EM Clustering of Incomplete Data Applied to Motion Segmentation

King Yuen Wong[1] , Lu Ye[2] , Minas E. Spetsakis[3]
Dept. of Computer Science[1,3] , Dept. of Mathematics[2],
Centre for Vision Research[1,3],
York University,
4700 Keele Street, Toronto, Ontario, Canada M3J 1P3
Emails: {kywong,minas}@cs.yorku.ca, lye@mathstat.yorku.ca

Abstract

Many clustering problems in Computer Vision group data points that are the result of statistical estimation and these data points can have a great amount of uncertainty. Motion segmentation by clustering of optical flow is such an example because very often optical flow cannot be estimated without significant uncertainty. We present a EM based clustering algorithm for incomplete data and we apply it to the problem of motion segmentation. The input to the algorithm are the velocity likelihoods and the number of clusters. The algorithm is mathematically very elegant because it does not impose any constraints on the velocity likelihood thus multi-modal likelihood is modeled without difficulty. Coupled with a sophisticated correlated image noise model, the algorithm can handle substantial deviations from the intensity constancy assumption. Experiments with real image sequences show excellent results.

1. Introduction

The process of grouping pixels having similar motion characteristics is called *motion segmentation*. A popular approach for describing motion similarity within a segment/layer [12] is by their optical flow. The computation of optical flow at a pixel is an under-constrained problem and the classical solutions [2] almost exclusively use constraints from neighboring pixels by assuming one of the several smoothness constraints which usually do not hold on object boundaries. Motion segmentation based on optical flow is thus a chicken and egg problem: In order to compute flow accurately, we need to know motion boundaries but locating the motion boundaries amounts to doing segmentation which requires flow as input. Our approach subscribes to the paradigm [4] that does motion segmentation without computing the full optical flow first.

Some of the most successful techniques start [4, 1] by computing flow or normal flow and then clustering the flow vectors. This clustering is applied to a subset of the image: anything ranging from a set of selected points to every pixel within a certain region. One of the difficulties with this approach is that due to the aperture problem, the uncertainty in the flow estimation can be high. In other words, our knowledge of the data is incomplete. In this paper, we apply Expectation Maximization to the clustering of incomplete data and develop a clustering algorithm that takes as input the likelihood of the displacement and the number of clusters.

2. Motion Segmentation

We identify small textured patches and for each patch, we estimate its likelihood of observing a set of flow velocities. Our velocity likelihood representation allows for modeling of multi-modal distribution and hence can account for temporal aliasing (multiple matches) and possibly motion discontinuities within a patch. We introduce a clustering method to fit multiple affine parameters to their flows and a simple statistical technique to obtain the affine flow as a by-product of clustering by maximizing the conditional probability of the flow given the location of the patch.

We then compute dense segmentation by warping the image with the computed motion models and identify the spatial support of each layer by statistical hypothesis testing using Mahalanobis distance as the underlying statistic. Mahalanobis distance can be seen as weighted SSD (Sum of Squared Differences) for correlated data.

Section 3 describes the evaluation of flow likelihood on features. Section 4 and 5 describes clustering of flow by EM based clustering algorithm that handles incomplete data. Section 6 shows the derivation of affine parameters for the layers as a by-product of EM clustering. Section 7 presents the dense segmentation technique and the correlated noise model. Section 8 presents the results of motion segmentation on real image sequences.

3. Evaluation of Velocity Likelihood

We identify feature points that are surrounded by adequate texture by using an algorithm proposed by Tomasi and Kanade [11]. The algorithm identifies a feature point as one for which the smallest eigenvalue for matrix M are bigger than a certain threshold λ_t where

$$M = \begin{bmatrix} E_{xx} & E_{xy} \\ E_{xy} & E_{yy} \end{bmatrix}, \tag{3.1}$$

and $E_{xx} = \int I_x^2$, $E_{xy} = \int I_x\, I_y$, $E_{yy} = \int I_y^2$. I is the image and I_x, I_y, are the spatial derivatives of the image. Out of all the points in the image that satisfy this condition we randomly select a number of them with preference to the ones with large smallest eigenvalue.

At each feature point i, we compute its likelihood of observing a set of flow velocities which is then input to our clustering algorithm that can handle incomplete data. Specifically, we retain a patch Im_{1i} centered around feature point i from image Im_1 and also retain the whole image Im_2. So our data is $D_i = \{ \mathrm{Im}_{1i}, \mathrm{Im}_2 \}$. The likelihood of the flow is then

$$L(u_i|D_i) = P(D_i|u_i) = P(\mathrm{Im}_{1i}, \mathrm{Im}_2\, |u_i) =$$

$$P(\mathrm{Im}_2\, |u_i)P(\mathrm{Im}_{1i}\, |\, \mathrm{Im}_2, u_i) = P(\mathrm{Im}_2)P(\mathrm{Im}_{1i}\, |\, \mathrm{Im}_2, u_i)$$

assuming that the static image Im_2 is independent of the flow u_i. Since $P(\mathrm{Im}_2)$ is constant

$$L(u_i|D_i) \propto P(\mathrm{Im}_{1i} \mid \mathrm{Im}_2, u_i) = P(\Delta\,\mathrm{Im}_i(u_i))$$

where $\Delta\,\mathrm{Im}_i(u_i)$ is the difference between the patch Im_{1i} and the corresponding patch on Im_2 displaced by u_i. If we treat $\Delta\,\mathrm{Im}_i(u_i)$ as a vector then we can write

$$L(u_i \mid D_i) = \frac{1}{\sqrt{(2\pi)^{k^2}|C_i|}}\, e^{-\frac{\Delta\,\mathrm{Im}_i(u_i)^T C_i^{-1} \Delta\,\mathrm{Im}_i(u_i)}{2}} \tag{3.2}$$

where C_i is the covariance matrix and the patch size is $k \times k$ pixels. The quantity $\Delta\,\mathrm{Im}_i(u_i)^T C_i^{-1} \Delta\,\mathrm{Im}_i(u_i)$ is the Mahalanobis distance between the aligned image patches. We will describe how to compute the Mahalanobis distance in section 7. In our experiments, we represent $L(u_i \mid D_i)$ by samples on a regular grid where the components of u_i range from $-10..10$.

4. Clustering of Flow by EM

Black and Jepson [4] and Ayer, *et al* [1] model multiple motions within a patch by a Mixture of Gaussians and find the flow of each motion component by Expectation Maximization (EM) [5]. A Mixture of Gaussians (or Gaussian Mixture Model, GMM) is a multi-modal probability distribution with density for a sample u

$$p(u \mid \theta) = \sum_{j=1}^{K} \pi_j \; N(u; \mu_j, C_j) \tag{4.1}$$

where $\theta = \{\theta_j = (\pi_j, \mu_j, C_j),\, j = 1..K\}$, π_j are the mixture probabilities, μ_j and C_j are the mean and covariance of the j^{th} Gaussian of the mixture and K is the number of layers. A sample u_i can be thought of as generated in a two step process. In the first step we choose a Gaussian j with probability π_j and then in the next step we generate x from the j^{th} Gaussian. Although Eq .(4.1) looks simple, it is impossible to have a closed form estimate of parameters θ using Maximum Likelihood Estimator. So we use EM.

EM is an iterative solution to the Maximum Likelihood Estimation problem and is useful when the observed data is incomplete, e.g. if we had the “complete” data, the problem would be easy to solve. The method also applies to problems that can be modeled as incomplete such as fitting a Mixture of Gaussians to a set of data, because if we were given the memberships of the data to clusters, it would become a problem with well known solution. In our clustering the incomplete data include the flow vectors as well.

Our approach differs from previous clustering approaches for motion segmentation in several respects. 1) We use a novel clustering algorithm that can explicitly handle uncertainty in the data and can model multi-modal distributions. 2) We apply the clustering to a small set of feature points and compute the affine parameters as a by-product of the clustering algorithm. 3) We perform dense segmentation as a separate step using a shift and align algorithm [8, 13] that incorporates a sophisticated correlated noise model [13].

Specifically, we obtain a small number (N) of textured image patches (eg. $N < 300$ for 320×240 images) by extracting $k \times k$ patches centered at randomly selected good features found using Eq. (3.1). For each image patch, we calculate its likelihood of observing a set of flow velocities u. The likelihood of the feature vectors is input to EM based clustering algorithm that is described in the next section.

5. Clustering of Incomplete Data

The well known EM Clustering algorithm takes as input a set of data and fits a GMM to the data. By convention it has two steps per iteration, the Expectation step (E-step) where we compute the membership probabilities for every datum using a guess for the mixture probabilities and the parameters of the Gaussians (mean and covariance) for each component in the GMM. The other step is the Maximization step (M-step) where we compute new values for the mixture probabilities and parameters for the Gaussians based on the computed membership probabilities. The expectation and maximization steps iterate until convergence.

The above EM Clustering algorithm as well as our EM Clustering algorithm is derived from the EM Algorithm [5] which also has two distinct steps: the Expectation and the Maximization. In the E-step, we calculate the conditional expectation

$$Q(\theta, \theta^t) = E_{D_z}\left\{ \ln p(D_y|\theta) \middle| D_x, \theta^t \right\} \tag{5.1}$$

where θ^t is a guess for θ (normally the result of the previous iteration), $D_x = \{D_i, i = 1 \cdots N\}$ is the observed data, $D_y = \{y_i, i = 1 \cdots N\}$ with $y_i = (D_i, \psi_i, u_i)$ is the complete data and $D_z = (z_i, i = 1, \ldots, N)$ with $z_i = (\psi_i, u_i)$ is the unobserved data. The cluster membership ψ_i is an array of length K such that $\psi_{ij} = 1$ if u_i was generated by the j^{th} cluster and zero otherwise. The only observed data are the images contained in D_x while the flow u_i and the cluster membership ψ_i are unobserved data. In the original EM clustering the observed data is the flow and the unobserved data is only the cluster membership.

In the M-step, we find the θ that maximizes $Q(\theta, \theta^t)$, which will become θ^{t+1} in the next iteration.

We start by deriving the $Q(\theta, \theta^t)$ for the E-step. The probability of the complete data given the clustering parameters is

$$p(D_y|\theta) = \prod_{i=1}^{N} p(y_i|\theta) = \prod_{i=1}^{N} p(\psi_i|\theta) p(u_i|\theta, \psi_i) p(D_i|\theta, \psi_i, u_i) =$$

$$\prod_{i=1}^{N} p(\psi_i|\theta) p(u_i|\theta, \psi_i) p(D_i|u_i) = \prod_{i=1}^{N} \left(\left(\prod_{j=1}^{K} {\pi_j}^{\psi_{ij}} \right) \left(\prod_{j=1}^{K} p(u_i|\theta_j)^{\psi_{ij}} \right) p(D_i|u_i) \right).$$

where π_j is the mixture probability for cluster j and θ_j is the parameter set for j^{th} cluster. We define $\bar{\psi}_{ij} = E\left\{ \psi_{ij} \middle| \theta^t, D \right\}$ and substitute back into Eq. (5.1) and after some mathematical manipulations [14]

$$Q(\theta, \theta^t) = \sum_{i=1}^{N} \left(\sum_{j=1}^{K} \bar{\psi}_{ij} \log \pi_j + \sum_{j=1}^{K} \bar{\psi}_{ij} \int \log p(u_i|\theta_j) p(u_i|D_i, {\theta_j}^t) du_i + E\left\{ \log p(D_i|u_i) \middle| \theta^t, D_x \right\} \right) \tag{5.2}$$

The third term in the equation does not contain θ so we do not consider it any more. We present the following lemma (for a proof see [14])

Lemma 1: If $p(u_i|\theta^t, D) = p(u_i|\theta^t, D_i)$ and $p(D_i|\theta^t, u_i) = p(D_i|u_i)$ then

$$\bar{\psi}_{ij} = \frac{\pi_j{}^t g_{ij}}{\sum_{k=1}^{K} \pi_k{}^t g_{ik}} \tag{5.3}$$

where

$$g_{ij} = p(D_i|\theta_j{}^t) = \int p(D_i|u_i)p(u_i|\theta_j{}^t)du_i \tag{5.4}$$

The M-step is just the following lemma (for a derivation see [14]):

Lemma 2: Q in Eq. (5.2) is maximized for

$$\pi_j = \frac{\sum_{i=1}^{N} \bar{\psi}_{ij}}{\sum_{k=1}^{K}\sum_{i=1}^{N} \bar{\psi}_{ik}}, \quad \mu_j = \frac{\sum_{i=1}^{N} \bar{\psi}_{ij} \frac{g'_{ij}}{g_{ij}}}{\sum_{i=1}^{N} \bar{\psi}_{ij}}, \quad C_j = \frac{\sum_{i=1}^{N} \bar{\psi}_{ij} \frac{g''_{ij}}{g_{ij}}}{\sum_{i=1}^{N} \bar{\psi}_{ij}}$$

where

$$g'_{ij} = \int p(D_i|u_i)p(u_i|\theta_j)u_i du_i \tag{5.5}$$

$$g''_{ij} = \int p(D_i|u_i)p(u_i|\theta_j)(u_i - \mu_j)(u_i - \mu_j)^T \tag{5.6}$$

This version of EM clustering becomes standard EM clustering if $p(D_i|u_i)$ in Eq. (5.4), (5.5), (5.6) is replaced by a delta function.

EM being an iterative technique, needs a guess. We use random initialization of means, covariances and mixture probabilities of the layers and we repeat the procedure a few times. At the end we select the run with the maximum likelihood.

We obtain the number of layers by using Bayesian Information Criterion (BIC) [6] which is defined as

$$BIC = 2l_M(x, \hat{\theta}) - m_M \log(n) \tag{5.7}$$

where $l_M(x, \hat{\theta})$ is the maximized mixture log likelihood for the model, m_M is the number of independent parameters to be estimated in the model and n is the number of data. In the BIC, a term is added to the log likelihood to penalize complex models. We determine the number of layers by selecting the layer that has the maximum BIC value out of the possible layers.

6. Affine Parameters

One can use a 6×1 vector to represent an affine flow for each feature point. The problem with this is that we would need a large patch centered on the feature point but at the same time incurring a higher risk of straddling a motion boundary. So we propose a simple technique that obtains the affine parameters as a by-product of the clustering process. We augment the flow vector u_i by the coordinates of the feature point x_i so it becomes a four dimensional vector and the clustering effectively gives us $P(u, x|\theta_j)$ instead of $P(u|\theta_j)$. Then we derive the affine flow u_j of each layer j by maximizing its conditional probability. Using the conditional probability formula [10],

$$P(u \mid x, \theta_j) = \frac{P(u, x \mid \theta_j)}{P(x \mid \theta_j)}$$

we maximize $P(u \mid x, \theta_j)$ and obtain

$$u_j = C_{j,\,ux}\, C_{j,\,xx}^{-1} x + (\mu_{j,u} - C_{j,\,ux}\, C_{j,\,xx}^{-1}\, \mu_{j,x}) \tag{6.1}$$

where μ_j the mean for cluster j and C_j the covariance for cluster j are

$$\mu_j = \begin{bmatrix} \mu_{j,u} \\ \mu_{j,x} \end{bmatrix} \qquad C_j = \begin{bmatrix} C_{j,\,uu} & C_{j,\,ux} \\ C_{j,\,ux}^T & C_{j,\,xx} \end{bmatrix}$$

Eq. (6.1) is an affine expression and although tedious to derive, it is an inexpensive computation since all the matrices are 2×2.

7. Dense Segmentation

We follow the shift and align paradigm for motion segmentation. For each layer we warp the one image with the affine flow for this layer toward the other image and subtract them. This difference is small for pixels that move consistently with the flow for this layer. To check this we examine small patches (5×5 in our experiments) and apply a χ^2 test to determine the spatial support of each layer.

7.1. Correlated Noise Model

In practice, the noise between neighboring pixels is often correlated. One situation where the independence assumption is inadequate is when the lighting conditions change from frame to frame. Then the noise in neighboring pixels becomes correlated assuming the pixels within a patch are under the same lighting conditions. Another situation where noise is correlated is when the tracking is uniformly inaccurate such that pixels within the same patch drift by a similar amount. This is particularly important when one approximates flow by fitting a simple model like affine over a large region. Such a model is accurate for some patches of the region but introduces a drift in other patches. In what follows, we describe a model that addresses both situations.

We identify pixels that follow the same affine motion model u_j as the jth layer by evaluating a statistic on $\Delta\,\mathrm{Im}$ the image difference between Im_2 and ${}^{u_j}\,\mathrm{Im}_1$ which is Im_1 warped by the affine flow u_j by examining small $k \times k$ patches. For each pixel $[i, j]$ within a patch,

$$\Delta\,\mathrm{Im}[i, j] = {}^{u_j}\,\mathrm{Im}_1[i, j] - \mathrm{Im}_2[i, j] \tag{7.1}$$

$$= \Delta n[i, j] + {}^{u_j}\,\mathrm{Im}_{1,\,x}[i, j]\;\Delta u_\alpha[i, j] + {}^{u_j}\,\mathrm{Im}_{1,\,y}[i, j]\;\Delta v_\alpha[i, j]$$

$$+ {}^{u_j}\,\mathrm{Im}_{1,\,x}[i, j]\;\Delta u + {}^{u_j}\,\mathrm{Im}_{1,\,y}[i, j]\;\Delta v + {}^{u_j}\,\mathrm{Im}_1[i, j]\;\Delta l + \Delta f$$

where $\Delta n[i, j]$ is a pixel-wise white noise that is attributed to image acquisition noise in the camera. Two components of pixel-wise motion noise $\Delta u_\alpha[i, j]$, $\Delta v_\alpha[i, j]$ that come from either independent motion of image details within an image patch (for example, the motion of individual moving leaves of a tree) or more commonly from aliasing in images. The aliasing arises as a consequence of resampling of the images or warping and we found empirically to be proportional to the derivatives of images. All three random variables $\Delta n[i, j]$, $\Delta u_\alpha[i, j]$ and $\Delta v_\alpha[i, j]$ are independent and identically distributed (i.i.d.) or white noise. ${}^{u_j}\,\mathrm{Im}_{1,\,x}[i, j]\;\Delta u_\alpha[i, j]$ is not identically distributed noise since its magnitude depends on Im_1 but it is independent. Next we model four components of dependent noise. Two components of patch-wise motion noise Δu, Δv that arises from the error in the tracking within an image patch. Also, there are two patch-wise noise components Δl, Δf that come from the change of illumination that we model as affine [9]. Of the seven random variables, Δn, Δu_α, Δv_α are arrays with k^2 elements, the rest Δu, Δv, Δl, Δf being scalar.

7.2. Mahalanobis Distance

Eq. (7.1) defines a noise model for any image patch that follow the same affine flow model as the jth layer. So we perform a χ^2 test for every patch in the image to decide which pixels belong to the jth layer. We identify pixels whose motion follows the affine flow u_j for the jth layer by computing their Mahalanobis distance which is defined as

$$D_j^2 = ({}^{u_j}\vec{\mathrm{Im}}_1 - \vec{\mathrm{Im}}_2)^T \, \mathbf{C}_{\Delta \vec{\mathrm{Im}}}{}^{-1} \, ({}^{u_j}\vec{\mathrm{Im}}_1 - \vec{\mathrm{Im}}_2) \tag{7.2}$$

where $\mathbf{C}_{\Delta \vec{\mathrm{Im}}}$ is the covariance matrix of $\Delta \vec{\mathrm{Im}}$.

$$\mathbf{C}_{\Delta \vec{\mathrm{Im}}} = \mathbf{C_n} + \mathbf{U} \; \mathbf{C_u} \; \mathbf{U}^T$$

$\mathbf{C_n}$ is a $k^2 \times k^2$ diagonal matrix whose elements are vector $\vec{1}\sigma_n^2 + {}^{u_j}\vec{\mathrm{Im}}_{1,\,d}\,\sigma_a^2$ where $\vec{1}$ is a vector of 1s, ${}^{u_j}\vec{\mathrm{Im}}_{1,\,d}$ is a vector from a patch of the sum of squares of the x and y derivatives of ${}^{u_j}\vec{\mathrm{Im}}_1$ and σ_n^2, σ_a^2 are scalar constants representing the variances of the camera and aliasing noise (Δn, Δu_α, Δv_α). $\mathbf{C_u}$ is a 4×4 diagonal matrix for patch-wise motion noise whose diagonal elements denote the variance of random variables Δu, Δv, Δl, Δf and $\mathbf{U}$ is a $k^2 \times 4$ matrix whose 1st, 2nd and 3rd columns are the elements of the $k \times k$ patch extracted from ${}^{u_j}\mathrm{Im}_{1,\,x}$, ${}^{u_j}\mathrm{Im}_{1,\,y}$ and ${}^{u_j}\mathrm{Im}_1$ respectively. The 4th column is a vector of all 1s.

A daunting problem associated with the Mahalanobis distance is efficiency. Since Mahalanobis requires the inversion of the covariance matrix and for a $k \times k$ patch the covariance is $k^2 \times k^2$, the cost of the inversion is $O(k^6)$. This is time-consuming even for a small patch size because we need to do that for every pixel in the image. In [13], we show how to speed up the computation by using the Sherman-Morrison-Woodbury identity [7] and amortize the cost of computation over the whole image thus reducing the cost from $O(k^6)$ to constant time.

The Mahalanobis distance between two $k \times k$ image patches follows a Chi-Square distribution [10] with k^2 degrees of freedom for appropriately estimated variances of the random variables. In [13], we show how to do that using a Maximum Likelihood Estimator. We can segment out pixels that belong to the jth layer by thresholding D_j^2. The value of threshold t_{χ^2} can be taken from a χ^2 table [10]. For patch size $k = 5 \times 5$, the degrees of freedom are 25, $t_{\chi^2} = 46.9$ for confidence level 0.995.

8. Experiments

In this section, we present the results of running our algorithm on real image sequences. In the first experiment, we run the algorithm on two frames from the Flower Garden Sequence [3]. The algorithm automatically selects the number of layer that yields the maximum BIC value for dense segmentation. For the flower garden sequence, the number of layers is 3.

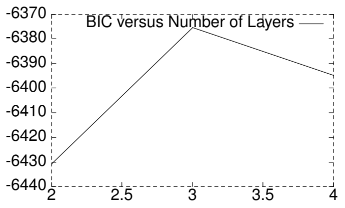

Figure 8.1: A graph of BIC against number of layers.

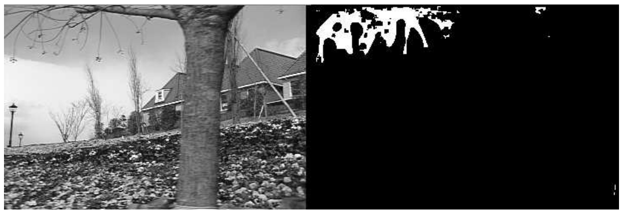

Figure 8.2: Frame 1, segment corresponding to tree branches

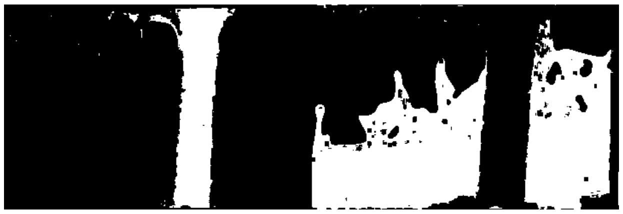

Figure 8.3: Segments corresponding to the tree and background

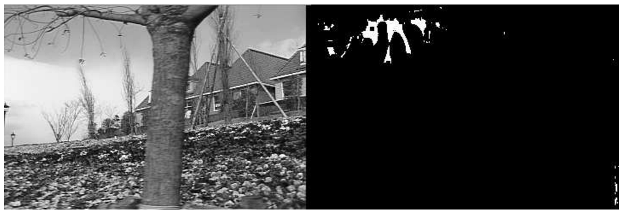

Figure 8.4: Frame 8, segment corresponding to tree branches

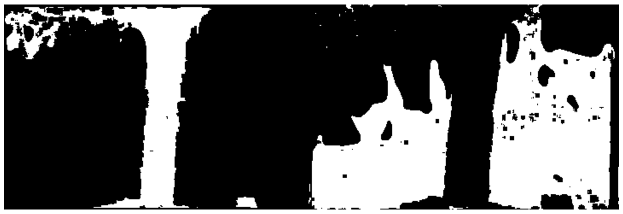

Figure 8.4: Segments corresponding to the tree and background

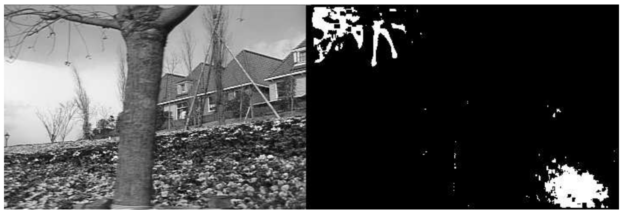

Figure 8.6: Frame 14, segment corresponding to branches

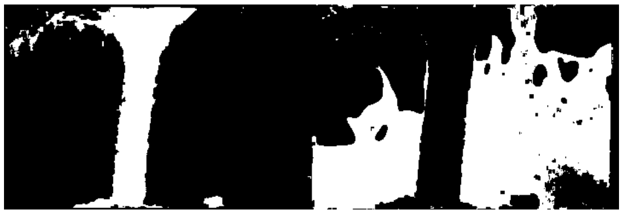

Figure 8.7: Segments corresponding to the tree and background

In the second experiment, we run the algorithm on an image sequence which shows a moving toy truck taken by a hand-held camera trying to follow the motion of the truck. Both the foreground and background are moving.

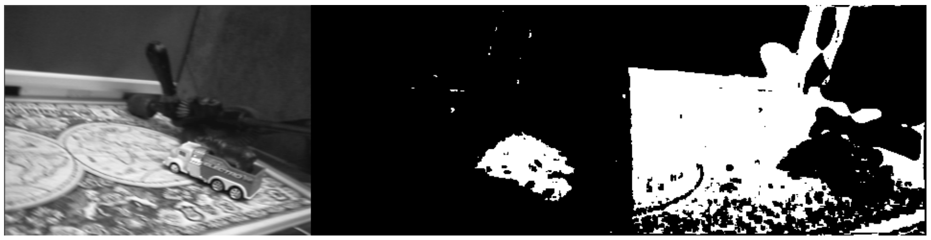

Figure 8.8: Frame 1, segments corresponding to truck and moving background

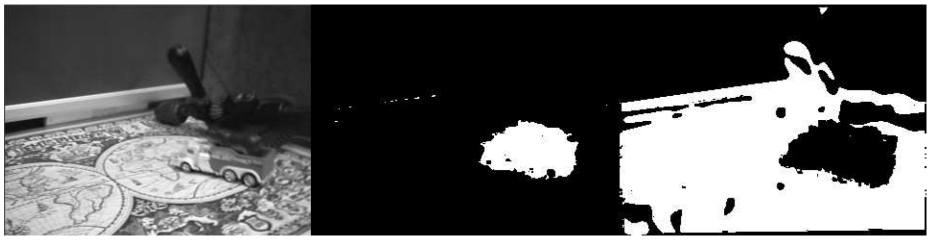

Figure 8.9: Frame 8, segments corresponding to truck and moving background

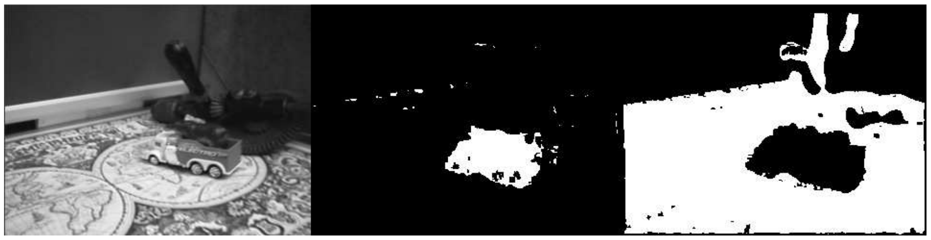

Figure 8.10: Frame 16, segments corresponding to truck and moving background

9. Conclusion

In this paper, we present a novel motion segmentation algorithm. The algorithm clusters the likelihood of observing a set of flow velocities at textured image patches using a specialized version of EM that treats the image flow velocities as incomplete data. We obtain the affine parametric flow models of each layer as a direct by-product of EM clustering. Our residual noise model allows for correlation between neighboring pixels in contrast to previous approaches which assumes an independent noise model. We identify the pixels belonging to a layer by thresholding their Mahalanobis distance and for an image patch of size $k \times k$, we reduce the cost of inversion of a big $k^2 \times k^2$

covariance matrix from $O(k^6)$ to constant time by using Sherman-Morrison-Woodbury identity and amortizing the computational cost over the whole image. Our experiments with real image sequences show the algorithm produces good layers from two frames.

References

[1] S. Ayer and H. Sawhney, "Layered Representation of Motion Video using Robust Maximum-Likelihood Estimation of Mixture Models and MDL Encoding," pp. 777-784 in *Proceedings of Fifth International Conference on Computer Vision*, (1995).

[2] J. L. Barron, D. J. Fleet, and S. S. Beauchernin, "Performance of Optical Flow Techniques," pp. 43-77 in *International Journal of Computer Vision*, (1994).

[3] M. Black, "Michael J Black: Image Sequences," in *http://www.cs.brown.edu/people/black/images.html*, (2002).

[4] M. Black and A. Jepson, "Mixture models for optical flow computation," pp. 760-761 in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, (1993).

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," pp. 1-38 in *Journal of the Royal Statistical Society Series B*, (1977).

[6] C. Fraley and A. E. Raftery, "How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis," in *Technical Report No. 329, Department of Statistics, University of Washington*, (1998).

[7] W. W. Hager, "Updating the Inverse of a Matrix," *SIAM Review* **31**(2) pp. 221-239 (1989).

[8] M. Irani, B. Rousso, and S. Peleg, "Computing Occluding and Transparent Motions," *International Journal of Computer Vision* **12**(1) pp. 5-16 (1994).

[9] S. Negahdaripour and C. Yu, "A Generalized Brightness Change Model for Computing Optical Flow," pp. 2-11 in *Proceedings of Fourth International Conference on Computer Vision*, (1993).

[10] H. Stark and J. W. Woods, *Probability and Random Processes with Applications to Signal Processing,* Prentice Hall (2002).

[11] C. Tomasi and T. Kanade, "Detection and Tracking of Point Features," in *Technical Report CMU-CS-91-132, Carnegie Mellon University*, (1991).

[12] J. Y. A. Wang and E. H. Adelson, "Representing moving images with layers," pp. 625-638 in *IEEE Transactions on Image Processing*, (1994).

[13] K. Y. Wong and M. E. Spetsakis, "Tracking Based Motion Segmentation under Relaxed Statistical Assumptions," in *Department of Computer Science, York University, Technical Report CS-2003-09*, (2003).

[14] L. Ye and M. E. Spetsakis, "Clustering on Unobserved Data using Mixture of Gaussians," in *Department of Computer Science, York University, Technical Report CS-2003-08*, (2003).