Optical Flow Estimation & Segmentation of Multiple Moving Dynamic Textures

René Vidal and Avinash Ravichandran
Center for Imaging Science, Johns Hopkins University
301 Clark Hall, 3400 N Charles St., Baltimore, MD, 21218, USA

Abstract

We consider the problem of modeling a scene containing multiple dynamic textures undergoing multiple rigid-body motions, e.g., a video sequence of water taken by a rigidly moving camera. We propose to model each moving dynamic texture with a time varying linear dynamical system (LDS) plus a 2-D translational motion model. We first consider a scene with a single moving dynamic texture and show how to simultaneously learn the parameters of the time varying LDS as well as the optical flow of the scene using the so-called dynamic texture constancy constraint (DTCC). We then consider a scene with multiple non-moving dynamic textures and show that learning the parameters of each time invariant LDS as well as their region of support is equivalent to clustering data living in multiple subspaces. We solve this problem with a combination of PCA and GPCA. Finally, we consider a scene with multiple moving dynamic textures, and show how to simultaneously learn the parameters of multiple time varying LDS and multiple 2-D translational models, by clustering data living in multiple dynamically evolving subspaces. We test our approach on sequences of flowers, water, grass, and a beating heart.

1. Introduction

The past few decades have witnessed significant advances in motion analysis under the assumption that the scene is *Lambertian*, *rigid* and *static*. The Lambertian assumption is crucial to the problems of tracking, optical flow and correspondences, because the intensity of a point is independent of the view point. Given optical flow or point correspondences, the assumption of a rigidly moving camera observing a static world enables us to both recover the camera motion as well as reconstruct the rigid shape of the scene.

Recently there have been attempts to relax each one of these assumptions separately. For example, the generalized constant brightness constraint [8] allows us to compute the optical flow in the case of non Lambertian scenes. Likewise, the multibody fundamental matrix [14] allows us to reconstruct dynamic scenes consisting of multiple rigid motions.

As for the assumption of rigidity, there have been three main approaches to dealing with nonrigid scenes. In feature-based methods such as [2, 1, 13, 6], a rigidly moving camera observes a single nonrigid shape. By modeling the nonrigid shape as a linear combination of rigid shapes, once can jointly recover rigid motion and nonrigid shape from point correspondences in multiple views by extending the classical algorithm for static shapes [12]. In direct approaches [10, 3, 16], a static camera observes a nonrigid scene whose temporal evolution exhibits certain stationarity, e.g., water, foliage, steam, etc. These scenes are called *dynamic textures*. By modeling the temporal evolution of the image intensities as the output of a time invariant linear dynamical system (LDS), one can jointly recover a model for the appearance and dynamics of the scene using classical system identification techniques. Other techniques, such as [9], do not use a model to describe the dynamics of the scene. Instead, the synthesis is done by choosing a frame in the video that is "close" to the current one.

To the best of our knowledge, there have been only two attempts at dealing with both nonrigid and dynamic scenes. The work of [5] introduces the concept of stochastic rigidity for dealing with scenes in which a rigidly moving camera observes a single dynamic texture. Unfortunately, the paper provides no computational framework for either estimating the optical flow or learning the appearance and dynamics of the scene. The work of [4] deals with scenes in which a static camera observes multiple dynamic textures. It proposes a level set-based technique to segment the scene into multiple regions, each one represented with a LDS. As with most level set techniques, the method is very computationally intense. Furthermore, it can only deal with scenes in which the dynamic textures do not move.

1.1. Paper contributions

In this paper we look at the problem of modeling and segmenting a scene consisting of *multiple dynamic textures undergoing multiple rigid motions*. More specifically, we consider the following three problems.

1. **Estimating the optical flow of a dynamic texture.** First, we consider the problem of jointly recovering the optical flow and a dynamical model for a single dynamic texture being observed by a rigidly moving camera. We are faced with two key challenges:
   (a) One cannot estimate a LDS for the dynamic texture as in [10], because the camera is moving, so the appearance model is no longer time invariant.
   (b) One cannot estimate the optical flow using the brightness constancy constraint, because the intensity of a pixel changes with the viewpoint due to the temporal evolution of the dynamic texture.

We solve the first problem by modeling moving dynamic textures using a time varying LDS together with a 2-D translational motion model. These models together capture both the temporal evolution of the image intensities as well as the camera motion and lead to a generalization of the brightness constancy constraint to nonrigid scenes, the so-called dynamic texture constancy constraint (DTCC). The DTCC allows us to jointly estimate the optical flow of the scene as well as a LDS for the dynamic texture.

2. **Segmenting non-moving dynamic textures.** Second, we consider the problem of modeling a static camera observing multiple non-moving dynamic textures. We demonstrate that the image intensities generated by a single dynamic texture live in a low-dimensional subspace of a high-dimensional linear space. Therefore, the problem of segmenting multiple dynamic textures can be transformed into one of segmenting multiple subspaces. We solve the latter problem in closed form using an algebraic technique called Generalized Principal Component Analysis (GPCA) [15].

3. **Segmenting moving dynamic textures.** Third, we look at the problem of modeling scenes with multiple moving dynamic textures. By combining step 1, which allows us to estimate the LDS and the 2-D translational model for a single moving dynamic texture, and step 2, which allows us to segment multiple non-moving dynamic textures, we show that one can simultaneously learn the LDS, the 2-D translational motion model, and the region of support of each one of the moving dynamic textures.

As we will see shortly, our approach provides an extremely simple and computationally straightforward solution to the very challenging problem of simultaneous motion estimation and segmentation of rigid and nonrigid motions.

2. Optical Flow of a Dynamic Texture

2.1. Modeling non-moving dynamic textures

Let $I(x, y)$ be an image of a static texture $\mathcal{T}$. A simple model for the appearance of $\mathcal{T}$ is to assume that I can be written as a linear combination of certain basis, i.e.

$$I(x, y) = \sum_{i=1}^{n} C_i(x, y) z_i. \tag{1}$$

In vector notation we write $I = Cz$, where $I \in \mathbb{R}^P$, $C \in \mathbb{R}^{P \times n}$ and $z \in \mathbb{R}^n$, with P the number of pixels and n the number of texture bases. By an abuse of notation, we denote the row of C associated with pixel (x, y) as $C(x, y) \in \mathbb{R}^{1 \times n}$, so that $I(x, y) = C(x, y)z$.

Let us now consider a video sequence $I(x, y, t)$ of a non-moving dynamic texture $\mathcal{T}$. As proposed in [10], we can model the appearance of $\mathcal{T}$ as a linear combination of certain basis, except that the coefficients z are now a function of time, i.e. the intensity value of pixel (x, y) at frame t is given by $I(x, y, t) = C(x, y)z(t)$. In vector notation we write $I(t) = Cz(t)$. The temporal evolution of the vector of coefficients can be described in a variety of ways. A simple model is to assume that z evolves according to a first-order linear dynamical system (LDS) driven by white noise $\eta(t) \in \mathbb{R}^{n_\eta}$, i.e. $z(t + 1) = Az(t) + B\eta(t)$, for some $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times n_\eta}$. The number of texture basis n is assumed to be the same as the order of the LDS in this model. We thus obtain the following ARMA model for a dynamic texture $\mathcal{T}$

$$\begin{aligned} z(t + 1) &= Az(t) + B\eta(t) \\ I(t) &= Cz(t) + w(t), \end{aligned} \tag{2}$$

where $w(t)$ is measurement white noise.

Given a video sequence $\{I(t)\}_{t=1}^{F}$ of a dynamic texture, one can learn the parameters of the ARMA model (A, B, C) optimally and in closed form using subspace identification techniques such as N4SID [7]. Since

$$W = [I(1) \cdots I(F)] = C[z(1) \cdots z(F)] = CZ, \tag{3}$$

when $\eta = w = 0$, a sub-optimal solution is to learn C and Z from the singular value decomposition of $[I(1) \cdots I(F)]$. Given Z, solving for A is a linear problem. Given A and Z, solving for B (assuming now that $\eta \neq 0$) is also a linear problem. We refer the reader to [10] for details about this sub-optimal solution. Once the parameters (A, B, C) have been learned, one can use the identified LDS for synthesis, classification and recognition of dynamic textures.

2.2. Modeling moving dynamic textures

Consider now a video sequence $I(x, y, t)$ of a dynamic texture $\mathcal{T}$ taken by a moving camera. In this case the temporal evolution of $I(x, y, t)$ is determined not only by the nonrigid motion of the dynamic texture, but also by the rigid motion of the camera.

In order to understand whether the model in (2) can capture both rigid and nonrigid motions, let us have a closer look at the model. First, notice that A, B and $z(t)$ do not depend on the pixel coordinates. Therefore, they can only capture the temporal evolution of the texture, i.e. the nonrigid motion. The matrix C, on the other hand, depends explicitly on the pixel coordinates and, by construction, each image $I(t)$ is a linear combination of the columns of C with time-varying coefficients $z(t)$. Therefore, we can interpret the C matrix as modeling the static texture or appearance of the dynamic texture. In fact, if the scene was static and rigid and $n = 1$, then we would have $A = 1$, $B = 0$, $z(t) = 1$, and hence the matrix C would coincide with the image $I(t)$. Since the matrix C does not change with time, the image would be forced to be constant. Therefore, the time invariant LDS (2) can capture the nonrigid motion the dynamic texture, but cannot capture the motion of the camera.

In order to capture both the rigid and nonrigid motions present in the scene, we propose to modify the LDS (2) by making the C matrix time dependent. That is, we propose to model a moving dynamic texture with a time varying LDS

$$\begin{aligned} z(t+1) &= Az(t) + B\eta(t) \\ I(t) &= C(t)z(t) + w(t). \end{aligned} \tag{4}$$

Unfortunately, the optimal identification of the model parameters of a stochastic time varying LDS is not a straightforward problem. Even in the case of time invariant LDS, the identification problem was solved in closed form only in the last decade [7]. Therefore, similar to the work of [10], in this paper we seek a sub-optimal solution under certain assumptions on the temporal evolution of $C(t)$.

As we will see in the next subsection, the rigid motion of the camera already imposes some constraints on the temporal evolution of $C(t)$. Such constraints will be used for estimating the optical flow given $C(t)$. In order to estimate $C(t)$ directly from image data, we assume that the camera motion is small so that $C(t)$ is slowly varying. More specifically, we assume that in each time window of size $\tau > n$ $C(t)$ is approximately constant and learn a time invariant LDS for that time window using the method in [10]. By applying this to each time window $[t-\tau+1, t]$ we obtain $C(t)$ for $t \in \{\tau, \tau+1, \cdots, F\}$ and $z(t)$ for $t \in \{1, 2, \cdots, F\}$. Since the parameters of a time invariant LDS can only be identified up to a change of basis $L \in \mathbb{R}^{n\times n}$, i.e. the models (A, B, C) and (LAL^{-1}, LB, CL^{-1}) are equivalent, the last step is to ensure that, as t varies, $C(t)$ and $z(t)$ are computed with respect to the same basis. We do so by using the basis at time $t = \tau$ as the reference. That is, given (A_t, C_t, B_t), we compute a change of basis L_t such that $A_\tau = L_t A_t L_t^{-1}$, and obtain our time varying LDS as $(A_\tau, L_t B_t, C_t L_t^{-1})$.

2.3. Estimating optical flow from the dynamic texture constancy constraint (DTCC)

In the proposed model of a moving dynamic texture (2) the time invariant matrix A plays the same role as in a non-moving texture, i.e. it captures the nonrigid motion of the scene. On the other hand, the time varying matrix $C(x, y, t)$ captures both the static texture or appearance of the scene, via its dependency on (x, y), as well as the camera motion, via its dependency on t. Nevertheless, notice that in order for the temporal evolution of $C(t)$ to capture only the camera motion, we cannot allow $C(t)$ to be arbitrary. Otherwise, a model with parameters $n = 1$, $A = 1$, $B = 0$, and $C(t) = I(t)$ would fit the image data perfectly, yet the resulting model would not be meaningful in the sense of capturing both the rigid and nonrigid dynamics of the scene.

In order to constraint the values that C can take on, we propose to use an equivalent of the brightness constancy constraint (BCC) for dynamic textures. To this end, recall that for a static Lambertian scene the image brightness $I(x, y, t)$ does not depend on the view point. Thus we have

$$I(x+u, y+v, t+1) = I(x, y, t), \tag{5}$$

where (u, v) is the image displacement or optical flow at pixel (x, y). The BCC (5) can also be expressed in differential form as a linear relationship between the spatio-temporal image derivatives and the optical flow as

$$\frac{\partial I(x,y,t)}{\partial x}u + \frac{\partial I(x,y,t)}{\partial y}v + \frac{\partial I(x,y,t)}{\partial t} = 0. \tag{6}$$

Since the BCC gives one equation in two unknowns (u, v), the simplest method for computing the optical flow is to integrate the image measurements in a neighborhood Ω of (x, y) and solve for the optical flow from

$$\sum_{\Omega} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -\sum_{\Omega} \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}. \tag{7}$$

Unfortunately, the BCC is not applicable to video sequences containing dynamic textures, because the brightness of a pixel does change as a function of time without any change of the view point. However, as we have alluded to earlier, the matrix $C(x, y, t)$ for a dynamic texture plays the analogous role of $I(x, y, t)$ for a static scene. By assuming that C does not depend on the view point, we obtain the following *dynamic texture constancy constraint* (DTCC)

$$C(x+u, y+v, t+1) = C(x, y, t) \tag{8}$$

or its differential form

$$\frac{\partial C(x,y,t)}{\partial x}u + \frac{\partial C(x,y,t)}{\partial y}v + \frac{\partial C(x,y,t)}{\partial t} = 0. \tag{9}$$

Unlike the BCC, which is a scalar equation, the DTCC is a vector equation because $C(x, y, t) \in \mathbb{R}^{1\times n}$, where n is the order of the LDS. Therefore, if $C(x, y, t)$ was known, we could immediate recover the optical flow at each pixel, without integrating C in a neighborhood of (x, y), from

$$\begin{bmatrix} C_x C_x^T & C_x C_y^T \\ C_x C_y^T & C_y C_y^T \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -\begin{bmatrix} C_x C_t^T \\ C_y C_t^T \end{bmatrix}. \tag{10}$$

The main problem with this approach to computing optical flow from dynamic textures is that it relies heavily on a good knowledge of the C matrix, which may not be feasible to obtain in practice. Therefore, it would be nice if we could express the DTCC directly in terms of our measurements $I(x, y, t)$. This can be easily achieved by multiplying the vector equation (9) by $z(t)$. We obtain (assuming $w = 0$)

$$\begin{aligned} &\frac{\partial C(x,y,t)}{\partial x}z(t)u + \frac{\partial C(x,y,t)}{\partial y}z(t)v + \frac{\partial C(x,y,t)}{\partial t}z(t) \\ &= \frac{\partial I(x,y,t)}{\partial x}u + \frac{\partial I(x,y,t)}{\partial y}v + \frac{\partial C(x,y,t)}{\partial t}z(t) = 0, \end{aligned}$$

which except for the last term, depends directly on the image data. Indeed, note that this version of the DTCC is essentially the same as the BCC, except that the last term $\frac{\partial I}{\partial t}$ is replaced by $\frac{\partial C}{\partial t}z(t)$. Therefore, one may solve for the optical flow using (7), or any of the existing approaches for optical flow estimation that are based on the BCC.

There is a very intuitive reason for replacing $\frac{\partial I}{\partial t}$ by $\frac{\partial C}{\partial t}z(t)$ when going from static scenes to dynamic textures. If there is no measurement noise in (4), i.e. $w = 0$, then

$$\frac{\partial I(x,y,t)}{\partial t} = C(x,y,t)\frac{\partial z(t)}{\partial t} + \frac{\partial C(x,y,t)}{\partial t}z(t). \quad (11)$$

Note that $\frac{\partial I}{\partial t}$ is the sum of two terms: the first one due to nonrigid motion and the other one due to rigid motion. Thus in order to estimate the optical flow due to rigid motion, we must only consider the rigid component $\frac{\partial C}{\partial t}z(t)$ of $\frac{\partial I}{\partial t}$. Because $C(t)$ may not be estimated accurately, we avoid computing $\frac{\partial C}{\partial t}$ by using (11) to approximate $\frac{\partial C}{\partial t}z(t)$ as $\frac{\partial I}{\partial t} - I(t) + C(t)z_{t-1}$.

3. Segmenting Non-moving Dynamic Textures using Generalized PCA

Consider now a scene consisting of K non-moving dynamic textures $\{\mathcal{T}_k\}_{k=1}^K$. Without knowing which pixels correspond to which dynamic texture, we would like to simultaneously learn the parameters $\{(A_k, B_k, C_k)\}_{k=1}^K$ of each texture as well as the spatial segmentation of the scene.

If there is only one dynamic texture in the scene, then from (3) we know that when $w = 0$ and $F, P \geq n$, where n is the order of the dynamic texture, then the $P \times F$ measurement matrix $W = [I(1) \cdots I(F)]$ is of rank at most n. Therefore, when $w \approx 0$, each row of W lives approximately in a subspace of $\mathbb{R}^F$ of dimension at most n and so if there are K dynamic textures in the scene, then each row of W lives in one out of K possible subspaces of $\mathbb{R}^F$. The problem of segmenting multiple non-moving dynamic textures is then equivalent to one of clustering the rows of W into K subspaces.

Segmenting data living in multiple subspaces has been an active topic of research over the past few years. Extant methods randomly choose a basis for each subspace to then iterate between data segmentation and subspace estimation. This can be done using, e.g., K-subspace, an extension of K-means to the case of subspaces, or Expectation Maximization (EM) for mixtures of Principal Component Analysis (PCA) [11].

An alternative algebraic approach, which does not require any initialization, is Generalized PCA (GPCA) [15]. In this approach, the data points are first projected onto a low-dimensional subspace to reduce dimensionality. Then, a polynomial is fit to the projected data points and a basis for each one of the projected subspaces is obtained from the derivatives of this polynomials at the data points.

For the problem of segmenting non-moving dynamic textures, the GPCA algorithm operates as follows:

1. *Project the rows of $W \in \mathbb{R}^{P\times F}$ onto a linear subspace of dimension $d = n + 1$.* This can be done by first computing the SVD of $W = USV^T$, and then letting the projected $W \in \mathbb{R}^{P\times d}$ be the first d columns of U.

2. *Fit a polynomial to the projected data.* The rationale behind this step is as follows. Let $\boldsymbol{w} \in \mathbb{R}^d$ be any of the rows of $W \in \mathbb{R}^{P\times d}$. Since $\boldsymbol{w}$ must belong to one of the K projected subspaces, say S_k, then there exists a vector $\boldsymbol{b}_k \in \mathbb{R}^d$ normal to S_k such that $\boldsymbol{b}_k^T\boldsymbol{w} = 0$. Then any row $\boldsymbol{w}$ of W must satisfy the following homogeneous polynomial of degree K in d variables

$$p_n(\boldsymbol{w}) = (\boldsymbol{b}_1^T\boldsymbol{w})(\boldsymbol{b}_2^T\boldsymbol{w})\cdots(\boldsymbol{b}_K^T\boldsymbol{w}) = 0. \quad (12)$$

   This polynomial can be expressed linearly in terms of its coefficients. For instance, if $n = 2$ and $d = 2$ we have $p_n(\boldsymbol{w}) = c_1w_1^2 + c_2w_1w_2 + c_3w_2^2$, and we can solve for the coefficients from the linear system

$$[c_1, c_2, c_3]\begin{bmatrix} w_{11}^2 & \cdots & w_{1P}^2 \\ w_{11}w_{21} & \cdots & w_{1P}w_{2P} \\ w_{21}^2 & \cdots & w_{2P}^2 \end{bmatrix} = 0. \quad (13)$$

3. *Solve for the normal vectors from the derivatives of p_n.* The polynomial $p_n = 0$ describes a surface in $\mathbb{R}^d$, i.e. the union of the K projected subspaces. Therefore, the derivative of p_n at a data point gives a vector normal to the subspace passing through that point, i.e.

$$\boldsymbol{b}_k = \frac{Dp_n(\boldsymbol{w})}{\|Dp_n(\boldsymbol{w})\|}. \quad (14)$$

4. *Segment the data.* Assign every point $\boldsymbol{w}$ to the closest subspace k if

$$k = \arg\min_{\ell=1,\ldots,K}\{(\boldsymbol{b}_\ell^T\boldsymbol{w})^2\}. \quad (15)$$

5. *Learn the models.* Use system identification to compute (A_k, B_k, C_k) for each texture $k = 1, \ldots, K$.

4. Optical Flow Estimation & Segmentation of Moving Dynamic Textures

Consider now a scene consisting of K moving dynamic textures $\{\mathcal{T}_k\}_{k=1}^K$. Without knowing at each frame which pixels belong to which dynamic texture, we would like to simultaneously learn the dynamical model $(A_k, C_k(t), B_k)$ associated with each texture, its optical flow $u_k(x, y, t)$ and $v_k(x, y, t)$, and the spatio-temporal segmentation of the scene.

In this case, the matrix $C(t)$ varies as a function of time, and so the rows of the matrix $W = [I(1) \cdots I(F)]$ no longer live in a collection of K subspaces of dimension n. However, if we assume that in each time window of size τ each one of the matrices $C_k(t)$ is slowly varying, then the intensities generated by the K dynamic textures live approximately in a union of K subspaces that are slowly moving and changing orientation as a function of time. At each frame, the problem of segmenting the dynamic textures is equivalent to segmenting data in multiple subspaces. The problem of optical flow is, on the other hand, equivalent estimating how these subspace evolve as a function of time.

In order to tackle this problem of simultaneous clustering and tracking multiple subspaces, we propose to combine dynamic texture segmentation via GPCA with dynamic texture optical flow estimation via the DTCC as follows:

1. Compute the spatio-temporal image derivatives for all F frames in $I(t)$.
2. For each time window $[t-\tau+1, t]$ compute $C(t)$ and $z(t)$ from the singular value decomposition of $W(t) = [I(t-\tau+1)\cdots I(t)]$, as described in Section 2.
3. For each time window $[t-\gamma+1, t]$ compute a spatial segmentation of $I(t)$ into K groups by applying the dynamic texture segmentation algorithm described in Section 3 to the matrix $W(t) = [I(t-\gamma+1)\cdots I(t)]$.
4. For each group $k = 1, \cdots, K$, compute the optical flow (u_k, v_k) at time t from $I_x(t)$, $I_y(t)$, $C_k(t)$ and $z_k(t)$, as described in Section 2.

5. Experimental Results

Estimating optical flow for one moving dynamic texture. To test the proposed algorithm for modeling and optical flow estimation of a single moving dynamic texture, we used a 120×160 video sequence of a flower bed taken by a panning camera as shown in Figure 1. The camera is initially stationary, then it moves to the right, and then it stops.

We learned a time varying LDS for the sequence by identifying a time invariant LDS of order $n = 20$ using a moving window of size $\tau = 22$ frames, as described in Section 2. In learning this model, we did not enforce that the matrix A be time invariant. In order to validate the assumption of A being time invariant, in Figure 2 we plot the eigenvalues of the A matrices for different time windows. Notice that indeed the eigenvalues of the A matrices are very similar for different time windows.

Given the estimated $C(t)$ and $z(t)$, we calculated the optical flow as described in Section 2. Since in this case the optical flow of all the pixels is the same, we took the region of integration Ω for the optical flow calculation in (7) to be the entire image, i.e. we computed only one value for the optical flow at each frame. Figure 3 plots the estimated optical flow as a function of time. Note that the optical flow in the vertical direction is estimated to be approximately zero, as expected. The optical flow in the horizontal direction is approximately zero for the first 50 frames, negative for the next 150 frames, and approximately zero for the last 40 frames. By integrating the horizontal optical flow, we estimated the overall displacement of the sequence as 60 pixels. The ground truth displacement was computed from the motion of one of the red flowers as 85 pixels. Therefore, our algorithm underestimates the amount of rigid motion for this sequence. This is because for $t \in [30, 50]$ the flowers oscillate more frequently when compared to the first 30 frames of the video sequence. This causes our algorithm to estimate a positive optical flow for a few frames.

Figure 1. A sequence of moving flowers taken by a moving camera. See `http://www.robots.ox.ac.uk/~awf/iccv01/`.

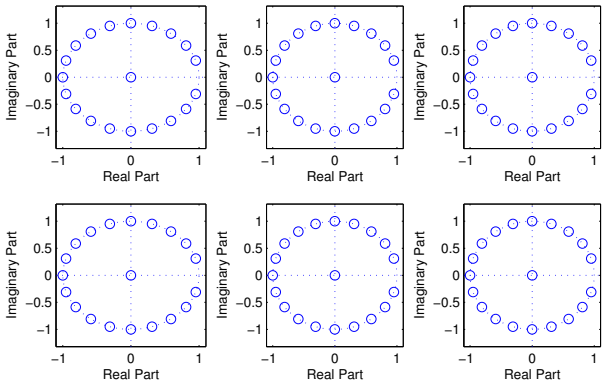

Figure 2. Eigenvalues of the A matrices for the flower sequence.

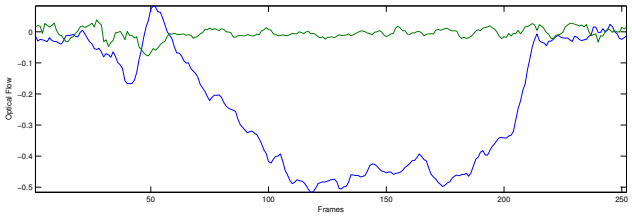

Figure 3. Optical flow estimation for the flower sequence.

Segmentation of non-moving dynamic textures. The proposed algorithm for segmentation of non-moving dynamic textures was applied to 20 ECG gated Magnetic Resonance images corresponding to one cardiac cycle of a beating heart surrounded by a static chest wall, as shown in Figure 4 (left). We modeled the visual dynamics of the heart with a 4th order LDS, and obtained the segmentation by applying GPCA to the first $d = 5$ principal components of the video sequence, as described in Section 3. Figure 4 shows the segmentation of the sequence into the chest wall (center) and the beating heart (right). Note that our algorithm performs very well using only the 5 principal components.

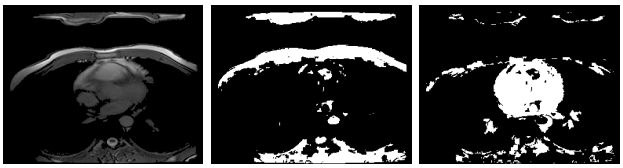

Figure 4. Segmentation of a beating heart and a static chest wall.

Segmentation of moving dynamic textures. In order to test our algorithm for optical flow and segmentation of moving dynamic textures, we generated a synthetic sequence by superimposing two real sequences, one containing moving

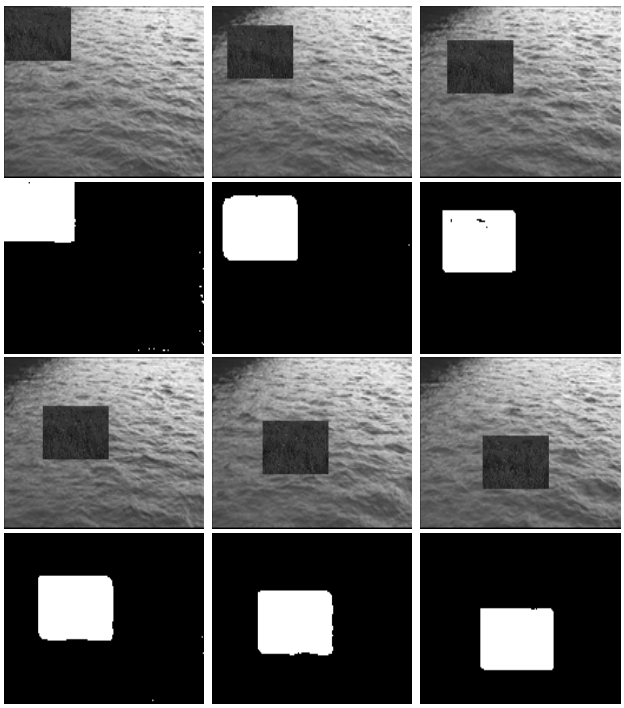

Figure 5. Segmentation of grass moving over static water.

water and the other containing the motion of grass due to wind. The grass sequence was scaled to 25% of its original size and then placed over the water sequence and moved in the x and y directions by 1 pixel/frame. Thus the resulting sequence consists of one non-moving and one moving dynamic texture. Simultaneous dynamic texture segmentation and optical flow estimation was performed using the algorithm of Section 4 with $\gamma = 5$, $n = 30$ and $\tau = 32$.

Figure 5 shows the segmentation of some frames of the sequence, which was near perfect for all frames. Figure 6 shows the estimated optical flow. For the group of pixels corresponding to the water, which is not undergoing rigid motion, we obtained an optical flow close to zero for all frames. For the group of pixels corresponding to the moving grass, the estimated optical flow in the x and y direction is approximately 0.8 pixels/frame. As before, note that the algorithm underestimates the amount of rigid motion.

6. Summary and Conclusions

We have presented a new approach to modeling scenes containing multiple dynamic textures undergoing multiple rigid-body motions. By combining time varying linear dynamical models with 2-D translational motion models, we demonstrated that it is possible to simultaneously estimate a dynamical model for each moving texture, its optical flow and the spatio-temporal segmentation of the scene using a combination of Generalized PCA with the so-called dynamic texture constancy constraint. Experiments in various video sequences demonstrated the ability of the method to estimate and segment both rigid and nonrigid motions.

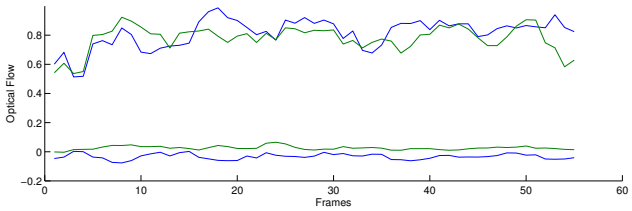

Figure 6. Optical flow estimation for grass (top) & water (bottom).

Acknowledgments

Work funded by Johns Hopkins Whiting School of Engineering startup funds and NSF CAREER Award ISS-0447739.

References

[1] M. Brand. Morphable 3D models from video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 456–463, 2001.

[2] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3D shape from image streams. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2690–2696, 2000.

[3] G. Doretto, A. Chiuso, Y. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.

[4] G. Doretto, D. Cremers, P. Favaro, and S. Soatto. Dynamic texture segmentation. In *IEEE Conference on Computer Vision*, pages 44–49, 2003.

[5] Andrew Fitzgibbon. Stochastic rigidity: Image registration for Nowhere-Static scenes. In *IEEE International Conference on Computer Vision*, pages 662–669, 2001.

[6] Takeo Kanade Jing Xiao, Jin-Xiang Chai. A closed-form solution to non-rigid shape and motion recovery. In *European Conference on Computer Vision*, pages 573–587, 2004.

[7] B. De Moor, P. Van Overschee, and J. Suykens. Subspace algorithms for system identification and stochastic realization. Technical Report ESAT-SISTA Report 1990-28, Katholieke Universiteit Leuven, 1990.

[8] S. Negahdaripour. Revised definition of optical flow: integration of radiometric and geometric clues for dynamic scene analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(9):961979, 1998.

[9] A. Schödl, R. Szeliski, D. H. Salesin, and I. Essa. Video textures. In *SIGGRAPH*, pages 489–498, 2000.

[10] S. Soatto, G. Doretto, and Y. Wu. Dynamic textures. In *IEEE International Conference on Computer Vision*, pages 439–446, 2001.

[11] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2), 1999.

[12] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[13] L. Torresani, D. Yang, E. Alexander, and C. Bregler. Tracking and modeling non-rigid objects with rank constraints. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 493–500, 2001.

[14] R. Vidal and Y. Ma. A unified algebraic approach to 2-D and 3-D motion segmentation. In *European Conference on Computer Vision*, pages 1–15, 2004.

[15] R. Vidal, Y. Ma, and J. Piazzi. A new GPCA algorithm for clustering subspaces by fitting, differentiating and dividing polynomials. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume I, pages 510–517, 2004.

[16] Lu Yuan, Fang Wen, Ce Liu, and Heung-Yeung Shum. Synthesizing dynamic texture with closed-loop linear dynamic system. In *European Conference on Computer Vision*, 2004.