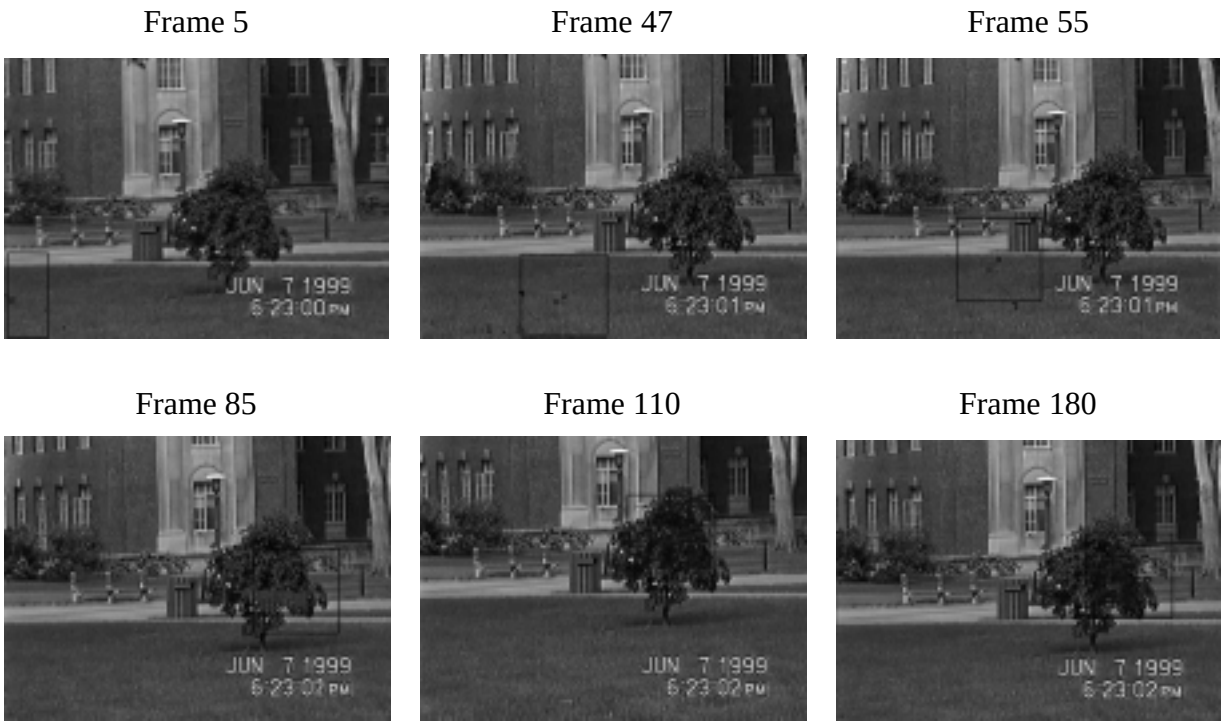


Fig. 10. Sequence showing the capture and subsequent tracking of the tree. The sequence is the same as in Figure 8.

Figure 11

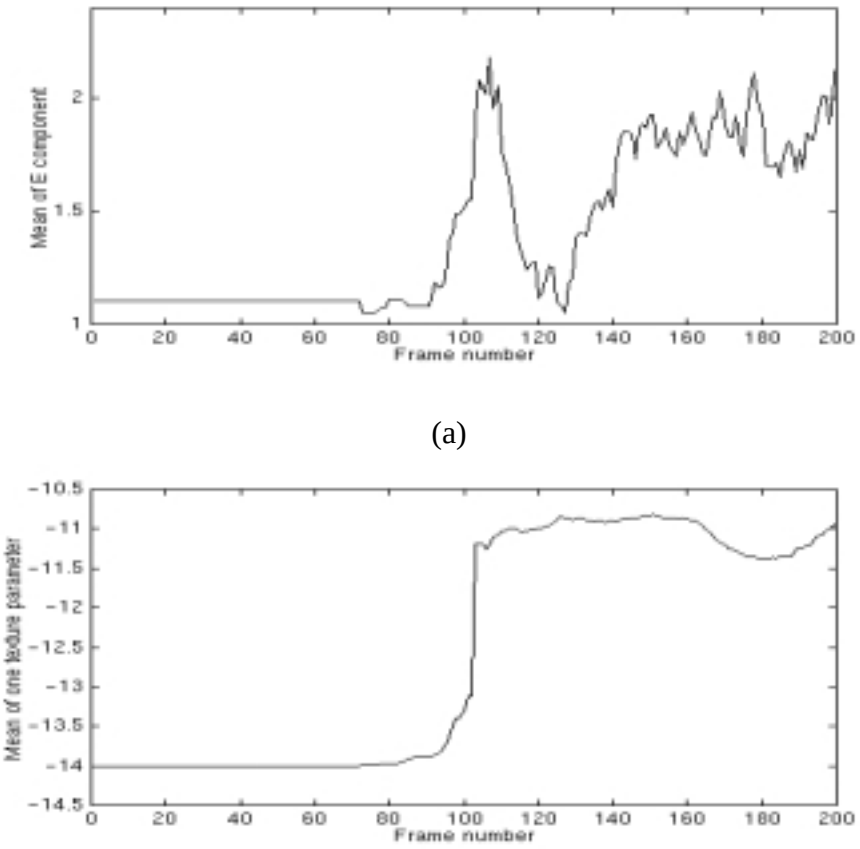
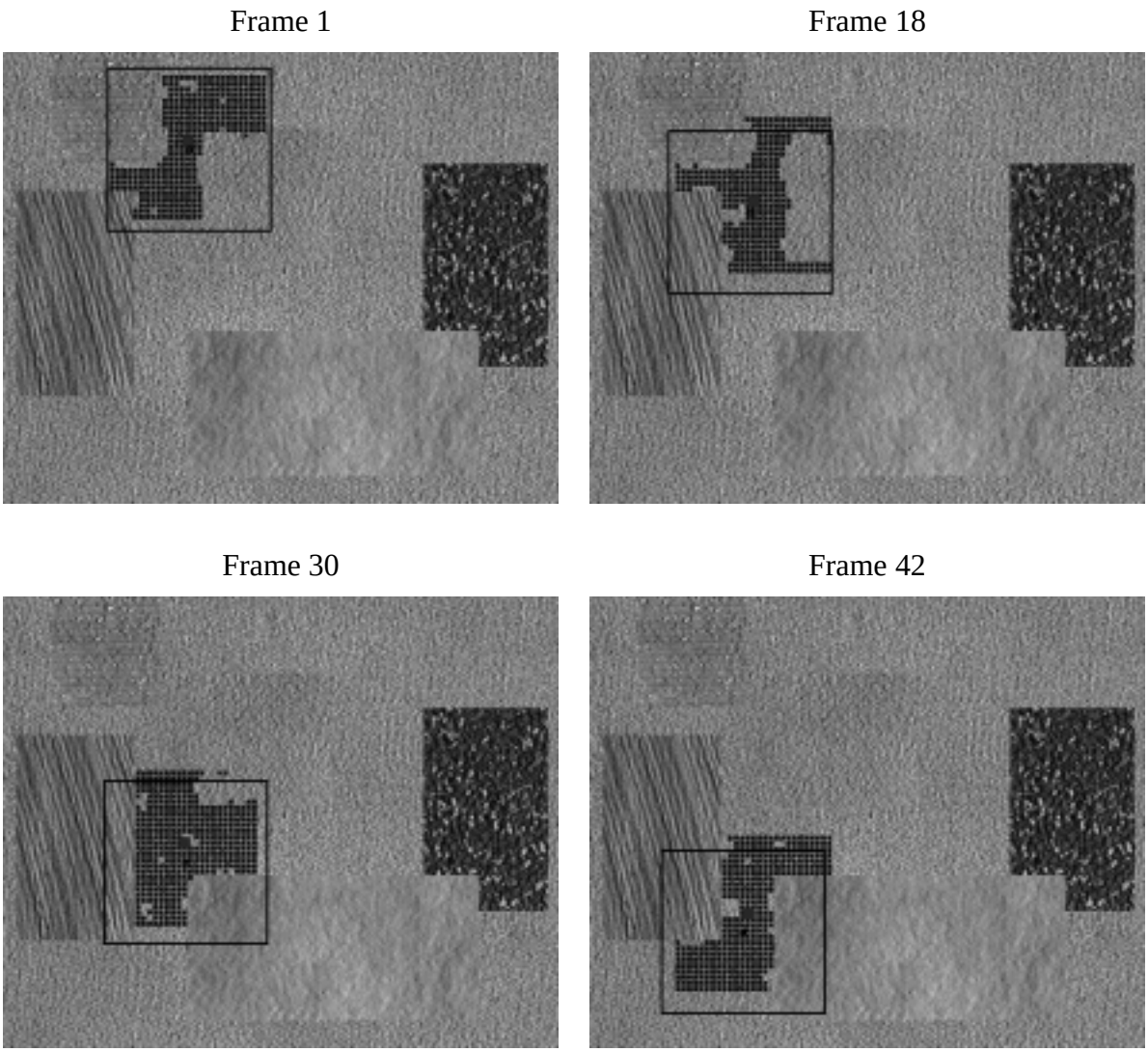


Fig. 11. The adaptation of the E component of the color parameter vector (a) and the first component of the texture parameter vector (b) during the tracking of the tree in Fig. 10. It can clearly be seen how the parameters are not adapted while the tree is not captured by the tracker.

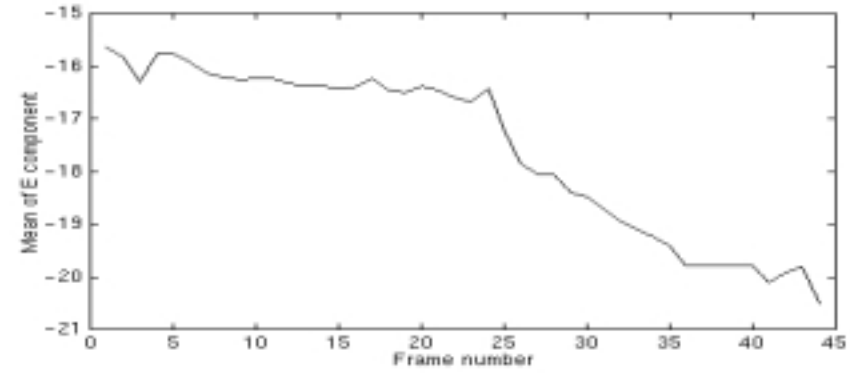
Figure 12



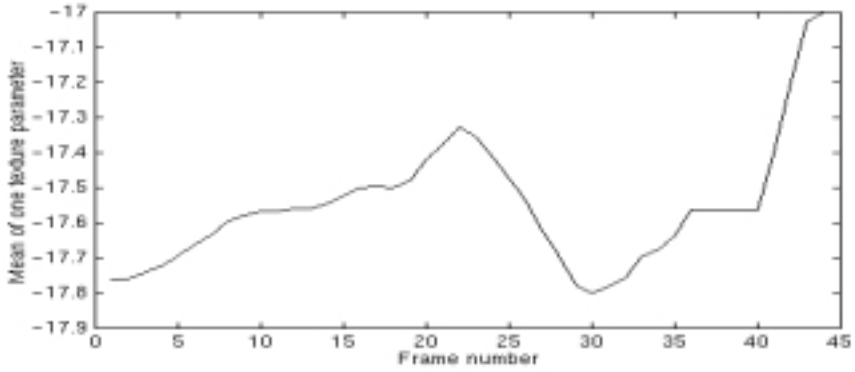
(a)

Fig. 12. Synthetic dynamic tracking sequence of a large textured object partially occluded by foreground textures.

Figure 13



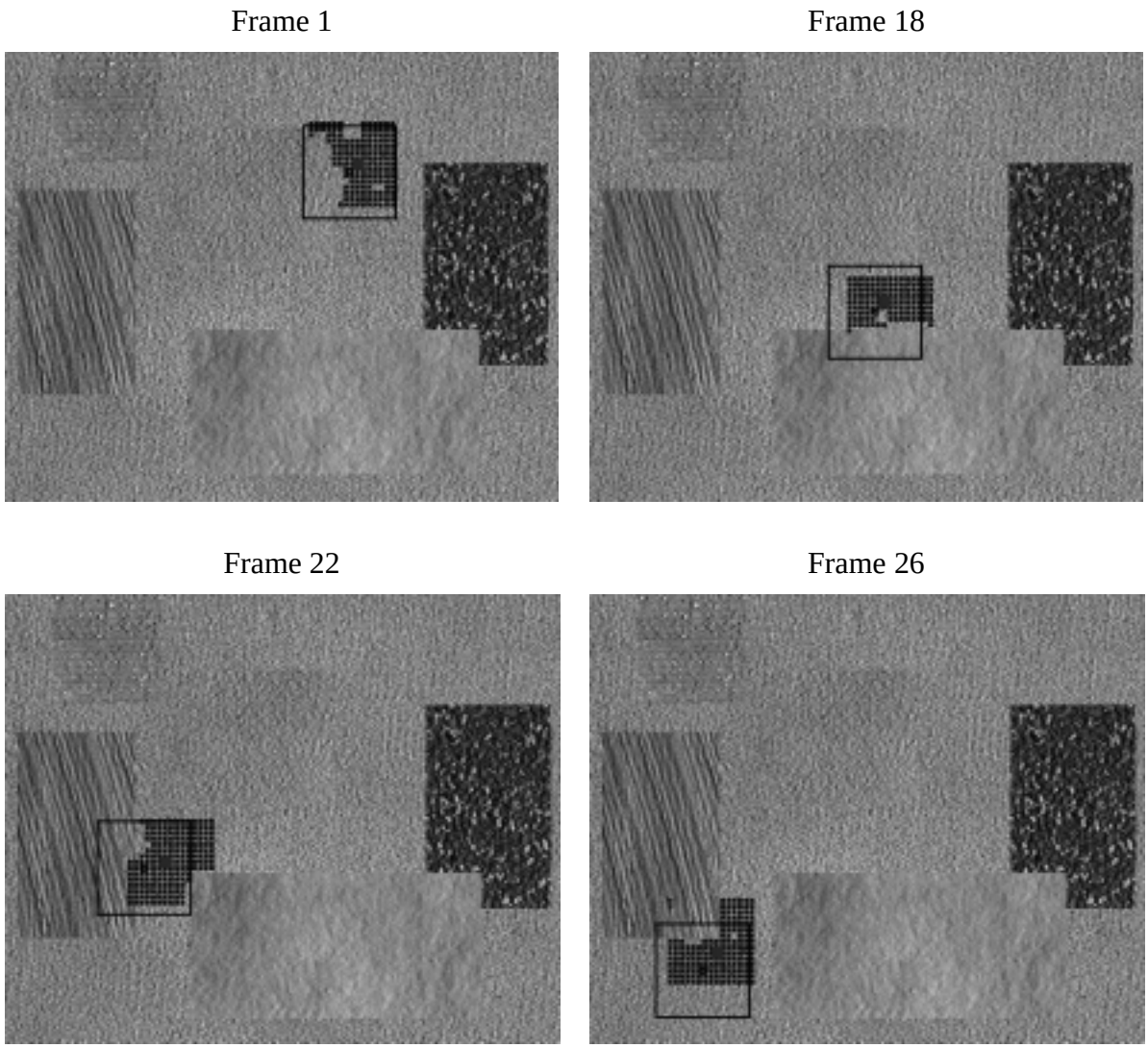
(a)



(b)

Fig. 13. The adaptation of the E component of the color parameter vector (a) and the first component of the texture parameter vector (b) during the tracking of the texture in Fig. 12.

Figure 14



(a)

Fig. 14. Synthetic dynamic tracking sequence of a small textured object partially occluded by foreground textures.