Pel-Recursive Motion Estimation Using the Expectation-Maximization Technique and Spatial Adaptation

Vania Estrela[(1)], Luis A. Rivera[(1)], Marcos H. S. Bassani[(2)]
[(1)]Universidade Estadual do Norte Fluminense, LCMAT-CCT, Grupo de Computação Científica
Av. Alberto Lamego 2000, Campos dos Goytacazes,
RJ, Brasil, CEP 28013-600
{vaniave,rivera}@uenf.br

[(2)]Departamento de Engenharia Mecânica, DEPES,CEFET-RJ
Av. Maracanã 229, Rio de Janeiro, RJ, BRAZIL, CEP 20271-110
marcos@cefet-rj.br

ABSTRACT

Pel-recursive motion estimation is a well-established approach. However, in the presence of noise, it becomes an ill-posed problem that requires regularization. In this paper, motion vectors are estimated in an iterative fashion by means of the Expectation-Maximization (EM) algorithm and a Gaussian data model. Our proposed algorithm also utilizes the local image properties of the scene to improve the motion vector estimates following a spatially adaptive approach. Numerical experiments are presented that demonstrate the merits of our methods.

Keywords
Motion estimation, EM algorithm, pel-recursive, maximum likelihood.

1. INTRODUCTION

Motion estimation is very important in multimedia video processing applications. For example, in video coding, the estimated motion is used to reduce the transmission bandwidth. The evolution of an image sequence motion field can also help other image processing tasks in multimedia applications such as analysis, recognition, tracking, restoration, collision avoidance and segmentation of objects [Jain89].

In coding applications, a block-based approach [Tek95] is often used for interpolation of lost information between key frames. The fixed rectangular partitioning of the image used by some block-based approaches often separates visually meaningful image features. If the components of an important feature are assigned different motion vectors, then the interpolated image will suffer from annoying artifacts.

Pel-recursive schemes [Brailean95, Estrela03, Jain89] can theoretically overcome some of the limitations associated with blocks by assigning a unique motion vector to each pixel. Intermediate frames are then constructed by resampling the image at locations determined by linear interpolation of the motion vectors. The pel-recursive approach can also manage motion with sub-pixel accuracy. However, its original formulation was deterministic. The update of the motion estimate was based on the minimization of the displaced frame difference (DFD) at a pixel. In the absence of additional assumptions about the pixel motion, this estimation problem becomes ill-posed because of the following problems: a) occlusion; b) the solution to the 2D motion estimation problem is not unique; and c) the solution does not continuously depend on the data due to the fact that motion estimation is highly sensitive to the presence of observation noise in video images.

In this work, we plan to use the MAP estimate to find the update of the motion vector from our observation model by means of the Expectation-Maximization technique. The main advantages of the EM method is that the final algorithm deals with

WSCG'2004, February 2-6, 2003, Plzen, Czech Republic.

closed-form expressions and it does not require the use of optimization techniques.

We organized this work as follows. Section 2 provides some necessary background on the pel-recursive motion estimation problem. Section 3 introduces our spatially adaptive approach. Section 4 describes the EM framework. Section 5 defines the metrics used to evaluate our results. Section 6 describes some implementation aspects and the experiments used to access the performance of our proposed algorithm. Finally, Section 7 has some conclusions.

2. PEL-RECURSIVE DISPLACEMENT ESTIMATION

The displacement of each picture element (pel or pixel) in each frame forms the displacement vector field (DVF) and its estimation can be done using at least two successive frames. The DVF is the 2D motion resulting from the apparent motion of the image brightness. A vector is assigned to each point in the image.

A pixel belongs to a moving area if its intensity has changed between consecutive frames. Hence, our goal is to find the corresponding intensity value $I_k(\boldsymbol{r})$ of the k-th frame at location $\boldsymbol{r} = [x, y]^T$, and the corresponding displacement vector (DV) at the working point $\boldsymbol{r}$ in the current frame, that is $\boldsymbol{d}(\boldsymbol{r}) = [d_x, d_y]^T$. Pel-recursive algorithms minimize the DFD function in a small area containing the working point assuming constant image intensity along the motion trajectory. The DFD is defined by

$$\Delta(\boldsymbol{r}; \boldsymbol{d}(\boldsymbol{r})) = I_k(\boldsymbol{r}) - I_{k-1}(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r})) \quad (1)$$

and the perfect registration of frames will result in $I_k(\boldsymbol{r})=I_{k-1}(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r}))$. The DFD represents the error due to the nonlinear temporal prediction of the intensity field through the DV. The relationship between the DVF and the intensity field is nonlinear. An estimate of $\boldsymbol{d}(\boldsymbol{r})$, is obtained by directly minimizing $\Delta(\boldsymbol{r},\boldsymbol{d}(\boldsymbol{r}))$ or by determining a linear relationship between these two variables through some model. This is accomplished by means of a Taylor series expansion of $I_{k-1}(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r}))$ about the location $(\boldsymbol{r}-\boldsymbol{d}^i(\boldsymbol{r}))$, where $\boldsymbol{d}^i(\boldsymbol{r})$ represents a prediction of $\boldsymbol{d}(\boldsymbol{r})$ in the i-th step. This results in

$$\Delta(\boldsymbol{r},\boldsymbol{r}-\boldsymbol{d}^i(\boldsymbol{r}))=-\boldsymbol{u}^T\nabla I_{k-1}(\boldsymbol{r}-\boldsymbol{d}^i(\boldsymbol{r})) + e(\boldsymbol{r},\boldsymbol{d}(\boldsymbol{r})), \quad (2)$$

where the displacement update vector $\boldsymbol{u}=[u_x, u_y]^T = \boldsymbol{d}(\boldsymbol{r}) - \boldsymbol{d}^i(\boldsymbol{r})$; $e(\boldsymbol{r}, \boldsymbol{d}(\boldsymbol{r}))$ represents the error resulting from the truncation of the higher order terms (linearization error); and $\nabla=[\partial/\partial_x, \partial/\partial_y]^T$ represents the spatial gradient operator. Applying (2) to all points in a neighborhood $\mathcal{R}$ gives

$$\boldsymbol{z} = \boldsymbol{G}\boldsymbol{u} + \boldsymbol{n}, \quad (3)$$

where the temporal gradients $\Delta(\boldsymbol{r}, \boldsymbol{r}-\boldsymbol{d}^i(\boldsymbol{r}))$, from (2), have been stacked to form the $N\times1$ observation vector $\boldsymbol{z}$ containing DFD information on all the pixels in a neighborhood $\mathcal{R}$, the $N\times2$ matrix $\boldsymbol{G}$ is obtained by stacking the spatial gradient operators at each observation, and the error terms have formed the $N\times1$ noise vector $\boldsymbol{n}$ which is assumed Gaussian with probability density function (pdf) $\boldsymbol{n}\sim N(0, \sigma_n^2\boldsymbol{I})$. Each row of $\boldsymbol{G}$ has entries $[g_{xi}, g_{yi}]^T$, with $i = 1, \ldots, N$. The spatial gradients of I_{k-1} are calculated through a bilinear interpolation scheme [Brailean95].

3. SPATIAL ADAPTION

Aiming to improve the estimates given by the pel-recursive algorithm, we introduced an adaptive scheme for determining the optimal shape of the neighborhood of pixels overdetermined system of equations given by (3).

Errors can be caused by the basic underlying assumption of uniform motion inside $\mathcal{R}$ (the smoothness constraint), by not grouping pixels adequately, and by the way gradient vectors are estimated, among other things. Since it is known that in a noiseless image not containing pixels with constant intensity, most errors, when estimating motion, occur close to motion boundaries. We propose a hypothesis testing approach to determine the best neighborhood shape for a given pixel. The masks in Figure 1 show the geometries of the neighborhoods used, where "x" indicates the position of the current pixel and "0" is a neighboring pixel. We pick up the neighborhood from the finite set of templates shown in Figure 1, according to the smallest $|DFD|$ criterion, in an attempt to adapt the model to local features associated to motion boundaries. The chosen mask is the one that better satisfies the smoothness constraint, capturing the motion behaviour around the current pixel, and, therefore accounting for motion discontinuities.

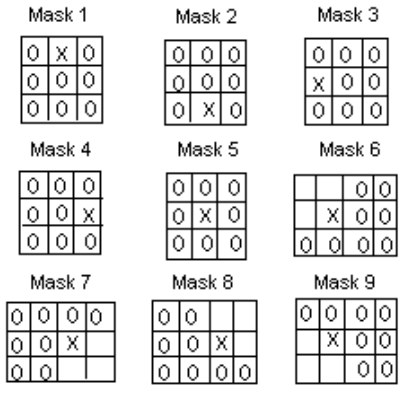

Figure 1. Neighborhood geometries.

4. THE EM APPROACH

The EM algorithm is a general numerical technique which can be used to determine the maximum likelihood estimate (MLE) of a set of parameters. It can be employed for identification of model and distribution parameters simultaneously. The parameters can be estimated iteratively even in situations where some variables cannot be observed.

4.1 The Ordinary Maximum Likelihood Estimate and Its Limitations

The vector $\boldsymbol{u}$ of (3) can be computed using the MAP estimate [Kay93] as

$$\hat{\boldsymbol{u}}_{MAP} = \boldsymbol{\Lambda}_u \boldsymbol{G}^T (\boldsymbol{G}\boldsymbol{\Lambda}_u \boldsymbol{G}^T + \boldsymbol{\Lambda}_n)^{-1} \boldsymbol{z}\,, \tag{4}$$

where $\boldsymbol{\Lambda}_u$ and $\boldsymbol{\Lambda}_n$ are, respectively, the covariance matrices of the update vector and of the noise/linearization error term.

The estimate in (4) was derived assuming that $\boldsymbol{u}$ and $\boldsymbol{n}$ are zero mean, and uncorrelated. The maximum a posteriori (MAP) estimate for the linear observation model in (3) is also the MAP estimate assuming a Gaussian prior on $\boldsymbol{u}$ and $\boldsymbol{n}$.

In this work, we resort to the maximum likelihood (ML) estimation of the second-order statistics $\boldsymbol{\Lambda}_u$ and $\boldsymbol{\Lambda}_n$ of the model. The calculation of the ML estimates of $\boldsymbol{\Lambda}_u$ and $\boldsymbol{\Lambda}_n$ is done iteratively by means of the Expectation-Maximization (EM) algorithm formalized by Dempster et al [Dem77] which has been used in a variety of applications as referenced in [Kat91]. The EM algorithm was previously used for motion estimation in [Fan96]. However, this work estimates parameters of affine models and it uses a block-matching framework.

For the model described by (3), let us assume that the update vector $\boldsymbol{u}$ and the noise $\boldsymbol{n}$ are normally-distributed with mean zero, and uncorrelated, with covariance matrices equal to $\boldsymbol{\Lambda}_u$ and $\boldsymbol{\Lambda}_n$ respectively. Then, the pdf of $\boldsymbol{z}$ is given by

$$f_z(\boldsymbol{z}) = \left|2\pi(\boldsymbol{G}\boldsymbol{\Lambda}_u\boldsymbol{G}^H + \boldsymbol{\Lambda}_n)\right|^{-\frac{1}{2}} \times exp\left\{-\frac{1}{2}\boldsymbol{z}^H(\boldsymbol{G}\boldsymbol{\Lambda}_u\boldsymbol{G}^H + \boldsymbol{\Lambda}_n)^{-1}\boldsymbol{z}\right\}.$$

where $\boldsymbol{G}^H$ *and* $\boldsymbol{z}^H$ are the Hermitian conjugates of $\boldsymbol{G}$ and $\boldsymbol{z}$ [Kat91].

Let us take $\boldsymbol{\Phi} = \{\boldsymbol{\Lambda}_u, \boldsymbol{\Lambda}_n\}$ as the parameter set to be estimated. The pdf of $\boldsymbol{z}$ can be considered a continuous and differentiable function of $\boldsymbol{\Phi}$. This dependency can be better captured by the notation $f_z(\boldsymbol{z};\ \boldsymbol{\Phi})$. Furthermore, we can assume that the additive noise is white, with covariance matrix $\boldsymbol{\Lambda}_n = \sigma_n^2\boldsymbol{I}$, where $\boldsymbol{I}$ is an $N \times N$ identity matrix.

The ML estimation of $\boldsymbol{\Phi}$ is the $\boldsymbol{\Phi}_{ML}$ that maximizes the logarithm of the likelihood function of $f_z(\boldsymbol{z};\boldsymbol{\Phi})$, which is given by

$$\begin{aligned}\boldsymbol{\Phi}_{ML} &= arg\{\max_{\boldsymbol{\Phi}} f_z(\boldsymbol{z};\boldsymbol{\Phi})\} \\ &= arg\{\max_{\boldsymbol{\Phi}} log\ f_z(\boldsymbol{z};\boldsymbol{\Phi})\}.\end{aligned} \tag{5}$$

Combining (4) and (5), we conclude that the maximization of the log-likelihood function is equivalent to minimizing the function $L_o(\boldsymbol{\Phi})$ where

$$\begin{aligned}L_o(\boldsymbol{\Phi}) = log\left|\boldsymbol{G}\boldsymbol{\Lambda}_u\boldsymbol{G}^H + \boldsymbol{\Lambda}_n\right| \\ + \boldsymbol{z}^H(\boldsymbol{G}\boldsymbol{\Lambda}_u\boldsymbol{G}^H + \boldsymbol{\Lambda}_n)^{-1}\boldsymbol{z}.\end{aligned} \tag{6}$$

This function is nonlinear and non-unimodal with respect to $\boldsymbol{\Phi}$. Analytical solutions for (6) are out of question. Thus, we have to resort to optimization techniques. Nevertheless, since $L_o(\boldsymbol{\Phi})$ is not unimodal, convergence and local extrema of iterative optimization methods are serious problems. The EM algorithm arises as an alternative for estimating $\boldsymbol{\Phi}$, since its convergence to a local maximum is guaranteed.

4.2 Problem Formulation According to the EM Framework

We assume that the update vector $\boldsymbol{u}$ is normally-distributed with mean zero and covariance matrix equal to $\boldsymbol{\Lambda}_u$, and that $\boldsymbol{n} \sim N(\boldsymbol{0},\ \sigma_n^2\boldsymbol{I})$ is the additive Gaussian noise.

If we choose $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{u} & \boldsymbol{n}\end{bmatrix}^T$ as the complete data where $\boldsymbol{x} \in \mathbb{R}^{(N+2)\times 1}$, then $\boldsymbol{x}$ will be normally distributed with diagonal covariance matrix $\boldsymbol{\Lambda}_x$ equal to

$$\boldsymbol{\Lambda}_x = \begin{bmatrix}\boldsymbol{\Lambda}_u & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Lambda}_n\end{bmatrix} \text{ and } \boldsymbol{\Lambda}_x^{-1} = \begin{bmatrix}\boldsymbol{\Lambda}_u^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Lambda}_n^{-1}\end{bmatrix},$$

with this, (3) can be re-written as

$$\boldsymbol{z} = [\boldsymbol{G};\boldsymbol{I}]\boldsymbol{x} = \boldsymbol{H}\boldsymbol{x}, \text{ for } \boldsymbol{H} = [\boldsymbol{G};\boldsymbol{I}].$$

The foundation of the EM algorithm is the maximization of the expectation of $log\{f_x(\boldsymbol{x};\boldsymbol{\Phi})\}$, given the incomplete observed data $\boldsymbol{z}$ and the current estimate of the parameter set $\boldsymbol{\Phi}$. $f_x(\boldsymbol{x};\boldsymbol{\Phi})$ is the pdf of the complete data, and it is given by

$$f_x(\boldsymbol{x};\boldsymbol{\Phi}) = \left|2\pi\boldsymbol{\Lambda}_x\right|^{-\frac{1}{2}} exp\left\{-\frac{1}{2}\boldsymbol{x}^H\boldsymbol{\Lambda}_x^{-1}\boldsymbol{x}\right\}, \tag{7}$$

that taking the logarithm of both sides leads to

$$\begin{aligned}log\{f_x(\boldsymbol{x};\boldsymbol{\Phi})\} &= -\frac{1}{2}log\left|2\pi\boldsymbol{\Lambda}_x\right| - \frac{1}{2}\boldsymbol{x}^H\boldsymbol{\Lambda}_x^{-1}\boldsymbol{x} \\ &= K - \frac{1}{2}[\boldsymbol{\Lambda}_x + \boldsymbol{x}^H\boldsymbol{\Lambda}]\end{aligned}$$

where $\boldsymbol{\Phi} = \boldsymbol{\Lambda}_x$ and K is a constant independent of $\boldsymbol{\Phi}$.

4.1.1 EM algorithm

We want to compute iteratively

$$\boldsymbol{\Phi}^{(p+1)} = arg\{\max_{\boldsymbol{\Phi}} Q(\boldsymbol{\Phi};\boldsymbol{\Phi}^{(p)})\}. \quad (8)$$

For that, we need to compute

$$Q(\boldsymbol{\Phi};\boldsymbol{\Phi}^{(p)}) = E[log\{f_x(\boldsymbol{x};\boldsymbol{\Phi})\}|\boldsymbol{z};\boldsymbol{\Phi}^{(p)}], \quad (9)$$

where $\boldsymbol{\Phi}^{(p)}=[\boldsymbol{\Lambda}_u^{(p)};\boldsymbol{\Lambda}_n^{(p)}]$ is the estimate of the parameter set $\boldsymbol{\Phi}=\{\boldsymbol{\Lambda}_u;\boldsymbol{\Lambda}_n\}$ at the p-th iteration.

We compute (9) and (8) in E-step and M-step, respectively.

E-Step:

This step can be re-written in terms of the parameter set $\boldsymbol{\Phi}$ by means of the relationships developed in terms of the complete data $\boldsymbol{x}$ and the pdf $f(\boldsymbol{x}|\boldsymbol{z};\boldsymbol{\Phi}^{(p)})$ as

$$\begin{aligned} F(\boldsymbol{\Phi};\boldsymbol{\Phi}^{(p)}) &= log|\boldsymbol{\Lambda}_u| + log|\boldsymbol{\Lambda}_n| + tr(\boldsymbol{\Lambda}_u^{-1}\boldsymbol{\Lambda}_{u|z}^{(p)}) \\ &+ tr(\boldsymbol{\Lambda}_n^{-1}\boldsymbol{\Lambda}_{n|z}^{(p)}) + \mu_{u|z}^{(p)H}\boldsymbol{\Lambda}_u^{-1}\mu_{u|z}^{(p)} \\ &+ \mu_{n|z}^{(p)H}\boldsymbol{\Lambda}_n^{-1}\mu_{n|z}^{(p)} \end{aligned} \quad (10)$$

Our goal is to estimate the update vector $\boldsymbol{u}$. The MAP/MMSE estimate of $\boldsymbol{u}$ is $\mu_{u|z}^{(p)}$ which corresponds to its conditional expectation at iteration p, given the observation $\boldsymbol{z}$. Since our observation model is linear and Gaussian statistics are assumed, we have $\hat{\boldsymbol{u}}_{MMSE} = \hat{\boldsymbol{u}}_{LMMSE} = \hat{\boldsymbol{u}}_{MAP}$.

This estimate is obtained as a byproduct of the estimation of $\boldsymbol{\Phi}$. The statistical assumptions about $\boldsymbol{u}$ and $\boldsymbol{n}$ result in $\boldsymbol{\Phi}=\{\sigma_1^2, \sigma_2^2, \sigma_n^2\}$. Therefore,

$$\begin{aligned} F(\boldsymbol{\Phi};\boldsymbol{\Phi}^{(p)}) &= log(\sigma_1^2) + log(\sigma_2^2) + \\ & m[log(\sigma_n^2)] + \frac{a_{11}^{(p)} + c_1^{2(p)}}{\sigma_1^2} + \frac{a_{22}^{(p)} + c_2^{2(p)}}{\sigma_2^2} \\ &+ \frac{1}{\sigma_n^2}[b_{11}^{(p)} + \cdots + b_{mm}^{(p)} + e_1^{2(p)} + \cdots + e_m^{2(p)}] \end{aligned} \quad (11)$$

where

$$\begin{aligned} \mathbf{A}^{(p)} &= \left[\Lambda_u^{(p)} - \Lambda_u^{(p)}\mathbf{G}^H(\mathbf{G}\Lambda_u^{(p)}\mathbf{G}^H + \Lambda_n^{(p)})^{-1}\mathbf{G}\Lambda_u^{(p)}\right], \\ \mathbf{A}^{(p)} &= \left[\Lambda_u^{(p)} - \Lambda_u^{(p)}\mathbf{G}^H(\mathbf{G}\Lambda_u^{(p)}\mathbf{G}^H + \Lambda_n^{(p)})^{-1}\mathbf{G}\Lambda_u^{(p)}\right], \\ \mathbf{B}^{(p)} &= \left[\Lambda_n^{(p)} - \Lambda_n^{(p)}\left(\mathbf{G}\Lambda_u^{(p)}\mathbf{G}^H + \Lambda_n^{(p)}\right)^{-1}\Lambda_n^{(p)}\right], \\ \mathbf{c}^{(p)} &= \mu_{u|z}^{(p)} = \left[\Lambda_u^{(p)}\mathbf{G}^H(\mathbf{G}\Lambda_u^{(p)}\mathbf{G}^H + \Lambda_n^{(p)})^{-1}\mathbf{z}\right] \text{ and} \\ \mathbf{e}^{(p)} &= \mu_{n|z}^{(p)} = \left[\Lambda_n^{(p)}(\mathbf{G}\Lambda_u^{(p)}\mathbf{G}^H + \Lambda_n^{(p)})^{-1}\mathbf{z}\right]. \end{aligned} \quad (12)$$

The previous matrices can be simplified and written as

$$\mathbf{A}^{(p)} = \begin{bmatrix} a_{11}^{(p)} & a_{12}^{(p)} \\ a_{21}^{(p)} & a_{22}^{(p)} \end{bmatrix}, \quad \boldsymbol{B}^{(p)} = \begin{bmatrix} b_{11}^{(p)} & \cdots & b_{1m}^{(p)} \\ \vdots & \ddots & \vdots \\ b_{m1}^{(p)} & \cdots & b_{mm}^{(p)} \end{bmatrix},$$

$$\boldsymbol{c}^{(p)} = \begin{bmatrix} c_1^{(p)} & c_2^{(p)} \end{bmatrix}^T \text{ and } \boldsymbol{e}^{(p)} = \begin{bmatrix} e_1^{(p)} & \cdots & e_m^{(p)} \end{bmatrix}^T.$$

M-Step:

The minimum of $F(\boldsymbol{\Phi};\boldsymbol{\Phi}^{(p)})$ occurs when $\frac{\partial F(\boldsymbol{\Phi};\boldsymbol{\Phi}^{(p)})}{\partial \sigma_n^2} = 0$. So,

$$\sigma_n^{2(p+1)} = \frac{1}{m}\left\{Tr\left[\boldsymbol{B}^{(p)}\right] + \left|\boldsymbol{e}^{(p)}\right|^2\right\}. \quad (13)$$

Applying similar procedure for σ_1^2 and σ_2^2 gives

$$\sigma_1^{2(p+1)} = a_{11}^{(p)} + c_1^{2(p)} \text{ and } \sigma_2^{2(p+1)} = a_{22}^{(p)} + c_2^{2(p)}. \quad (14)$$

Now, we are ready to state the resulting EM algorithm using multiple masks.

4.3 The EM-Based Pel Recursive Motion Estimation Algorithm

For each pixel in the current frame k, located at $\boldsymbol{r}$, do the following:

```
Initialize: d^0(r), σ_n^{2(0)}, σ_1^{2(0)} and σ_2^{2(0)};
   m = p ← 0 (mask and iteration counter);
   thresholds T, ε and ξ.
Do {
   Calculate G^p, z^p for current mask
   Calculate c^(p) from (12)
   Perform M-Step
   Calculate d^{p+1}(r) = d^p(r) + u^p
   p ← p + 1
   If ( p = MAX ) then
      m←m+1  (try another neighborhood)
      reset d^0(r), σ_n^{2(0)}, σ_1^{2(0)}, σ_2^{2(0)}, p ← 0
   end if
   if (all masks where used), d^{p+1}(r) = 0
} while ( ||Φ^{p+1} − Φ^p|| ≤ ξ and
          ||d^{p+1}(r) − d^p(r)|| ≤ ε and |DFD| < T
          and p < MAX )
stop.
```

5. METRICS

This work assesses the motion field quality through the use of the four metrics [Brailean95, Estrela03] as described below.

5.1 Mean Squared Error (MSE)

Since the MSE provides an indication of the degree of correspondence between the estimates and the true value of the motion vectors, we can apply this measure to two consecutive frames of a sequence with known motion. We can evaluate the MSE in the horizontal (MSE_x) and in the vertical (MSE_y) directions as follows:

$$MSE_x = \frac{1}{RC}\sum_{\boldsymbol{r}\in\boldsymbol{S}}[d_x(\boldsymbol{r}) - \hat{d}_x(\boldsymbol{r})]^2 \text{ , and}$$

$$MSE_y = \frac{1}{RC}\sum_{\boldsymbol{r}\in\boldsymbol{S}}[d_y(\boldsymbol{r}) - \hat{d}_y(\boldsymbol{r})]^2 ,$$

where $\boldsymbol{S}$ is the entire frame, $\boldsymbol{r}$ represents the pixel coordinates, R and C are, respectively, the number of rows and columns in a frame, $\boldsymbol{d}(\boldsymbol{r})=(d_x(\boldsymbol{r}),\ d_y(\boldsymbol{r}))$ is the true deviation vector at $\boldsymbol{r}$, and $\hat{\boldsymbol{d}}(\boldsymbol{r}) = (\hat{d}_x(\boldsymbol{r}), \hat{d}_y(\boldsymbol{r}))$ its estimation.

5.2 Bias

The bias gives an idea of the degree of correspondence between the estimated motion field and the original optical flow. It is defined as the average of the difference between the true DV's and their predictions, for all pixels inside a frame S, and it is defined along the x and y directions as

$$bias_x = \frac{1}{RC}\sum_{\boldsymbol{r}\in\boldsymbol{S}}\left[d_x(\boldsymbol{r}) - \hat{d}_x(\boldsymbol{r})\right], \text{ and}$$

$$bias_y = \frac{1}{RC}\sum_{\boldsymbol{r}\in\boldsymbol{S}}\left[d_y(\boldsymbol{r}) - \hat{d}_y(\boldsymbol{r})\right]$$

5.3 Mean-Squared Displaced Frame Difference

This metric evaluates the behavior of the average of the squared displaced frame difference ($\overline{DFD}^2$). It represents an assessment of the evolution of the temporal gradient as the scene evolves by looking at the squared difference between the current intensity $I_k(\boldsymbol{r})$ and its predicted value $I_{k-1}(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r}))$. Ideally, the $\overline{DFD}^2$ should be zero, which means that all motion was identified correctly ($I_k(\boldsymbol{r})= I_{k-1}(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r}))$ for all $\boldsymbol{r}$'s). In practice, we want the $\overline{DFD}^2$ to be as low as possible. Its is defined as

$$\overline{DFD}^2 = \frac{\sum_{k=2}^{K}\sum_{\boldsymbol{r}\in\boldsymbol{S}}\left[I_k(\boldsymbol{r}) - I_{k-1}\left(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r})\right)\right]^2}{RC\left(K-1\right)},$$

where K is the length of the image sequence.

5.3 Improvement in Motion Compensation

The improvement in motion compensation ($\overline{IMC(dB)}$) between two consecutive frames is given by

$$\overline{IMC}_k(dB) = 10\log_{10}\left\{\frac{\sum_{\boldsymbol{r}\in\boldsymbol{S}}\left[I_k(\boldsymbol{r}) - I_{k-1}(\boldsymbol{r})\right]^2}{\sum_{\boldsymbol{r}\in\boldsymbol{S}}\left[I_k(\boldsymbol{r}) - I_{k-1}\left(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r})\right)\right]^2}\right\},$$

where S is the frame being currently analyzed. It shows the ratio in decibel (dB) between the mean-squared frame difference ($\overline{FD}^2$) defined by

$$\overline{FD}^2 = \frac{\sum_{\boldsymbol{r}\in\boldsymbol{S}}[I_k(\boldsymbol{r}) - I_{k-1}(\boldsymbol{r})]^2}{RC}$$

and the $\overline{DFD}^2$ between frames k and (k-1).

As far as the use of the this metric goes, we chose to apply it to a sequence of K frames, resulting in the following equation for the average improvement in motion compensation:

$$\overline{IMC}(dB) = 10\log_{10}\left\{\frac{\sum_{k=2}^{K}\sum_{\boldsymbol{r}\in\boldsymbol{S}}\left[I_k(\boldsymbol{r}) - I_{k-1}(\boldsymbol{r})\right]^2}{\sum_{k=2}^{K}\sum_{\boldsymbol{r}\in\boldsymbol{S}}\left[I_k(\boldsymbol{r}) - I_{k-1}\left(\boldsymbol{r}-\boldsymbol{d}(\boldsymbol{r})\right)\right]^2}\right\}$$

When it comes to motion estimation, we seek algorithms that have high values of $\overline{IMC}(dB)$. If we could detect motion without any error, then the denominator of the previous expression would be zero (perfect registration of motion) and we would have $\overline{IMC(dB)}=\infty$.

6. IMPLEMENTATION

In this section, we present several experimental results illustrating the effectiveness of the EM algorithm and compare it with the Wiener filter described by

$$\hat{\mathbf{u}}_{Wiener} = \hat{\mathbf{u}}_{LMMSE} = (\mathbf{G}^T\mathbf{G} + \mu\mathbf{I})^{-1}\mathbf{G}^T\mathbf{z} ,$$

with μ=50 in which the statistics of $\boldsymbol{\Lambda}_u$ and $\boldsymbol{\Lambda}_n$ remain constant for the entire frame [Bie87]. As before, the algorithms were tested on a synthetic sequence and two real video sequences: the "Foreman" and the "Mother and Daughter". The sequences are 144 x 176, 8-bit (QCIF).

6.1 Experiment 1

In this sequence, there is a moving rectangle immersed in a moving background. In order to create textures for the rectangle and its background (otherwise motion detection would not be possible), the following auto-regressive model was used:

$I(m,n) = \frac{1}{3}[I(m,n-1)+I(m-1,n)+I(m-1,n-1)]+ n_i(m,n)$,

where i=1,2. For the background (i=1), n_1 is a Gaussian random variable with mean μ_1 = 50 and variance σ_1^2 = 49. The rectangle (i=2) was generated with μ_2 = 100 and variance σ_2^2= 25. All pixels from the background move to the right, and the displacement from frame 1 to frame 2 is $\boldsymbol{d}_b(\boldsymbol{r})= (d_{bx}(\boldsymbol{r}),d_{by}(\boldsymbol{r}))=(2,0)$. The rectangle moves diagonally from frame 1 to 2 with $\boldsymbol{d}_r(\boldsymbol{r})=(d_{rx}(\boldsymbol{r}),d_{ry}(\boldsymbol{r}))=(1,2)$.

Table 1 shows the values for the MSE, bias, $\overline{IMC}$ (dB) and $\overline{DFD}^2$ for the estimated image brightness for frames 1 and 2 of the noiseless

"Synthetic" sequence using the following algorithms: Wiener filter, EM with one mask (EM), and EM multi-mask (EMm). All implementations using the EM technique performed better than the Wiener filter. Table 2 shows the values of the same metrics when *SNR = 20dB*.

The motion-compensated frames corresponding to the estimated DVF's for the noisy case (*SNR=20dB*) are presented in Figure 2. Visually speaking, there is an improvement in the outcome around the borders of the rectangle for the multi-mask implementation of the EMm.

6.2 Experiment 2.

Figure 3 presents the values of the $\overline{IMC}$ for frames 31 to 40 of the "Mother and Daughter" sequence for the noiseless and noisy (*SNR=20dB*) cases, respectively, for the multi-mask algorithm EMA=EMm. If we look at the noiseless case, we will see that the algorithm EMA=EMm has very good performance. For the noisy case EMA=EMm algorithm can perform extremely well most of the time, except for two pairs of frames. Their qualitative (visual) performance can be observed in Figure 4. The superior performance of the EMA=EMm routine can be better noticed by looking at the errors resulting from motion compensation.

6.3 Experiment 3

Figure 5 presents the values of the $\overline{IMC}$ for frames 11 to 20 of the "Foreman" sequence for the noiseless and noisy (*SNR=20dB*) cases, respectively, for the multi-mask algorithm EMm. The EM implementations outperform the Wiener filter all the time. Their qualitative performance can be observed in Figure 6 which shows the error in motion compensation. Visually speaking, we perceive that the EM seems to be more robust to the motion introduced by the camera, because the background of the frame showing the errors due to motion-compensation has less artifacts than the one generated with the Wiener filter.

7. CONCLUSIONS

For all the sequences, the EM algorithm performed better than the Wiener filter for both the one mask and multi-mask cases, regardless of the presence of noise.

The EM algorithm showed some sensitivity to the choice of initial estimates. We used more than one initial parameter set $\boldsymbol{\Phi}^0 = [\sigma_n^{2(0)}, \sigma_1^{2(0)}, \sigma_2^{2(0)}]^T$ to improve the rate of convergence , but even with this extra feature, the resulting algorithms using one mask and multiple masks where faster than the corresponding Wiener filter counterparts.

For the EM method, we have a simple algorithm that is guaranteed to converge and it does not equire numerical optimization. Given that there are multiple iterations at every pixel location, the speed advantage gained by means of the EM algorithm is considerable.

The authors think this framework has great potential for applications such as video coding and image segmentation.

8. REFERENCES

[Bie87] Biemond, J.; Looijenga, L.; Boekee, D.E. and Plompen, R. H. J. M. A Pel-Recursive Wiener-Based Displacement Estimation Algorithm, Signal Processing, 13, pp. 399-412, December 1987.

[Brailean95] Brailean, J.C. and Katsaggelos, A. K. Simultaneous Recursive Displacement Estimation and Restoration of Noisy-Blurred Image Sequences, IEEE Trans. Im. Proc., Vol. 4, No. 9, pp. 1236-1268, September 1995.

[Estrela03] Estrela, V.; Rivera, L.A.; Beggio, P.C. and Lopes, R.T. Regularized Pel-Recursive Motion Estimation Using Generalized Cross-Validation and Spatial Adaptation, Proc. of SIBGRAPI2003 XVI Brazilian Symposium Computer Graphic and Image Processing, pp. 331-338, Oct. 2003.

[Dem77] Dempster, A.P.; Laird, N.M. and Rubbin, D.B. Maximum Likelihood From Incomplete Data via the EM Algorithm, Ann. Roy. Statist. Soc., Vol. 39, pp. 1-38, Dec. 1977.

[Fan96] Fan, C.M.; Namazi, N.M. and Penafiel, P.B. A New Image Motion Estimation Algorithm Based on the EM Technique, IEEE Trans. on P.A.M.I., Vol. 18, No. 3, pp. 348-352, 1996.

[Jain89] Jain, A.K. Fundamentals of Digital Image Processing, Prentice-Hall, New Jersey, 1989.

[Kat91] Katsaggelos, A.K. Image Restoration, Springer-Verlag, Berlin, Heidelberg, 1991.

[Kay93] Kay, S. Fundamentals of Statistical Signal Processing: Estimation Theory, Prentice Hall, 1993.

[Tek95] Tekalp, M., Digital Video Processing, Prentice Hall, 1995.

	Wiener	EM	EMm
MSE_x	0.1548	0.1436	0.1311
MSE_y	0.0740	0.0681	0.0572
$bias_x$	0.0610	0.0571	0.0527
$bias_y$	-0.0294	-0.0189	-0.0180
$\overline{IMC}$ (dB)	19.46	19.93	20.47
$\overline{DFD}^2$	4.16	4.08	3.26

Table 1. Comparison between EM implementations and the Wiener filter. *SNR* = ∞.

	Wiener	EM	EMm
MSE_x	0.2563	0.2401	0.2323
MSE_y	0.1273	0.1249	0.1226
$bias_x$	0.0908	0.0842	0.0822
$bias_y$	-0.0560	-0.0558	-0.0503
$\overline{IMC}(dB)$	14.74	15.09	15.45
$\overline{DFD}^2$	12.24	10.91	10.03

Table 2. Comparison between EM implementations and the Wiener filter. *SNR =20dB*.

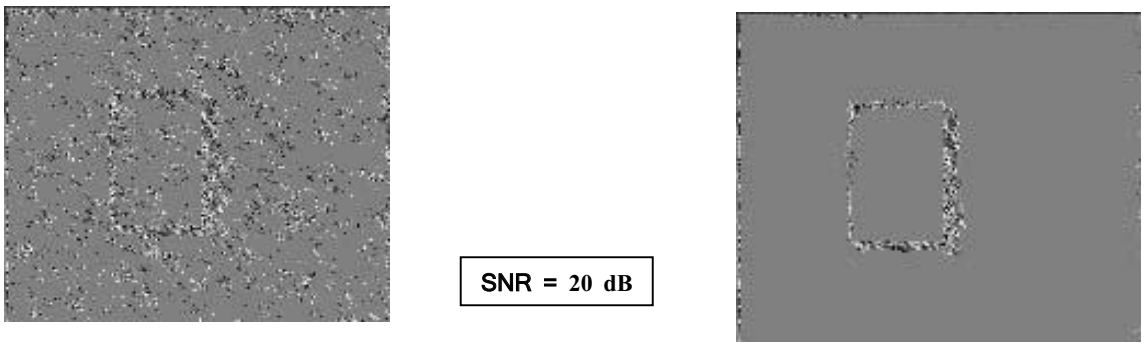

Figure 2. Motion-compensation errors for the Wiener (left) and the EMm (right) algorithms.

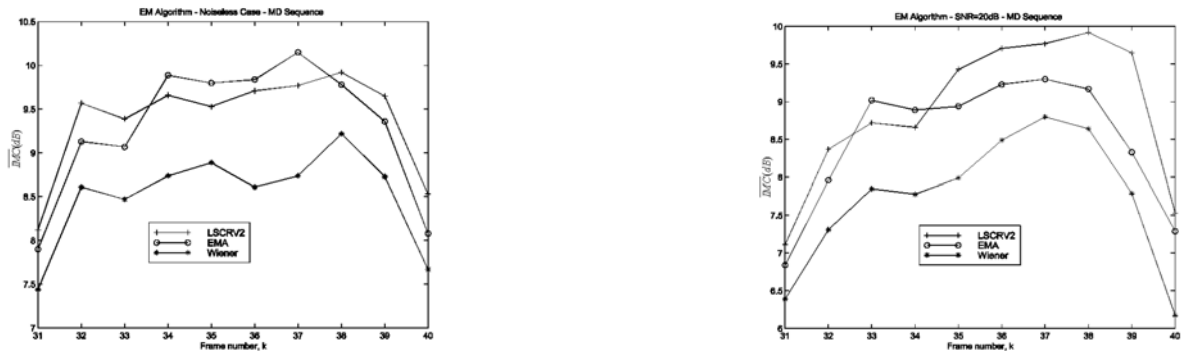

Figure 3. $\overline{IMC(dB)}$ for the noiseless (left) and noisy (right) cases for the "Mother and Daughter" sequence.

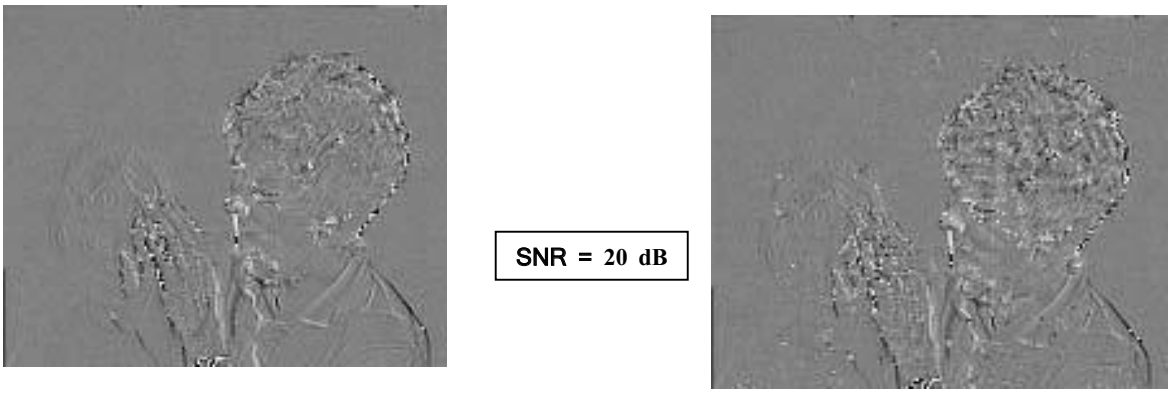

Figure 4. Motion-compensated errors for frame 32 of the "Mother and Daughter" sequence: the Wiener filter (right) and the EMm (left) algorithms.

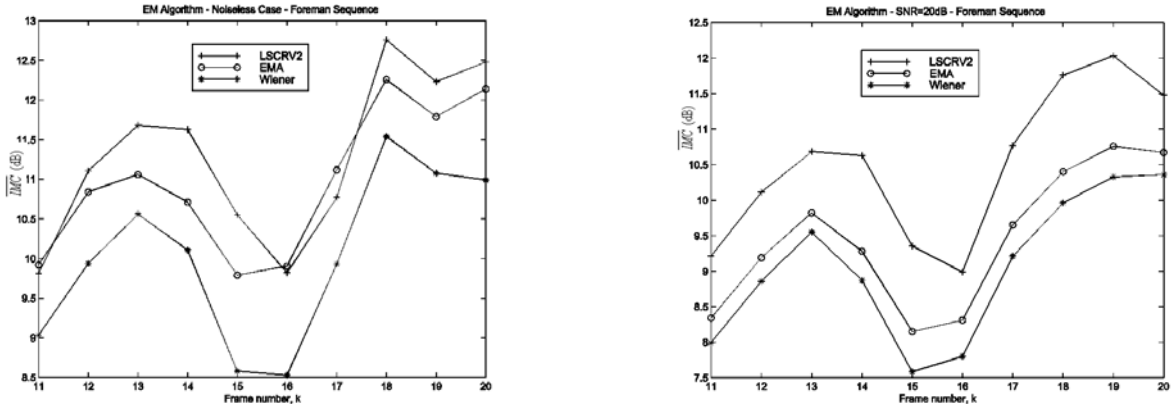

Figure 5. $\overline{IMC}(dB)$ **for frames 11-20 of the noiseless (left) and noisy cases with SNR=20dB (right) for the "Foreman" sequence.**

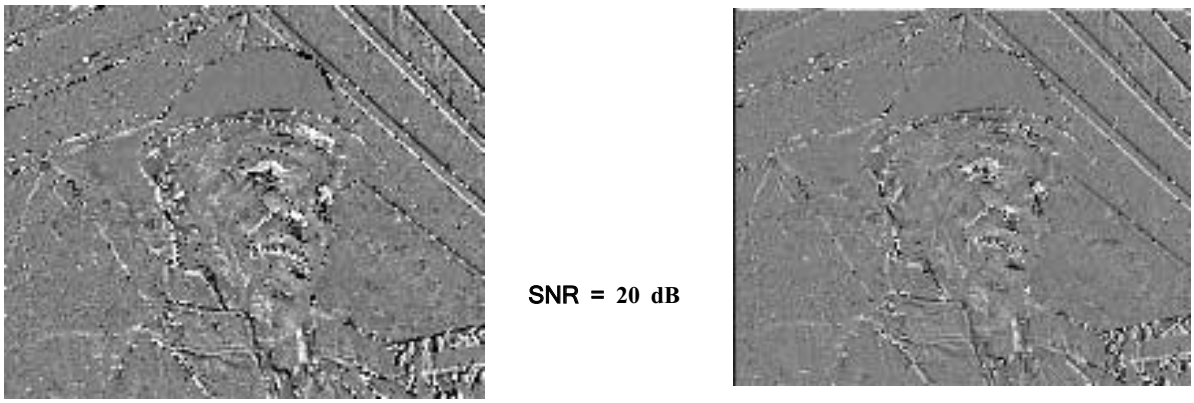

Figure 6. Motion-compensated errors for frame 16 of the "Foreman" sequence: the Wiener filter (left) and the EMm (right) algorithms .