SPATIOTEMPORAL MODEL-BASED OPTIC FLOW ESTIMATION [1]

Nuno Vasconcelos *Andrew Lippman*
MIT Media Laboratory, 20 Ames Street, Cambridge, MA 02139
nuno@media.mit.edu, lip@media.mit.edu

ABSTRACT

We introduce a spatiotemporal model-based algorithm capable of providing estimates of optic flow which are coherent along a set of video frames. The algorithm is based on a spatiotemporal motion model that consists of a quadratic constraint in time and an affine constraint in space. Optic flow is computed through a delayed-decision process that incorporates knowledge about both image correlation along time, and the goodness of fit to the underlying motion-model. The temporal coherence and parametric nature of the recovered optic flow can facilitate interactive access to the video stream and improve the efficiency of tasks such as video compression, interpolation or classification.

1. INTRODUCTION

Current digital video representations, such as MPEG or H.261, are geared towards compression efficiency, giving less emphasis to the semantic content of the video sequences or the ability to perform interactive operations such as browsing, filtering, sorting, or, more generally, allowing non-linear access to the video stream. The inadequacy to perform such operations is a consequence of a purely statistical model of the video sources and a computational model that is highly temporally localized, considering no more than two or three frames of the video sequence at a time.

Recent research efforts have addressed the issue of obtaining semantically more meaningful source models [1, 5]. These semantic models typically represent some of the scene attributes (such as motion and texture) in a parametric form, leading to a highly compact representation. Compactness is important, not only because it allows for efficient transmission, but also because it reduces the complexity of the analysis. Parametric models provide a transformation from the pixel space, to a smaller parameter or feature space where tasks like image segmentation are easier to perform.

Most of these efforts have, however, been carried out within the temporally localized computational model referred above. Texture analysis is typically performed in stills, and even motion analysis algorithms only occasionally rely on more than two frames at a time. This limited temporal support significantly diminishes the capability of these representations to provide some of the cues necessary for interactivity.

[1] This work was supported in part by the members of the Television of Tomorrow consortium.

The ability to access non-linearly a digital video stream is a function of the ability of the underlying video representation to support a description of significant temporal events. These significant events are, in general, associated with temporal discontinuities either in the form of scene discontinuities (scene cuts) or motion discontinuities within the same scene. As an example of the information provided by motion discontinuities consider a video shot of a baseball play. A typical play would be described by a human as: "pitcher throws the ball, batter hits the ball, ball falls on the grass, center-fielder picks up the ball, center-fielder throws the ball, 1^{st} base-man catches the ball and tags runner". Notice that the significant events in this sequence are all associated with discontinuities in the motion of the ball, and the ability to detect and signal those discontinuities is paramount for a representation that aims to provide hints about scene content to the user, or to allow the user to browse through the sequence in a non-linear fashion.

The ability to detect motion discontinuities can only be provided by a representation with a sense for motion continuity, something which is hard to achieve within the temporally localized computational model described above. In this work we relax the temporal localization constraint by introducing a spatiotemporal motion model that is valid for a sequence of frames, but retains the compactness associated with parametric representations. We believe that the spatiotemporal motion-model will make the task of detecting motion discontinuities feasible. Other tasks that can benefit from the larger temporal support associated with a spatiotemporal motion representation include compression, motion-based interpolation, classification, and generation of salient stills from video [3].

2. OPTIC FLOW ESTIMATION

The spatiotemporal optic flow estimator is based on the concept of *motion paths*. A motion path $\mathbf{p}_\kappa^{(t)} = (x_\kappa^{(t)}, y_\kappa^{(t)})^T$ is the locus of coordinates in the image plane onto which a point κ in the 3D world is projected as time evolves. To compute motion paths we rely on a quadratic motion model

$$\begin{bmatrix} x_\kappa^{(t+\delta t)} \\ y_\kappa^{(t+\delta t)} \end{bmatrix} = \begin{bmatrix} x_\kappa^{(t)} + v_{\kappa_x}\delta t + a_{\kappa_x}\delta t^2 \\ y_\kappa^{(t)} + v_{\kappa_y}\delta t + a_{\kappa_y}\delta t^2 \end{bmatrix}, \qquad (1)$$

where a_x, a_y, v_x and v_y are the horizontal and vertical components of the velocity and acceleration associated with motion path κ. Equation 1 can be seen as a second order approximation to the Taylor series expansion of $\mathbf{p}_\kappa^{(t)}$ in the neighborhood of t.

The straightforward application of the quadratic motion model does not lead to accurate recovery of the optic flow because it does not guarantee spatial smoothness. Without an additional spatial constraint, neighboring motion paths are free to take intersecting trajectories, particularly in the presence of noise. There is, therefore, a need to combine the temporal model with a spatial smoothness constraint, and we show next [2] that this is possible if we impose an affine constraint [5].

Proposition 1 *Consider a quadratic motion path $p_\kappa^{(t+\delta t)}$ determined by equation 1. Then $p_\kappa^{(t+\delta t)}$ is an affine transformation of $p_\kappa^{(t)}$ if and only if each of the motion parameters $a_{\kappa_x}, a_{\kappa_y}, v_{\kappa_x}$ and v_{κ_y} is an affine transformation of $p_\kappa^{(t)}$. I.e. for a motion path satisfying equation 1,*

$$\mathbf{p}_\kappa^{(\mathbf{t}+\delta\mathbf{t})} = \begin{bmatrix} p_{\kappa_{xx}}^{(\delta t)} & p_{\kappa_{xy}}^{(\delta t)} \\ p_{\kappa_{yx}}^{(\delta t)} & p_{\kappa_{yy}}^{(\delta t)} \end{bmatrix} \mathbf{p}_\kappa^{(\mathbf{t})} + \begin{bmatrix} p_{\kappa_{x0}}^{(\delta t)} \\ p_{\kappa_{y0}}^{(\delta t)} \end{bmatrix},$$

if and only if

$$\begin{bmatrix} v_{\kappa_x} \\ v_{\kappa_y} \\ a_{\kappa_x} \\ a_{\kappa_y} \end{bmatrix} = \begin{bmatrix} v_{xx} & v_{xy} \\ v_{yx} & v_{yy} \\ a_{xx} & a_{xy} \\ a_{yx} & a_{yy} \end{bmatrix} \begin{bmatrix} x_\kappa^{(t)} \\ y_\kappa^{(t)} \end{bmatrix} + \begin{bmatrix} v_{x0} \\ v_{y0} \\ a_{x0} \\ a_{y0} \end{bmatrix}. \qquad (2)$$

3. DELAYED-DECISION OPTIC FLOW ESTIMATION

A simple way to compute a motion path would be to compute the optical flow between consecutive frames using any of the conventional approaches and then, given a pixel in the first image, follow the cascade of motion vectors starting at that pixel. This would not be a very effective procedure because an incorrect choice of motion vector at a given frame would originate an error for all subsequent motion vectors.

A better solution is a *delayed-decision optic-flow estimator*. The principle is the same as above, but the motion vectors are no longer chosen in a greedy manner. Instead of keeping just the best motion vector at each instant, the delayed-decision algorithm keeps the best M candidate vectors. In this way, for each pixel in the first frame, it grows a M-ary tree of possible motion paths through the sequence. Each of these trees can later be searched for the best overall motion path.

Given the noisy nature of real images we cannot expect the motion model to be exactly satisfied by any of the possible paths. We can however use the closeness of a given path to the motion model as an additional criterion for the best path selection. For this, we use a least squares framework.

3.1. Least squares fit

Combining discretized versions of equations 2, and 1; assuming, without loss of generality, that the motions paths start at the instant 0, that each tree has depth K (sequence of K frames), and that there are P motion paths; and letting $\Delta x_\kappa^{([i])} = x_\kappa^{[i]} - x_\kappa^{[0]}$, we obtain [4] [3]

$$\begin{aligned}\Delta x_\kappa^{[i]} = \; & \Delta t_i{}^2 x_\kappa^{[0]} a_{xx} + \Delta t_i{}^2 y_\kappa^{[0]} a_{xy} + \Delta t_i{}^2 a_{x0} + \\ & \Delta t_i x_\kappa^{[0]} v_{xx} + \Delta t_i y_\kappa^{[0]} v_{xy} + \Delta t_i v_{x0},\end{aligned} \qquad (3)$$

where $\kappa = 1 \ldots P$, and $i = 1 \ldots K$.

In order to solve this system of equations we would need to know simultaneously the values of all the coordinates associated with all the motion paths. I.e., we would need to build all the trees associated with an object or image region before computing the fit to the motion model. Since each tree has M^K leaves and an object can cover a significant portion of the image area, this approach would rapidly become computationally infeasible. Fortunately, as we demonstrate in [4], the least-squares solution to this system of equations is separable in time and space, and the overall least squares fit can be computed in two steps as follows.

Discretizing the x-component of the system of equations 1, we obtain

$$\mathbf{\Delta}_\kappa = \mathbf{T} \begin{bmatrix} a_{\kappa_x} \\ v_{\kappa_x} \end{bmatrix} = \mathbf{T}\,\mathbf{F}_\kappa, \qquad (4)$$

where

$$\mathbf{T} = \begin{bmatrix} | & | \\ \Delta_{t_i}^2 & \Delta_{t_i} \\ | & | \end{bmatrix}, \quad \mathbf{\Delta}_\kappa = \begin{bmatrix} | \\ \Delta x_\kappa^{[i]} \\ | \end{bmatrix},$$

$\kappa = 1 \ldots P$, and $i = 1 \ldots K$. We first compute the least-squares solution to this system of equations [2]

$$\hat{\mathbf{F}}_\kappa = (\mathbf{T}^T\mathbf{T})^{-1}\mathbf{T}^T\mathbf{\Delta}_\kappa. \qquad (5)$$

Now, from equation 2

$$\mathbf{F}_\kappa = \mathbf{P}_\kappa\,\mathbf{A}_{\mathbf{xaff}},$$

where

$$\mathbf{P}_\kappa = \begin{bmatrix} x_\kappa^{[0]} & y_\kappa^{[0]} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_\kappa^{[0]} & y_\kappa^{[0]} & 1 \end{bmatrix}$$

and $\mathbf{A}_{\mathbf{xaff}} = (a_{xx}, a_{xy}, a_{x0}, v_{xx}, v_{xy}, v_{x0})^T$.

Combining the previous equations for all values of κ, we obtain

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_1 \\ \vdots \\ \mathbf{F}_P \end{bmatrix} = \begin{bmatrix} \mathbf{P}_1 \\ \vdots \\ \mathbf{P}_P \end{bmatrix} \mathbf{A}_{\mathbf{xaff}} = \mathbf{N}\,\mathbf{A}_{\mathbf{xaff}}. \qquad (6)$$

Given the least-squares estimate of $\mathbf{F}_\kappa$ obtained from equation 5, the least-squares solution to this system of equations is

$$\hat{\mathbf{A}}_{\mathbf{xaff}} = (\mathbf{N}^T\mathbf{N})^{-1}\mathbf{N}^T\hat{\mathbf{F}}. \qquad (7)$$

[2] The proof of the proposition can be found in [4]

[3] We analyze only the least squares fit to the x component the results are similar for the y component.

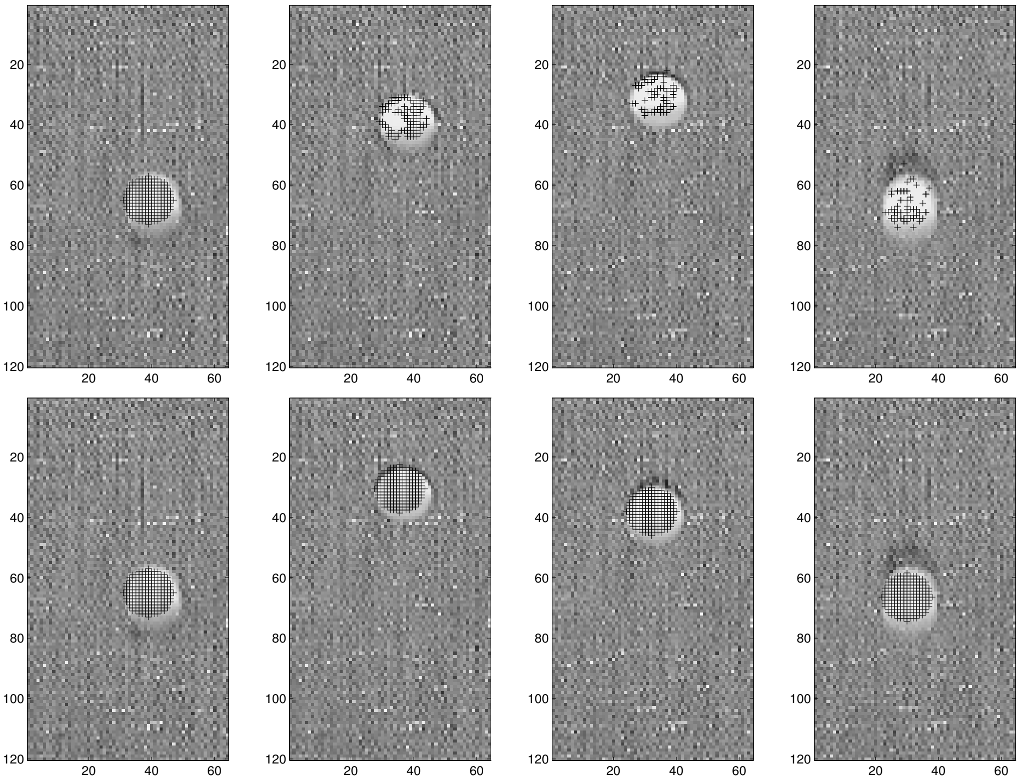


Figure 1. Top: Motion trajectories associated with unconstrained motion paths. Bottom: Trajectories estimated by the model-based algorithm. In this example the affine model is restricted to translation and, in the first frame, the ball was manually segmented out of the background.

The overall least-squares problem is, in this way, divided into two smaller sub-problems. First, for each motion path we find the vector of quadratic motion parameters that solves equation 4 in a least-squares sense. Notice that for this computation we only need to consider one tree at a time. We then use the motion parameters associated with all motion paths to find the least-squares affine fit $\hat{\mathbf{A}}_{\mathbf{aff}}$ of equation 7. I.e., we decouple the computation of the parameters relative to the temporal constraint from those relative to the spatial constraint, without compromising the optimality of the least-squares fit [4].

3.2. The algorithm

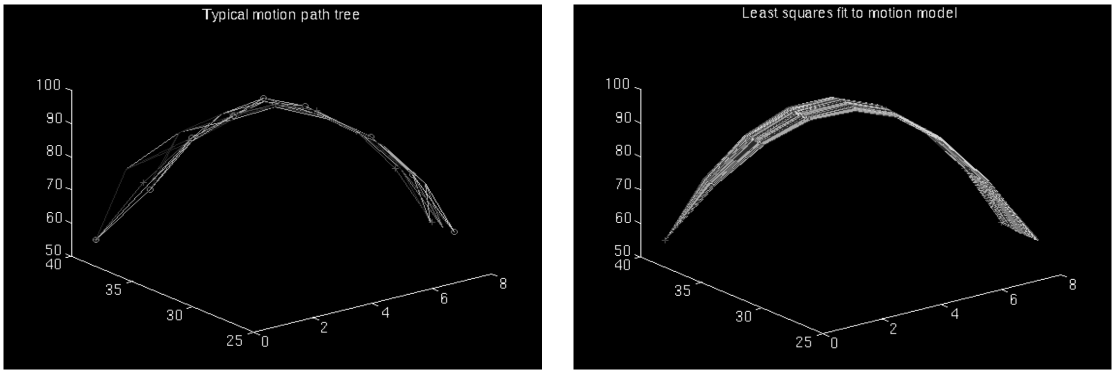
In this paper, we implement the delayed-decision motion estimator with dense block matching. We start by applying block-matching between pairs of consecutive frames, and grow the motion trees by keeping the *best* M vectors at each pixel. For each tree, we then select the path which minimizes a cost function that incorporates both the cumulative mean square block-matching error and the error of the least squares fit to the motion model:

$$\min_{\mathbf{p}_\kappa} \left(\mathcal{L}^{(\mathbf{p}_\kappa)} + \lambda\, \mathcal{D}^{(\mathbf{p}_\kappa)} \right), \tag{8}$$

where $\mathbf{p}_\kappa = [a_{\kappa_x}, a_{\kappa_y}, v_{\kappa_x}, v_{\kappa_x}]^T$ is the vector of quadratic motion parameters associated with the motion path $\mathbf{p}_\kappa^{[i]}$, $\mathcal{L}^{(\mathbf{p}_\kappa)}$ is the cumulative mean-square correlation error along the path, and $\mathcal{D}^{(\mathbf{p}_\kappa)}$ is the error of the least squares fit. We next perform the second step of the least squares fit, this time in the temporal dimension, using equation 7. In this way, we impose the affine constraint in space, and from proposition 1 the optical flow is affine for every frame in the sequence.

4. SIMULATION RESULTS

The top row of figure 1 presents frames 1, 3, 6 and 9 of a sequence of 9 frames consisting of a bouncing ping-pong ball and a textured background. The crosses superimposed on the figure represent the trajectories followed by the motion paths in the absence of motion model, i.e. when the best motion path is simply the one which maximizes the correlation across frames. Notice that, due to the textureless nature of the ball, these paths tend to cross each other and even follow a common route. Figure 2 depicts a typical motion path tree rooted in one of the pixels of the first frame, and the least squares approximation after fitting the quadratic motion model according to equation 5. This and the remaining trees associated with the ball were then pruned according to equation 8 and the resulting spatial motion parameters fitted to the spatial affine model according to equation 7. The spatial fitting is illustrated in figure 3. Finally, the bottom row of figure 1 presents the trajectories obtained with the model-based algorithm, superimposed on the original sequence. The accuracy of these trajectories is



a) b)

Figure 2. a) tree of motion paths, b) least square approximations. In both graphs the vertical and horizontal axis represent image position while depth is associated with time.

very good, in contrast to that of the trajectories on the top row of the figure.

REFERENCES

[1] F. Liu and R. Picard. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. Technical Report 320, MIT Media Laboratory Perceptual Computing Section, 1995.

[2] G. Strang. *Linear Algebra and its Applications.* Harcourt Brace Jovanovich, Inc., 1985.

[3] L. Teodosio. Salient Stills. Master's thesis, MIT Media Lab, 1992.

[4] N. Vasconcelos. Spatiotemporal Model-Based Optic Flow Estimation. Technical report, MIT Media Lab, December 1994.

[5] J. Wang and E. Adelson. Representing Moving Images with Layers. *IEEE Trans. on Image Processing*, Vol. 3, September 1994.

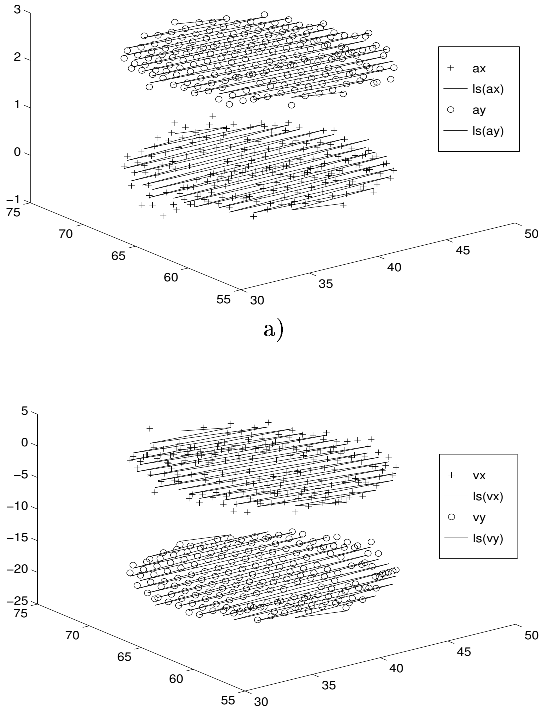


Figure 3. Quadratic motion parameters associated with best motion paths and associated affine fit. a) acceleration b) velocity. In these graphs the horizontal and depth axis represent the spatial image coordinates and the vertical axis the value of the motion parameter at those coordinates. Circles and crosses are measured parameters while the lines represent the least squares affine fits.