OBJECT TRACKING BY ADAPTIVE MODELING
A. Rares and M.J.T. Reinders
Faculty of Information Technology and Systems
Delft University of Technology
e-mail: {A.Rares, M.J.T.Reinders}@its.tudelft.nl

ABSTRACT
This paper addresses the problem of object tracking in image sequences. The approach taken is based upon adaptive statistical models. An object selected in a frame by a user is tracked throughout the sequence by using a blob-like description of its features. The object features are continuously updated by using the on-line version of the Expectation-Maximization algorithm. The proposed object description results in a flexible representation.

1. INTRODUCTION
To restore old movie material with high quality one has to deal with numerous artifacts [Roo99]. Restoration techniques for these artifacts can, however, be complementary in their behavior (e.g. one sharpens and the other smoothens) so that it becomes important to select the appropriate restoration technique locally and automatically. In such a restoration framework, a restorer selects regions or objects that must be restored. The restoration technique can then be chosen such that it is most effective for the selected region. For a successful restoration these selected items have to be tracked throughout the sequence. This paper addresses the tracking of such regions.

In this paper, we present a system that tracks objects in an image sequence. The proposed system tracks both the position and the shape of the object. In the first frame, the user marks the object by drawing a polygon around it. Calculating the statistics of the respective areas then initializes the models of the object and of the background (remaining part of the scene). For the subsequent frames, these object models are then analysed and updated according to the tracked data. The tracked items (object or the background) are represented statistically and approximated with a mixture of Gaussians. By including position into the feature set a coarse blob-like description of the shape of the region is readily achieved. The analysis consists of two phases: segmentation and learning. In the first phase, we calculate support maps for the tracked objects. The second phase uses the support maps for updating the statistics of the object models. Because not much data of the objects is available, one needs statistical techniques that can deal with incomplete data sets. Further, we need flexible learning techniques that can deal with rapidly changing object appearances.

A number of algorithms developed in the frame of statistical mathematics are suitable for such computer vision based object tracking. For achieving the real-time performances that are required only a part of these algorithms can be employed, such as the iterative Expectation Maximization (EM) algorithm [Dem77], Icondensation [Isa98], or Ransac [Boi81]. In this paper, we use the on-line version of the EM algorithm as proposed in [Oli97].

2. THE FEATURE SPACE
When tracking an object in an image sequence, we look for features that can distinguish it from other objects. We have chosen a feature set that describes position, normalized color, and Gabor-based textures [Ham99].

The position feature consists of the two dimensions of the input image. The color feature includes normalized red and green (calculated in the RGB color space). We used normalized colors for their relative invariance to lighting changes.

The texture feature is represented by a multi-dimensional set consisting of the responses of the Gabor wavelets, called “jets”. For a pixel \( x=(x, y) \) and a pixel value \( I(x) \), the jets \( J(x) \) are calculated using a convolution, as described in [Ham99]:

\[
J_j(x) = \int I(x') \psi_j(x-x') d^2 x'
\]  

(1)

with a family of Gabor kernels \( \psi_j(x) \):

\[
\psi_j(x) = \exp \left[ -\frac{1}{2} \left( \frac{(x \cos \theta_j + y \sin \theta_j)^2}{\sigma_x^2} + \frac{(-x \sin \theta_j + y \cos \theta_j)^2}{\sigma_y^2} \right) \right]
\]

\[
\cdot \exp \left[ \frac{2\pi i (x \cos \theta_j + y \sin \theta_j)}{k_x} \right]
\]

(2)
where we used 3 different frequencies ($\nu=\{0, 1, 2\}$) and 4 orientations ($\mu=\{0, 1, 2, 3\}$):

$$k_{\nu} = 2^{y+2} \pi, \varphi_{\mu} = \mu \pi/4, j = \mu + 4\nu$$

(3)

$$\sigma_x = 0.5\lambda, \quad \sigma_y = 0.5\lambda$$

Thus, in our implementation, a jet $J$ is defined by a set of $3 \times 4 = 12$ complex values for each pixel. Although we used a number of 12 values, this number can vary, depending on the chosen number of directions and frequencies. Each calculated jet value can be written as:

$$J_j = a_j \exp(i \varphi_j)$$

(4)

where $a_j$ represents the magnitude, and $\varphi_j$ represents the phase. The magnitude does not change quickly with translation, distortion, rotation, and scaling, whereas the phase changes rapidly with translation. Therefore, we have chosen to include in our feature set only the magnitudes. The magnitudes were normalized with respect to their energy:

$$a'_j = \frac{a_j}{\sqrt{\sum_i a_i^2}}$$

(5)

In Fig. 1, one can see how the responses of each Gabor kernel make up the feature set for the texture.

The complete feature set is built by concatenating all features. In the $N$-dimensional feature space, a tracked object $o_j$ is represented by a mixture $\Theta_j$ of $N$-dimensional Gaussians, $\theta_k$ (with mean $\mu_k$ and covariance $\Sigma_k$). Increasing the number of Gaussians in such a description allows for a more specific representation (the coarsest description is thus achieved by using only one Gaussian).

3. OBJECT TRACKING USING THE EM ALGORITHM

Having such statistical representation, it becomes possible to classify each pixel $x$ into one of the objects (say, $o_j$) by using a Maximum A Posteriori (MAP) decision that makes use of the Bayes rule:

$$x \rightarrow o_j,$$

$$\lambda = \arg \max_{\lambda} p(\Theta_{\lambda} | x) = \arg \max_{\lambda} p(x | \Theta_{\lambda})$$

(6)

with

$$p(x | \Theta) = \sum_{j=1}^{M_i} w_j p(x | \theta_j), i = 1..K; \sum_{j=1}^{M_i} w_j = 1$$

(7)

where $\Theta = \{\theta_j | j=1..M_i\}$ denotes the object model for the $i$-th object $o_i$, $M_i$ is the number of components (Gaussians) of that object, $\theta_j$ is the $j$-th component of object model $\Theta$, $p(x|\theta_j)$ is the probability density function (pdf) corresponding to component $\theta_j$, and $w_j$ is the weight of this component with respect to the object model. In Eq. (6) we assumed that all objects are equally probable.

The steps of the proposed algorithm are then as follows. The statistics of each object are initialized in the first frame by letting the user select it manually with a polygon. After this initial phase, the algorithm performs two steps for each frame. First, based on the available statistical model (coming from the initialization phase, or from the previous frame), the MAP decision is made for every pixel of the new image to be processed. In this way, support maps for the tracked object(s) are created (indicating that these pixels belong to the specified object). Together with the original image, these support maps offer new information about the statistics of the region to be tracked. In the next step, the statistics of each object model are updated on the basis of these support maps by using the EM learning algorithm. The updated object models are finally used as estimates when processing the next frame.

![Figure 1. The responses of the Gabor kernels constitute the jets](image1.png)

![Figure 2. The analysis loop of the object tracker](image2.png)

These steps of the algorithm are illustrated in Fig. 2. The scheme is made more efficient by incorporating a focus-of-attention mechanism, namely the motion between the frames (a simple thresholded frame...
difference) which makes the distinction between moving objects and still background.

For the learning step, we used update formulas derived from [OH97]. If \( n \) pixels of the current frame already have been processed, then, at the \( n+1 \)-th pixel, the updating goes as follows. First, we know to which object the \( n+1 \)-th pixel belongs (from the support maps that are initially created), say this is denoted by label \( \lambda \). Then, for each component \( k \) of object model \( \Theta_k \) we calculate the “responsibility” \( h_{jk}^{n+1} \):

\[
h_{jk}^{n+1} = \frac{w_{jk}^{n} p(x^{n+1} | \theta_{jk}^n)}{\sum_{j=1}^{N_k} w_{jk}^{n} p(x^{n+1} | \theta_{jk}^n)}, \quad k = 1, \ldots, N_i
\]

where \( \theta_{jk} \) or \( \theta_{kj} \) represent the \( k \)-th and \( j \)-th component of the object model \( \Theta_k \), respectively. The responsibility \( h_{jk}^{n+1} \) can be interpreted as the normalized contribution of the \( k \)-th component to object model \( \Theta_k \) based on the measured values of pixel \( x_{n+1} \). Given these responsibilities, we can update the model’s statistics (for all components \( \theta_{jk} \), including their weights \( w_{jk}^{n+1} \)):

\[
w_{jk}^{n+1} = w_{jk}^{n} + \frac{h_{jk}^{n+1} - w_{jk}^{n}}{n + 1}
\]

\[
\mu_{jk}^{n+1} = \mu_{jk}^{n} + \frac{h_{jk}^{n+1}}{(n + 1)w_{jk}^{n+1}}(x^{n+1} - \mu_{jk}^{n})
\]

\[
\Sigma_{jk}^{n+1} = \Sigma_{jk}^{n} + \frac{h_{jk}^{n+1}}{(n + 1)w_{jk}^{n+1}}((x^{n+1} - \mu_{jk}^{n})^T(x^{n+1} - \mu_{jk}^{n}) - \Sigma_{jk}^{n})
\]

Some special comments have to be made about the initial and maximum number of objects in the system, and the initial and maximum number of components for each object. These values can be set in a configuration file. The current number of objects and/or components varies as follows:

**Number of objects.** In the MAP step, if a pixel is not well explained by any of the current objects (i.e. the Mahalanobis distance to any of the current object models exceeds a certain threshold), an object can be added to the system (note that if one would require a constant number of objects this can be skipped). This object has one component, having as mean values the calculated features for the current pixel. The covariance matrices are set to some predefined value.

**Number of components.** In the EM step, if a pixel that was assigned to an object is too far from any of the object’s components (i.e. the Mahalanobis distance to any of the current object’s components exceeds another threshold), then a new component can be added to that object, with mean values the calculated features for the current pixel, and some predefined covariances.

**Cleanup.** After each frame has been processed, components that cannot explain at least one pixel, or have a very small weight, are purged, in order to avoid computational overload. Objects left with no components are then deleted as well.

### 4. RESULTS AND DISCUSSION

We have performed several experiments on different image sequences, and different parameters. Fig. 3 shows an example where we used only position and color features. The ellipse represents a cross-section through the two-dimensional Gaussian representing the positional feature set (\( x \) and \( y \) position of the object pixels). The cross-section is taken at about 2\( \sigma \) distance from its mean.

The black ellipse corresponds to the background. The ellipses are initialized with large values (see the first frame, where the background’s ellipse falls actually outside the image), but they rapidly shrink to adapt to the tracked items. The tracking is quite accurate, and robust, i.e. even partial occlusions are handled. We should, however, emphasize that in this particular case, the sequence is relatively unconstrained, i.e. there is no clutter and the background is not moving.

Fig. 4 shows an example of a result when using a more difficult sequence, where the content is harder to distinguish, and there are scene exits, occlusions and appearances in the scene. Again, we used only color and position as object features.

The toy train sequence consists of a locomotive, and three wagons. Although we initially indicated only two wagons as objects of interest (Fig. 4a), in the second frame, the tracker has correctly identified that there is a preceding moving object (the locomotive) that is different in color.

As we can see, the tracking performs satisfactorily as long as the situation is relatively clear. At some point in the image sequence, a third wagon appears (Fig. 4b and 4c). At that moment, the last ellipse tends to extend to the third wagon because of its proximity and similarity in color between parts of the last two wagons (Fig. 4c). Meanwhile, the locomotive enters the tunnel (Fig. 4d) and the ellipse which was representing it now extends over the rest of the wagons. This happens, because the lower parts of these wagons have again a similar color. Having a less confusing task, the middle
ellipse follows its wagon quite well. From this experiment, we conclude that the tracking scheme is quite adaptive which enables it to quickly fit the changing appearances. However, in this sequence, this strong feature works somewhat against it since, due to the appearance and disappearance of the objects, the tracker becomes distracted.

In our opinion, it turns out that Gabor-based textures cannot be used in highly cluttered images, since they cannot make a clear distinction between the objects. In fact they seem to represent the borders of the object instead of regional aspects. This hampers the tracking process, resulting in wrongly placed ellipses.

We have performed another experiment on the same sequence, including Gabor-based textures this time. The results can be seen in Fig.5.

It seems that using the additional Gabor features does not improve the object representation and consequently the tracking results. Similar results can be observed for the “person” sequence (Fig. 6), although its disturbing influence is less then for the train sequence.

In our opinion, it turns out that Gabor-based textures cannot be used in highly cluttered images, since they cannot make a clear distinction between the objects. In fact they seem to represent the borders of the object instead of regional aspects. This hampers the tracking process, resulting in wrongly placed ellipses.

5. CONCLUSIONS AND FUTURE WORK

In the view of our current results, the tracking performs satisfactorily for “clear” situations, i.e. the regions are described relatively accurate and the system rapidly adapts to the tracked items. The flexible object model can cope relatively easy with changing object appearances. Unlike motion estimation techniques, this method can adapt to changes of lighting conditions, to deformation and/or occlusion of objects, and it can overcome the problems posed by noise, up to a certain limit.

However, there are situations where the tracking fails, such as the heavily cluttered images. Here, even the texture recognition does not help in improving the results. The latter has mainly to do with the high adaptivity of the tracker.

Future work should include some more experiments in the area of textures, in order to gain more insight on how we can describe and identify textures. Also, as an addition to the current work, the feature set can be extended to include other features, such as object motion. As a final remark, by representing the feature set of the objects statistically, these statistics can be used for selecting the appropriate restoration techniques to be applied to these areas, which needs to be explored in future work.

6. REFERENCES


