J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*
Copyright © 1999 John Wiley & Sons, Inc.

IMAGE SEQUENCES

This article covers the fundamentals of image sequence (also referred to as moving picture or digital video) processing, including motion estimation and compensation, motion segmentation, visual object tracking, sampling structure conversion (standards conversion), and video filtering. These subjects are essential to the understanding of digital video systems, which have enabled the convergence of computing, telecommunications, broadcasting and entertainment technologies.

Basics of Motion

Time-varying images are 2-D projections of the 3-D world (scene) at successive time instants. The projection can be modeled by a perspective or orthographic mapping depending on the camera model and imaging conditions (1). For analog video, each image is further sampled in the vertical direction, resulting in a one-dimensional continuous-time video signal representing concatanation of image lines. For digital video, both horizontal and vertical dimensions are sampled resulting in pixelization of the time-varying image.

Projected Motion versus Apparent Motion. Variation in the intensity patterns of successive images are generally due to relative motion between the scene and camera, including motion of scene objects and/or camera motion such as zoom, pan, and tilt. Other factors that affect image intensity patterns are sudden or gradual changes in scene illumination, changes in reflective properties of objects, and imaging noise. Relative motion between the scene and camera can best be modeled in 3-D, resulting in a 3-D motion description. Projected motion refers to projection of this 3-D motion onto the 2-D image plane using the camera model. The concept of projected motion is illustrated in Fig. 1.

However, the projected motion is not observable. Instead, what we observe is the so-called “apparent” motion, which refers to image plane motion that can be deduced from observable differences in successive image intensity patterns. The apparent motion is also referred to as *optical flow.* Clearly, the apparent motion may differ from the projected motion; that is, all projected motion does not generate optical flow, and all optical flow does not correspond to projected motion. Two examples are shown in Fig. 2. In what follows, motion estimation refers to apparent motion estimation.

Motion vectors may be defined in the forward or backward direction. Given the intensity samples at frames k and $k + l$, which are related by

$$s_k(x_1, x_2) = s_{k+\ell}(x_1 + d_1(\mathbf{x}), x_2 + d_2(\mathbf{x})) \qquad (1)$$

computation of the real-valued correspondence vector field $\mathbf{d}(\mathbf{x}) = [d_1(\mathbf{x})\, d_2(\mathbf{x})]^T$, where the temporal arguments of $\mathbf{d}(\mathbf{x})$ are dropped, is known as *forward motion estimation.* If we define the motion vectors between frames k and $k - l$, then the 2-D motion model can be stated as

$$s_k(x_1, x_2) = s_{k-\ell}(x_1 + d_1(\mathbf{x}), x_2 + d_2(\mathbf{x}))$$

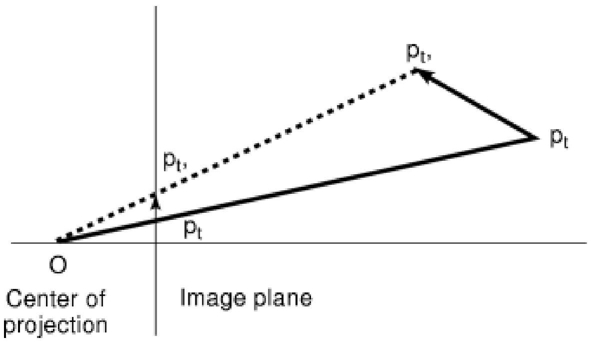

Fig. 1. The projected motion. Suppose a point P_t in 3-D moves to $P_{t'}$. The projection of the resulting 3-D motion vector into the image plane is depicted. Notice that all 3-D motion vectors whose tip lie on the dotted line project into the same 2-D motion vector.

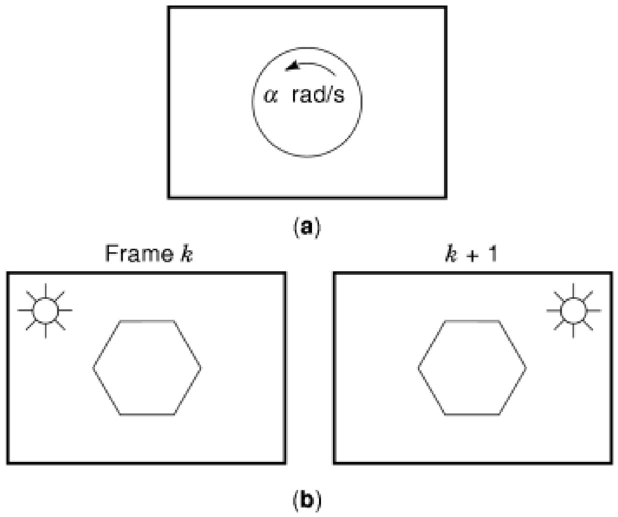

Fig. 2. (a) A uniform rotating disc does not give rise to any intensity variations in time; (b) Changing illumination direction results in intensity variations in the absence of any physical motion.

or

$$s_k(x_1, x_2) = s_{k-\ell}(x_1 - d_1(\mathbf{x}), x_2 - d_2(\mathbf{x}))$$

Computation of the motion field in this case is known as *backward motion estimation.*

However, even estimation of apparent motion is an ill-posed problem in the sense that it may not have a solution (occlusion problem) or the solution may not be unique (aperture problem), as discussed below.

Occlusion Problem. Occlusion refers to covering and uncovering of an image region by either another object (object-to-object occlusion) or the motion of the object itself such as out-of-plane rotations (self-occlusion). The covered and uncovered region concepts are illustrated in Fig. 3. For those pixels in frame k which are in the background-to-be-covered, no corresponding pixel can be found in frame $k + 1$. Similarly, for those pixels in frame $k + 1$ which are in the uncovered background, there is no corresponding pixel in frame k.

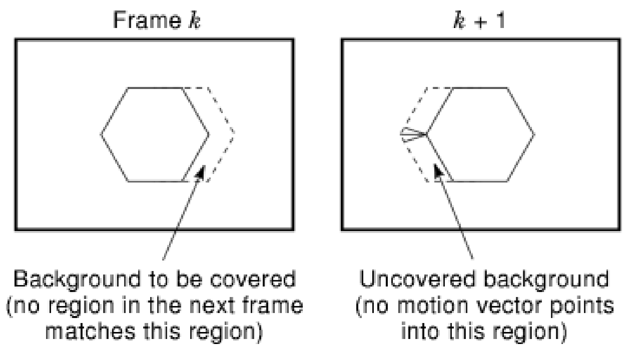
Frame k
$k + 1$
Background to be covered (no region in the next frame matches this region)
Uncovered background (no motion vector points into this region)

Fig. 3. The covered/uncovered region problem. The object indicated by solid lines translates in the x_1 direction between frames k and $k + 1$. The dotted region in frame k indicates the background to be covered in frame $k + 1$. The dotted region in frame $k + 1$ indicates the background uncovered by the motion of the object.

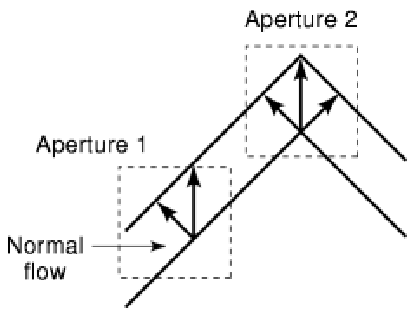

Fig. 4. The aperture problem. Suppose we have a corner moving in the x_2 direction (upward). If we estimate this motion based on a local window, indicated by Aperture 1, then it is not possible to determine whether it moves upward or perpendicular to the edge. The motion in the direction perpendicular to the edge is called the normal flow. However, if we observe Aperture 2, then it is possible to estimate the correct motion, since the image has gradient in two perpendicular directions within this aperture.

Aperture Problem. The aperture problem is a restatement of the fact that the solution to 2-D motion estimation problem is not unique. If motion vectors at each pixel are considered as independent variables, then there are twice as many unknowns as there are equations given by Eq. (1). The number of equations is equal to the number of pixels in the image, but for each pixel the motion vector has two components. Theoretical analysis indicates that we can only determine motion that is orthogonal to the spatial image gradient, called the *normal flow,* at each pixel (1). The aperture problem is illustrated in Fig. 4. It follows that it is possible to overcome the aperture problem by estimating the motion based on a block of pixels that contain sufficient gray-level variation. Of course, implicit here is the assumption that all these pixels translate by the same motion vector.

Motion Models. The aperture and occlusion problems in 2-D motion estimation can be overcome by employing global motion models. Motion models can be classified as nonparametric and parametric ones.

Nonparametric Models. Nonparametric models impose spatio-temporal smoothness constraints on the estimated motion field. That is, we seek the smoothest solution out of all feasible solutions. The use of smoothness constraints to define regularized solutions to ill-proposed problems is also well known in other areas of science and engineering (2). The nonparametric constraints can be classified as deterministic versus stochastic smoothness models. Deterministic models usually require solution of variational problems (3) whereas stochastic models lead to Bayesian estimation problems (4). Unlike parametric models, nonparametric models are suitable for estimation of deformable motion fields.

Parametric Models. Parametric models aim to describe the orthographic or perspective projection of 3-D rigid motion (displacement or velocity) of a surface into the image plane. In general, parametric 2-D motion models depend on a representation of the 3-D surface. For example, a 2-D motion field resulting from 3-D rigid motion of a planar surface under orthographic projection can be described by a 6-parameter affine model, while under perspective projection it can be described by an 8-parameter nonlinear model (5). There also exist more complicated models for quadratic surfaces (6).

The simplest parametric motion model is the translational block model, which can be characterized by a single translation vector

$$\begin{aligned} x_1' &= x_1 + d_1 \\ x_2' &= x_2 + d_2 \end{aligned} \tag{2}$$

where (x'_1, x'_2) denotes the coordinates of a point in the frame $k + l$.

The affine motion model, given by

$$\begin{aligned} x_1' &= a_1x_1 + a_2x_2 + a_3 \\ x_2' &= a_4x_1 + a_5x_2 + a_6 \end{aligned} \tag{3}$$

has six free parameters, and represents the orthographic projection of a rigid motion of a planar object in 3-D.

The perspective and bilinear motion models, given by

$$\begin{aligned} x_1' &= \frac{a_1x_1 + a_2x_2 + a_3}{a_7x_1 + a_8x_2 + 1} \\ x_2' &= \frac{a_4x_1 + a_5x_2 + a_6}{a_7x_1 + a_8x_2 + 1} \end{aligned} \tag{4}$$

and

$$\begin{aligned} x_1' &= a_1x_1 + a_2x_2 + a_3x_1x_2 + a_4 \\ x_2' &= a_5x_1 + a_6x_2 + a_7x_1x_2 + a_8 \end{aligned} \tag{5}$$

respectively, have 8 free parameters. The perspective model represents the perspective projection of a rigid motion of a planar object in 3-D.

Motion Estimation and Compensation

Motion estimation methods can be broadly classified into differential methods which require estimation of spatial and temporal image intensity gradients, numerical optimization methods including pel-recursive and Bayesian methods, search-based methods including block-matching and its variations, and transform-domain methods. The essentials of these methods are summarized in the following.

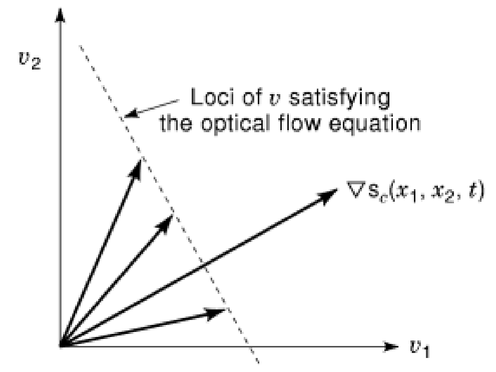
v_2
Loci of v satisfying the optical flow equation
$\nabla s_c(x_1, x_2, t)$
v_1

Fig. 5. The normal flow. All vectors whose tips lie on the dotted line satisfy Eq. (8).

Optical Flow Equation and Differential Methods. Let $s_c(x_1, x_2, t)$ denote a continuous space-time intensity distribution. If the intensity remains constant along a motion trajectory, we have

$$\frac{ds_c(x_1, x_2, t)}{dt} = 0 \tag{6}$$

where x_1 and x_2 vary by t according to the motion trajectory. Equation (6) is a total derivative expression and denotes the rate of change of intensity along the motion trajectory. Using the chain rule of differentiation, it can be expressed as

$$\frac{\partial s_c(\mathbf{x}; t)}{\partial x_1} v_1(\mathbf{x}, t) + \frac{\partial s_c(\mathbf{x}; t)}{\partial x_2} v_2(\mathbf{x}, t) + \frac{\partial s_c(\mathbf{x}; t)}{\partial t} = 0 \tag{7}$$

where $v_1(\mathbf{x}, t) = dx_1/dt$ and $v_2(\mathbf{x}, t) = dx_2/dt$ denote components of the coordinate velocity vector in terms of the continuous spatial coordinates. The expression Eq. (7) is known as the optical flow equation (*OFE*) or optical flow constraint, which can alternatively be expressed as

$$\langle \nabla s_c(\mathbf{x}, t), \mathbf{v}(\mathbf{x}, t) \rangle + \frac{\partial s_c(\mathbf{x}; t)}{\partial t} = 0 \tag{8}$$

where $\nabla s_c(\mathbf{x}; t) \doteq [\partial s_c(\mathbf{x}; t)/\partial x_1 \cdot \partial s_c(\mathbf{x}; t)/\partial x_2]^T$ and $\langle \cdot, \cdot \rangle$ denotes vector inner product.

The OFE yields one scalar equation in two unknowns, $v_1(\mathbf{x}, t)$ and $v_2(\mathbf{x}, t)$, at each site $(\mathbf{x}, t)$. Inspection of Eq. (8) reveals that we can only estimate the component of the flow vector that is in the direction of the spatial image gradient $\nabla s_c(\mathbf{x}; t)/\|\nabla s_c(\mathbf{x}; t)\|$, called the normal flow $v_\perp(\mathbf{x}, t)$, because the component that is orthogonal to the spatial image gradient disappears under the dot product. This is illustrated in Fig. 5. The normal flow at each pixel can be computed from Eq. (8) as

$$v_\perp(\mathbf{x}, t) = \frac{-\dfrac{\partial s_c(\mathbf{x}; t)}{\partial t}}{\|\nabla s_c(\mathbf{x}; t)\|} \tag{9}$$

The OFE Eq. (7) imposes a constraint on the component of the flow vector that is in the direction of the spatial gradient of the image intensity at each site (pixel), which is consistent with the aperture problem. Observe that the OFE approach requires that first, the spatiotemporal image intensity be differentiable, and second, the partial derivatives of the intensity be available. In the following we present two approaches to estimate optical flow from estimates of normal flow.

Lucas–Kanade Method (7). A simple scheme to overcome the aperture problem is to assume that the motion vector remains unchanged over a particular block of pixels, denoted by B; that is,

$$\mathbf{v}(\mathbf{x},t) = \mathbf{v}(t) = [v_1(t)\ v_2(t)]^T, \qquad \text{for } \mathbf{x} \in \mathscr{B} \qquad (10)$$

Although such a model cannot accurately handle rotation and/or zoom, it is possible to estimate a purely translational motion vector under this model provided that the block of pixels contain sufficient gray-level variation.

Define the error in the optical flow equation over the block of pixels B as

$$E = \sum_{\mathbf{x}\in\mathscr{B}} \left(\frac{\partial s_c(\mathbf{x}:t)}{\partial x_1} v_1(t) + \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} v_2(t) + \frac{\partial s_c(\mathbf{x},t)}{\partial t} \right)^2 \qquad (11)$$

Computing the partials of the error E with respect to $v_1(t)$ and $v_2(t)$, respectively, and setting them equal to zero, the result is

$$\sum_{\mathbf{x}\in\mathscr{B}} \left(\frac{\partial s_c(\mathbf{x},t)}{\partial x_1} \hat{v}_1(t) + \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} \hat{v}_2(t) + \frac{\partial s_c(\mathbf{x},t)}{\partial t} \right) \frac{\partial s_c(\mathbf{x},t)}{\partial x_1} = 0$$
$$\sum_{\mathbf{x}\in\mathscr{B}} \left(\frac{\partial s_c(\mathbf{x},t)}{\partial x_1} \hat{v}_1(t) + \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} \hat{v}_2(t) + \frac{\partial s_c(\mathbf{x},t)}{\partial t} \right) \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} = 0$$

where $\hat{}$ denotes the estimate of the respective quantity. Solving these equations simultaneously, the result is

$$\begin{bmatrix} \hat{v}_1(t) \\ \hat{v}_2(t) \end{bmatrix} = \begin{bmatrix} \sum_{\mathbf{x}\in\mathscr{B}} \frac{\partial s_c(\mathbf{x},t)}{\partial x_1} \frac{\partial s_c(\mathbf{x},t)}{\partial x_1} & \sum_{\mathbf{x}\in\mathscr{B}} \frac{\partial s_c(\mathbf{x},t)}{\partial x_1} \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} \\ \sum_{\mathbf{x}\in\mathscr{B}} \frac{\partial s_c(\mathbf{x},t)}{\partial x_1} \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} & \sum_{\mathbf{x}\in\mathscr{B}} \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} \end{bmatrix}^{-1} \begin{bmatrix} -\sum_{\mathbf{x}\in\mathscr{B}} \frac{\partial s_c(\mathbf{x},t)}{\partial x_1} \frac{\partial s_c(\mathbf{x},t)}{\partial t} \\ -\sum_{\mathbf{x}\in\mathscr{B}} \frac{\partial s_c(\mathbf{x},t)}{\partial x_2} \frac{\partial s_c(\mathbf{x},t)}{\partial t} \end{bmatrix} \qquad (12)$$

It is possible to increase the influence of the constraints toward the center of the block B by employing weighted summations. A suitable weighting function may be in the form of a 2-D triangular window. Clearly, the accuracy of the flow estimates depends on the accuracy of the estimated spatial and temporal partial derivatives.

Horn–Schunck Method (3). Horn and Schunck seek a motion field that satisfies the OFE with the minimum pixel-to-pixel variation among the flow vectors, in order to impose a less restrictive global smoothness constraint on the velocity field. Let

$$\mathscr{E}_{of}(\mathbf{v}(\mathbf{x},t)) = \langle \nabla s_c(\mathbf{x}:t),\ \mathbf{v}(\mathbf{x},t)\rangle + \frac{\partial s_c(\mathbf{x}:t)}{\partial t} \tag{13}$$

denote the error in the optical flow equation. Observe that the OFE is satisfied when $E_{of}(\mathbf{v}(\mathbf{x}, t))$ is equal to zero. In the presence of occlusion and noise, we aim to minimize the square of $E_{of}(\mathbf{v}(\mathbf{x}, t))$, in order to enforce the optical flow constraint. The pixel-to-pixel variation of the velocity vectors can be quantified by the sum of the magnitude squares of the spatial gradients of the components of the velocity vector, given by

$$\begin{aligned}\mathscr{E}_s^2(\mathbf{v}(\mathbf{x},t)) &= \|\nabla v_1(\mathbf{x},t)\|^2 + \|\nabla v_2(\mathbf{x},t)\|^2 \\ &= \left(\frac{\partial v_1}{\partial x_1}\right)^2 + \left(\frac{\partial v_1}{\partial x_2}\right)^2 + \left(\frac{\partial v_2}{\partial x_1}\right)^2 + \left(\frac{\partial v_2}{\partial x_2}\right)^2\end{aligned} \tag{14}$$

where we assume that the spatial and temporal coordinates are continuous variables. It can easily be verified that the smoother the velocity field, the smaller $E^2{}_s(\mathbf{v}(\mathbf{x}, t))$.

Then the Horn and Schunck method minimizes a weighted sum of the error in the OFE and a measure of the pixel-to-pixel variation of the velocity field

$$\min_{\mathbf{v}(\mathbf{x},t)} \int_{\mathscr{A}} (\mathscr{E}_{of}^2(\mathbf{v}) + \alpha^2 \mathscr{E}_s^2(\mathbf{v}))\, d\mathbf{x} \tag{15}$$

to estimate the velocity vector at each point $\mathbf{x}$, where A denotes the continuous image support. The parameter α^2, usually selected heuristically, controls the strength of the smoothness constraint. Larger values of α^2 increase the influence of the constraint.

The minimization of the functional Eq. (15), using the calculus of variations, results in the Gauss–Seidel iteration

$$\begin{aligned}\hat{v}_1^{(n+1)}(\mathbf{x},t) &= \bar{v}_1^{(n)}(\mathbf{x},t) \\ &\quad - \frac{\partial s_c}{\partial x_1}\,\frac{\dfrac{\partial s_c}{\partial x_1}\bar{v}_1^{(n)}(\mathbf{x},t) + \dfrac{\partial s_c}{\partial x_2}\bar{v}_2^{(n)}(\mathbf{x},t) + \dfrac{\partial s_c}{\partial t}}{\alpha^2 + \left(\dfrac{\partial s_c}{\partial x_1}\right)^2 + \left(\dfrac{\partial s_c}{\partial x_2}\right)^2} \\ \hat{v}_2^{(n+1)}(\mathbf{x},t) &= \bar{v}_2^{(n)}(\mathbf{x},t) \\ &\quad - \frac{\partial s_c}{\partial x_2}\,\frac{\dfrac{\partial s_c}{\partial x_1}\bar{v}_1^{(n)}(\mathbf{x},t) + \dfrac{\partial s_c}{\partial x_2}\bar{v}_2^{(n)}(\mathbf{x},t) + \dfrac{\partial s_c}{\partial t}}{\alpha^2 + \left(\dfrac{\partial s_c}{\partial x_1}\right)^2 + \left(\dfrac{\partial s_c}{\partial x_2}\right)^2}\end{aligned} \tag{16}$$

where n is the iteration counter, the overbar denotes weighted local averaging (excluding the present pixel), and all partials are elevated at the point $(\mathbf{x}, t)$. The reader is referred to Ref. 3 for the derivation of this iterative estimator. The initial estimates of the velocities $v^{(0)}{}_1(\mathbf{x}, t)$ and $v^{(0)}{}_2(\mathbf{x}, t)$ are usually taken as zero. Of course, all spatial and temporal image gradients need to be estimated from the observed image samples (1).

Pel-Recursive Methods. Pel-recursive methods are predictor-corrector type displacement estimators computed sequentially at each pixel. The prediction can be taken as the value of the motion estimate at the previous pixel location or as a linear combination of previous motion estimates in a neighborhood of the current pixel. The update is computed by gradient-based minimization of a positive-definite function E of the displaced frame difference (*DFD*) at each pixel. The DFD is defined as

$$dfd(\mathbf{x}, \mathbf{d}) = s_c(\mathbf{x} + \mathbf{d}(\mathbf{x}), t + \Delta t) - s_c(\mathbf{x}, \mathbf{t})$$

where $\mathbf{d}(\mathbf{x})$ denotes the displacement vector at the spatial location $\mathbf{x}$. The positive definiteness of E ensures that the minimum occurs when the DFD is zero. Pel-recursive motion estimation is usually preceded by a change detection stage, where the frame difference at each pixel is tested against a threshold. Estimation is performed only at those pixels belonging to the changed region.

An early pel-recursive estimator is the Netravali-Robbins algorithm which minimizes the square of the DFD at each pixel, using a gradient descent method (8). Then, the criterion function E to be minimized is given by

$$E(\mathbf{x}; \mathbf{d}) = [dfd(\mathbf{x}, \mathbf{d})]^2 \qquad (17)$$

where *dfd* denotes the displaced frame difference. Minimization of $E(\mathbf{x}; \mathbf{d})$ with respect to $\mathbf{d}$, at pixel $\mathbf{x}$, by the steepest descent method yields the iteration

$$\hat{\mathbf{d}}^{i+1}(\mathbf{x}) = \hat{\mathbf{d}}^{i}(\mathbf{x}) - \epsilon\, dfd(\mathbf{x}, \hat{\mathbf{d}}^{i}) \nabla_{\mathbf{x}} s_c(\mathbf{x} - \hat{\mathbf{d}}^{i} : t - \Delta t) \qquad (18)$$

where $\nabla_{\mathbf{x}}$ is the gradient with respect to $\mathbf{x}$, and ϵ is the step size. Note that negative of the gradient points to the direction of the steepest descent. In Eq. (18), the first and second terms are the prediction and update terms, respectively. The aperture problem is also apparent in the pel-recursive algorithms. Since the update term is a vector along the spatial gradient of the image intensity, no correction can be performed in the direction perpendicular to the gradient vector.

The convergence and the rate of convergence of the Netravali-Robbins algorithm depend on the choice of the step size parameter ϵ. For example, if $\epsilon = 1/16$, then at least 32 iterations are required to estimate a displacement by 2 pixels. On the other hand, a large choice for the step size may cause an oscillatory behavior. Several strategies can be advanced to facilitate faster convergence of the algorithm (9,10). For example, Caffario and Rocca (10) developed an adaptive step-size expression

$$\epsilon = \frac{1}{\|\nabla_{\mathbf{x}} s_c(\mathbf{x} - \hat{\mathbf{d}}^{i} : t - \Delta t)\|^2 + \eta^2} \qquad (19)$$

which includes a bias term η^2 to avoid division by zero in areas of constant intensity where the spatial gradient is almost zero. A typical value for $\eta^2 = 100$. In addition, Walker and Rao (9) have introduced the following heuristic rules: (1) If the DFD is less than a threshold, the update term is set equal to zero. (2) If the DFD exceeds the threshold, but the magnitude of the spatial image gradient is zero, then the update term is again set equal to zero. (3) If the absolute value of the update term (for each component) is less than $\frac{1}{16}$, then it is set equal to $\pm\,\frac{1}{16}$. (4) If the absolute value of the update term (for each component) is more than 2, then it is set equal to ± 2. Experimental results indicate that using an adaptive step size greatly improves the convergence of the algorithm.

Extension of the pel-recursive approach to block-based estimation results in the so-called Wiener-type motion estimation strategies (1).

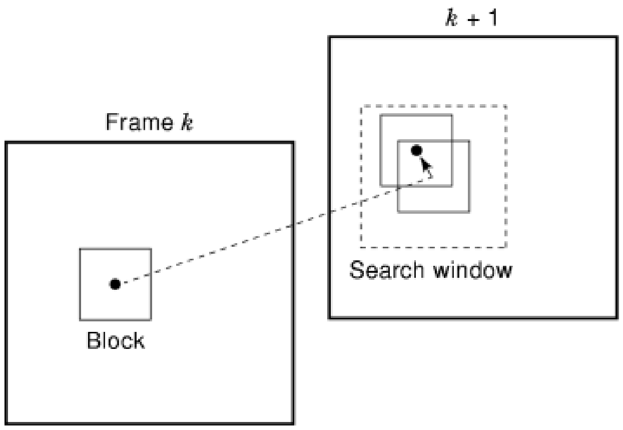

Fig. 6. Block matching.

Bayesian Methods. Bayesian methods utilize probabilistic smoothness constraints, usually in the form of a Markov–Gibbs random field, to model the local interaction of displacement vectors. A maximum a posteriori probability estimate of the displacement field can then be computed by minimizing a cost function, which consists of an optical flow error term and a Gibbs potential term (penalizing discontinuities in the estimated motion field) (4). This global minimization problem can then be solved by a simulated annealing procedure or a deterministic approximation of it (1). The main drawback of Bayesian methods is that simulated annealing procedures require an extensive amount of computation, whereas deterministic procedures may be caught in a local minimum of the cost function.

Block-Matching. Block-matching is perhaps the most commonly used motion estimation technique since it fits well with block-based video compression standards such as ISO/IEC MPEG-1/2 and ITU-T H.261/263. Block matching takes a fixed size block from the present frame, and searches for the location of the best-matching block of the same size in the reference frame (see Fig. 6). Block-matching algorithms differ in the choice of the matching criteria and search strategy employed.

The matching of the blocks can be quantified according to various criteria including maximum cross-correlation (similar to the phase-correlation method (1)), minimum mean square error (*MSE*), minimum mean absolute difference (*MAD*), maximum matching pel count (*MPC*), and so on. For example, the minimum MSE criterion is defined by

$$MSE(d_1, d_2) = \frac{1}{N_1N_2} \sum_{(n_1,n_2)\in\mathcal{B}} [s(n_1, n_2, k) - s(n_1 + d_1, n_2 + d_2, k + 1)]^2 \qquad (20)$$

where B is an $N_1 \times N_2$ block, and (d_1, d_2) denotes a candidate motion vector. The estimate of the motion vector is the value of (d_1, d_2) which minimizes Eq. (20). Alternatively, the minimum MAD criterion, defined as

$$MAD(d_1, d_2) = \frac{1}{N_1N_2} \sum_{(n_1,n_2)\in\mathcal{B}} |s(n_1, n_2, k) - s(n_1 + d_1, n_2 + d_2, k + 1)| \qquad (21)$$

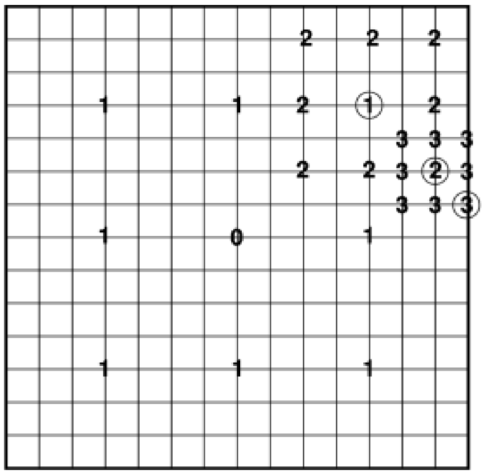

Fig. 7. Three-step search. Only pixels marked "1" are searched in the first step. The pixel providing the best match is circled. The second step is now centered about this pixel and the step size is halved.

is the most popular choice for very large scale integration (*VLSI*) implementations. Then the displacement estimate is given by

$$[\hat{d}_1 \hat{d}_2]^T = \arg \min_{(d_1, d_2)} MAD(d_1, d_2) \qquad (22)$$

The performance of the MAD criterion deteriorates as the search area becomes larger due to the presence of several local minima.

Finding the best-matching block requires computation of the matching criterion for all candidate displacement vectors (d_1, d_2) at each pixel (n_1, n_2). This procedure, known as *full search,* is extremely time-consuming. As a first measure to reduce the computational burden, we limit the candidate vectors to within a "search window"

$$-M_1 \leq d_1 \leq M_1 \qquad \text{and} \qquad -M_2 \leq d_2 \leq M_2$$

which is centered about each pixel for which a motion vector will be estimated, where M_1 and M_2 are predetermined integers. A search window is shown in Fig. 6. Another commonly employed practice to lower the computational burden is to estimate motion vectors on a sparse grid of pixels, e.g., once every eight pixels and eight lines using a 16×16 block, and then interpolate the motion field to estimate the remaining vectors.

In most cases, however, search strategies faster than the full search are utilized, although they lead to suboptimal solutions. Some examples of faster search algorithms include the three-step search and cross-search. These faster search algorithms evaluate the criterion function only at a predetermined subset of the candidate motion vectors. The three-step search is illustrated in Fig. 7. Note that the expected accuracy of motion estimates varies according to the application. In motion-compensated compression, all we seek is a matching block, even if the match does not correlate well with the actual projected motion. It is for this reason that faster search algorithms serve video compression applications reasonably well.

Generalized Block-Matching. Block-matching method is based on the motion model of translational blocks. The spatial transformations [Eqs. (3), (4), and (5)] provide superior motion tracking and rendering, especially in the presence of rotation and zoom, compared to the translational model [Eq. (2)]. Thus, it is of

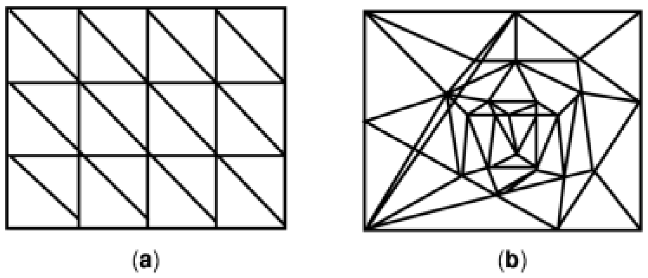

Fig. 8. Mesh connectivity constraints.

interest to extend the concept of block-matching to estimate the parameters of these more sophisticated motion models. This, of course, leads to increased computational complexity; we now have to perform search in a 6- or 8-D parameter space (parameters of affine or perspective/bilinear transformation) instead of a 2-D space (components of the translation vector). Several generalized motion estimation schemes have been proposed, including a full-search method (11) and fast search method (12).

The full-search method can be summarized as follows: (1) Segment the current frame into rectangular blocks. (2) Perturb the coordinates of the corners of a matching quadrilateral in the search frame starting from an initial guess. (3) For each quadrilateral, find the parameters of a prespecified spatial transformation that maps this quadrilateral onto the rectangular block in the current frame using the coordinates of the four matching corners. (4) Find the coordinates of each corresponding pixel within the quadrilateral using the computed spatial transformation, and calculate the MSE between the given block and the matching patch. (5) Choose the spatial transformation that yields the smallest MSE or MAD. In order to reduce the computational burden imposed by generalized block-matching, it is only used for those blocks where standard block-matching is not satisfactory. The displaced frame difference resulting from standard block matching can be used as a decision criterion.

Mesh-Based Motion Estimation. A 2-D mesh refers to a tesellation of the plane into polygonal (commonly rectangular or triangular) patches. Each patch is motion compensated by means of an appropriate spatial transformation such as Eqs. (3), (4), or (5). The parameters of the mapping for a patch are computed from the motion vectors at the vertices of the respective patch. The main difference between generalized block matching and mesh-based motion estimation is that the mesh elements are not allowed to overlap with each other in the reference frame. That is, mesh connectivity constraints must be observed in motion estimation of vertex points. This is illustrated in Fig. 8. Methods for motion estimation at the vertex points include block-matching (13) and hexagonal matching (14).

Transform-Domain Motion Estimation. Transform domain motion estimation methods can be classified as energy-based methods and phase-based methods. Energy-based methods exploit the fact that Fourier power spectrum of a translating image lies on a plane through the origin of the 3-D spatiotemporal frequency domain (1). Thus, energy-based methods compute translational velocity by comparing the output energy of a set of velocity-tuned filters (15,16). Phase-based methods exploit the fact that translation in the image domain correspond to a linear phase-shift in the 2-D spatial frequency domain. The simplest such method is the phase correlation method, which consists of: (1) Compute the phase correlation function as

$$\tilde{C}_{k,k+1}(f_1, f_2) = \frac{S_{k+1}(f_1, f_2)S_k^*(f_1, f_2)}{|S_{k+1}(f_1, f_2)S_k^*(f_1, f_2)|} \tag{23}$$

where $S_k(f_1, f_2)$ is the 2-D Fourier transform of the frame k with respect to the spatial variables x_1 and x_2, and $*$ denotes complex conjugation. (2) In the case of a translational motion between frames k and $k + 1$, the inverse 2-D Fourier transform of Eq. (23) yields

$$\tilde{c}_{k,k+1}(n_1, n_2) = \delta(n_1 - d_1, n_2 - d_2) \qquad (24)$$

which is an impulse whose location indicates the displacement (d_1, d_2). Implementation details are discussed in Ref. 1. Other phase-based motion estimation methods can be found in Refs. 16 and 17.

Hierarchical Motion Estimation. Hierarchical (multiresolution) representation of video (e.g., in the form of Laplacian pyramid or wavelet transform of frames), proves useful for motion estimation, especially to overcome the aperture problem. Although hierarchical estimation improves the performance of any of the above motion estimation methods, here we only discuss hierarchical block-matching.

The basic idea of hierarchical block-matching is to perform motion estimation successively at different levels of hierarchy, starting with the lowest resolution level (18). The lower resolution levels serve to determine a rough estimate of the displacement using relatively larger blocks. Note that the “relative size of the block” can be measured as the size of the block normalized by the size of the image at a particular resolution level. The estimate of the displacement vector at a lower resolution level is then passed onto the next higher resolution level as an initial estimate. The higher resolution levels serve to fine-tune the displacement vector estimate. At higher resolution levels, relatively smaller window sizes can be used, since we start with a good initial estimate.

Figure 9 illustrates hierarchical block-matching with 2 levels, where the maximum allowed displacement $M = 7$ for level 2 (lower resolution) and $M = 3$ for level 1 (higher resolution). Here, for simplicity, the pyramid contains images that are all the same size but successively more blurred as we go to the lower resolution levels. We simply skip over pixels in the lower resolution images (when computing matching criterion) to simulate the effect of subsampling. The best estimate at the lower resolution level is indicated by the circled “3.” The center of the search area in level 1 (denoted by “0”) corresponds to the best estimate from the second level. The estimates in the second and first levels are $[7, 1]^T$ and $[3, 1]^T$, respectively, resulting in an overall estimate of $[10, 2]^T$. Hierarchical block-matching can also be performed with subpixel accuracy by incorporating appropriate interpolation.

Motion Segmentation, Video Objects, and Video Object Tracking

Up to now, we considered video as a collection of frames without analyzing its content. There are several applications that motivate content-based analysis of video, which require segmentation of frames into regions of uniform motion and/or color, or semantically meaningful objects. They are: (1) if a scene contains multiple moving objects, motion estimation performance can be improved at object boundaries if a segmentation map is available (unfortunately, this is a chicken-and-egg problem, since automatic object segmentation often requires prior motion estimation); (2) 3-D motion and structure analysis requires segmentation of the scene into individual rigidly moving objects; and (3) object-based interactivity and manipulation require segmentation of semantically meaningful video objects.

Motion segmentation refers to labeling of pixels which are associated with different coherently moving regions. Various approaches exist for motion segmentation which may be classified as segmentation by change-detection analysis and temporal integration (19), by dominant (global) motion compensation (20,21,22,23), by affine clustering (24), by Hough transform analysis (25,26), by MRF modeling (27), by mixture modeling (28), by morphological methods (29,30), and simultaneous motion estimation and segmentation (31). Some of these methods are summarized below.

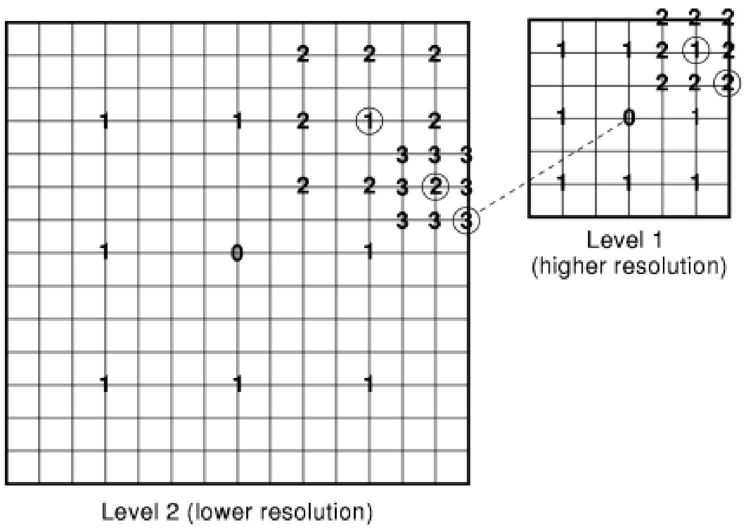

Fig. 9. Example of hierarchical block-matching with 2 levels.

It is difficult to associate a generic figure of merit with a motion segmentation result. If motion segmentation is employed to improve the compression efficiency, then oversegmentation may not be a concern. On the other hand, if it is used for object definition with application to object-based functionalities (as in the upcoming MPEG-4 standard), then it is of utmost importance that resulting motion boundaries align with actual object boundaries. Although it may not be possible to achieve this perfectly in a fully automatic manner, elimination of outlier motion vector estimates and imposing spatiotemporal smoothness constraints on the segmentation map improve the chances of obtaining more meaningful segmentation results.

A video object (*VO*) is defined as a semantically meaningful part of a scene. It is a spatiotemporal quantity as it relates to a group of pictures of a sequence. Video objects can be extracted using one of the following means: fully automatic extraction based on motion, color, or other relevant features of the object; semi-automatic extraction where the first instance of the video object is manually marked and then automatically tracked until it disappears; a third option is to capture each VO separately against a chroma-keyed background so it can be easily segmented, as long as the VO does not have any common colors with the background key. This last process is also referred to as *blue-screening,* because blue is often used as the chroma key. Extraction of semantically meaningful objects by fully automatic methods is extremely difficult except for a few special cases (e.g., foreground/background separation in the case of a static camera). However, automatic methods can be successfully employed to extract image regions with uniform motion and/or color between pairs of frames. Integration of these segments for video object definition, as well as tracking, is still an active area of research. In the following, we summarize recent methods for motion-based video segmentation, and automatic video object tracking.

Foreground/Background Separation with Stationary Camera. The simplest motion segmentation problem is separation of moving foreground objects from a stationary background (in the case of a static camera), which is also known as the motion detection problem. This can be achieved by thresholding or two-class clustering of successive or accumulative frame differences (19,32). Similar to motion estimation, motion detection and segmentation are also plagued with two fundamental limitations: occlusion and aperture problems. For example, pixels in a flat image region may appear to be stationary even if they are moving (due to the aperture problem); and/or erroneous motion vectors in the vicinity of covered/uncovered image regions (due to the occlusion problem) may yield false segmentation boundaries. These problems may be alleviated by hierarchical processing schemes.

Dominant Motion Estimation and Compensation. In the presence of a moving camera (e.g., zoom, pan, and/or tilt), the global camera motion must be compensated for before individual moving foreground objects can be detected. This leads to scene segmentation by dominant motion analysis, which refers to extracting one object (with the dominant motion) from the scene at a time (21,23). The dominant object may be the background or a foreground object. Multiple object segmentation can be achieved by repeating the procedure on the residual image after each object is extracted.

The global (dominant) motion is modeled by an affine transformation as

$$\begin{aligned} & s(x_1, x_2, k) \\ & = s(x_1 + a_1x_1 + a_2x_2 + a_3, x_2 + a_4x_1 + a_5x_2 + a_6, k+1), \end{aligned} \quad (25)$$

where $s(x_1, x_2, k)$ denotes image intensity at the pixel (x_1, x_2) in frame k. The parameters $c_1, \ldots, c_6$ for the first dominant object are estimated over the whole frame using a least-squares solution of the optical flow formulation expressed in terms of the Taylor series expansion of the video signal $s(x_1, x_2, t)$ (21). Because the Taylor series expansion assumes a small motion, the algorithm is used iteratively. Bergen et al. (21) made the interesting observation that the model yields the motion of the dominant object even in the presence of other moving objects. Given the motion parameter set $c_1, \ldots, c_6$, the dominant object boundary is identified by comparing $s(x_1, x_2, k)$ with the motion compensated frame $s_{MC}(x_1, x_2, k)$. Clearly, those pixels that are well compensated by Eq. (25) are marked to belong to the first dominant object. The procedure is then repeated on the remaining pixels to identify the next dominant object, and so on. Some difficulties with the dominant motion approach were reported when there is no dominant object in the scene.

Motion Segmentation by Parameter Clustering. Whereas the dominant motion method is a top-down method, motion segmentation by parameter clustering can be considered a bottom-up approach. It starts with a large number, N, of seed blocks uniformly distributed over the image, and fits a parametric model to the estimated motion field (optical flow) within each seed block. Then, the resulting model parameter vectors are clustered to find representative motion models.

Affine parameter clustering for motion segmentation was first proposed by Wang and Adelson (W–A) (24). This procedure can be mathematically described as: Given the affine parameter vectors $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N$, where

$$\mathbf{A}_n = \begin{bmatrix} a_{n,1} \\ a_{n,2} \\ a_{n,3} \\ a_{n,4} \\ a_{n,5} \\ a_{n,6} \end{bmatrix}, \qquad n = 1, \ldots, N, \quad (26)$$

find K cluster centers $\bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \ldots, \bar{\mathbf{A}}_K$, where $K \ll N$, and the label k, $k = 1, \ldots, K$, assigned to each affine parameter vector $\mathbf{A}_n$ which minimizes

$$\sum_{n=1}^{N} \mathcal{D}(\mathbf{A}_n, \bar{\mathbf{A}}_k) \quad (27)$$

The distance measure D between two affine parameter vectors $\mathbf{A}_n$ and $\bar{\mathbf{A}}_k$ is given by

$$\mathscr{D}(\mathbf{A}_n, \bar{\mathbf{A}}_k) = \mathbf{A}_n^T \mathbf{M} \bar{\mathbf{A}}_k \tag{28}$$

where $\mathbf{M}$ is a 6×6 scaling matrix.

The solution to this problem can be found by the well-known K-means algorithm, which consists of the following iteration: (1) Initialize $\bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \ldots, \bar{\mathbf{A}}_K$ arbitrarily. (2) For each seed block n, $n = 1, \ldots, N$, find k given by

$$k = Arg \min_s \mathscr{D}(\mathbf{A}_n, \bar{\mathbf{A}}_s) \tag{29}$$

where s takes values from the set $\{1, 2, \ldots, K\}$. (3) Define S_k as the set of seed blocks whose affine parameter vector is closest to $\bar{\mathbf{A}}_k$, $k = 1, \ldots, K$. Then, update the class means

$$\bar{\mathbf{A}}_k = \frac{\sum_{n \in S_k} \mathbf{A}_n}{\sum_{n \in S_k} \mathbf{1}} \tag{30}$$

(4) Repeat steps 2 and 3 until the class means $\bar{\mathbf{A}}_k$ do not change by more than a predefined amount between successive iterations. Statistical tests can be applied to eliminate some parameter vectors which are deemed as outliers. Furthermore, the number of clusters can be varied by splitting or merging of clusters between iterations.

Once the K cluster centers are determined, a label assignment procedure is employed to complete the motion segmentation. The segmentation label $L(i, j)$ for each pixel (i, j) is determined by

$$L(i, j) = Arg \min_k \|\mathbf{v}(i, j) - \mathscr{P}(\bar{\mathbf{A}}_k : (i, j))\|^2 \tag{31}$$

where k is from the set $\{1, 2, \ldots, K\}$, the operator P is defined as

$$\mathscr{P}(\bar{\mathbf{A}}_k : (i, j)) = \begin{bmatrix} (\bar{a}_{k,1} - \mathbf{1})i + \bar{a}_{k,2} j + \bar{a}_{k,3} \\ \bar{a}_{k,4} i + (\bar{a}_{k,5} - \mathbf{1}) j + \bar{a}_{k,6} \end{bmatrix} \tag{32}$$

and $\mathbf{v}(i, j)$ is the dense motion vector at pixel (i, j) given by

$$\mathbf{v}(i, j) = \begin{bmatrix} v_x(i, j) \\ v_y(i, j) \end{bmatrix} \tag{33}$$

where v_x and v_y denote the horizontal and vertical components, respectively. Several postprocessing operations may be employed to improve the accuracy of the segmentation gap.

The method of clustering in the affine parameter space has some drawbacks: (1) the metric Eq. (28) is not physically meaningful and clustering results are sensitive to the choice of the weight matrix $\mathbf{M}$, (2) clustering results are sensitive to small errors in the estimation of affine parameters, and (3) parameter clustering and label assignment procedures are decoupled; hence, ad hoc postprocessing operations which depend on some threshold values are needed to clean up the final segmentation map. However, successful results have been obtained through temporal integration of segmentation maps.

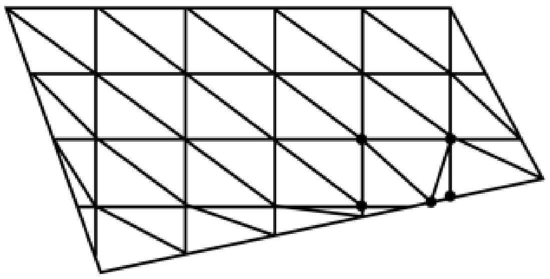

Fig. 10. A 2-D uniform mesh overlaid on an object with arbitrary shape which is approximated by a quadrilateral.

Combined Motion Estimation and Segmentation. Motion estimation and segmentation are interrelated since good motion segmentation requires good motion estimation, and vice versa. For example, the occlusion and aperture problems in motion estimation are usually responsible for misalignment of motion and actual object boundaries and oversegmentation of the motion field. Furthermore, all pixel-based motion segmentation methods, including (21,24), suffer from the drawback that the resulting segmentation map may contain isolated labels.

In recognition of these facts, a Bayesian approach to simultaneous motion estimation and segmentation that is based on modeling the 2-D motion field as the sum of a parametric field and a residual field has been proposed by Chang et al. (31). This method iterates between motion estimation and segmentation imposing mutually consistent constraints within a Bayesian estimation framework. Spatial continuity constraints in the form of Gibbs random field models have also been introduced in Ref. 27 to obtain smoother segmentation results. However, the computation cost of these Bayesian motion segmentation algorithms limits their practical use.

Video Object Tracking. Existing methods for object tracking can be broadly classified as boundary (shape) tracking, and region tracking methods. Boundary tracking has been addressed by using locally deformable (active) contour models (33) [or snakes (34)], and deformable template models (35). Region tracking methods can be categorized as those that employ global deformation models (36) and those that allow for local deformations. For example, the region tracking method of Meyer and Bouthemy (37) uses a single affine motion model within each region of interest and assigns a second-order temporal trajectory to each affine model parameter. More recently, a 2-D triangular mesh-based object representation has been proposed for tracking local deformations of objects (14) as well as their global motion. The mesh model describes the dense motion field as a collection of flexibly connected non-overlapping rigid patches, where the motion of each patch is modeled by an affine mapping. In the following, we briefly summarize 2-D mesh-based object tracking since it allows for more general object motions.

Here, in order to define semantically meaningful objects, we assume that the initial contour of the object is marked by the user manually. In Ref. 14, the boundary of the object is approximated by a polygon with a small number of vertices, and a 2-D uniform mesh is overlaid on the object, as depicted in Fig. 10. A generalized block matching method was employed at the vertices of the boundary polygon to predict its location in the next frame. This is depicted in Fig. 11. The motion of each mesh node point is linearly predicted from the displacements of the vertices of the boundary polygon under the assumption of “mild” deformations, and then refined by a connectivity preserving fast search algorithm (14). However, this technique does not account for occlusion of any node points.

In a subsequent work (38), the boundary of the object has been modeled by an active contour (uniform B-spline), which is snapped to the actual boundary at each frame using energy minimization for better boundary tracking. Improved motion estimation and content-adaptive triangulation methods (13) are also employed to handle occlusions. Triangular mesh-based object mosaics are also introduced to perform texture mapping in the

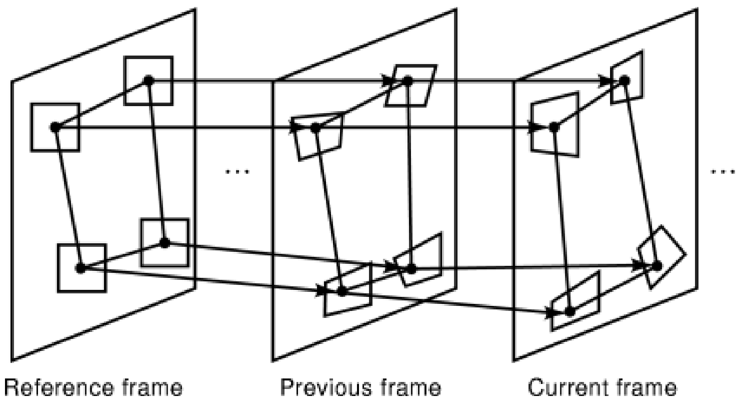

Fig. 11. Tracking of the boundary nodes.

presence of partial occlusions. However, robust tracking in the presence of self-occlusion (e.g., object entry/exit, out-of-plane rotations) or an object covering/uncovering another is still an active area of research.

Image Sequence Filtering

Filtering problems can be broadly classified as standards conversion, noise filtering (de-noising), de-blurring, and high-resolution filtering. Image sequence filtering (de-noising, de-blurring, and superresolution) becomes especially important when still images from video are desired. This is because the blur, noise, and aliasing artifacts become rather objectionable when observing a "freeze-frame," although they may not be visible to the human eye at the usual frame rates. Since many video signals encountered in practice are interlaced, we address the cases of both progressive and interlaced video. Filters used for these tasks may be intraframe/field, motion-adaptive, or motion-compensated filters. Motion-compensated filters employ explicit estimates of motion vectors/fields in the filtering operation. Although motion-compensated filtering is optimal when motion trajectories can be accurately estimated, it is very sensitive to motion estimation errors. This fact, along with the high cost of hardware implementation of motion estimation, motivates use of suboptimal but more robust algorithms which do not require explicit estimation of motion such as intraframe/field and motion-adaptive filters. Intra-frame/field filtering refers to algorithms that employ data from a single frame/field. The form of motion-adaptive filters depends on the value of a motion detection signal; however, they do not require motion vector/field estimation.

Standards Conversion. Video signals are usually in different formats for different applications; e.g., NTSC and PAL broadcast signals are 2:1 interlaced with 525/60 and 625/50 lines per frame and fields per second, respectively, and employ rectangular pixels; a typical high-resolution computer monitor is progressive (non-interlaced) with 1280 pixels × 1024 lines × 72 frames per second and employs square pixels; and movies are progressively digitized with 4K pixels × 3K lines and 24 frames per second. Standards conversion refers to spatial and temporal video format manipulations to decouple the source format from the display format. This allows re-purposing content by, for example, conversion of movies to NTSC or PAL, display of NTSC or PAL video on progressive computer monitors, and so on.

Simple Field/Frame Rate Conversion.

"3 to 2 Pull-Down". The 3:2 pull-down method has long been used for conversion of motion picture sources to NTSC video. It is a simple frame repetition method, where each odd frame of the digitized motion picture is repeated three times and each even frame is repeated twice, or vice versa, yielding a 60 Hz field rate from

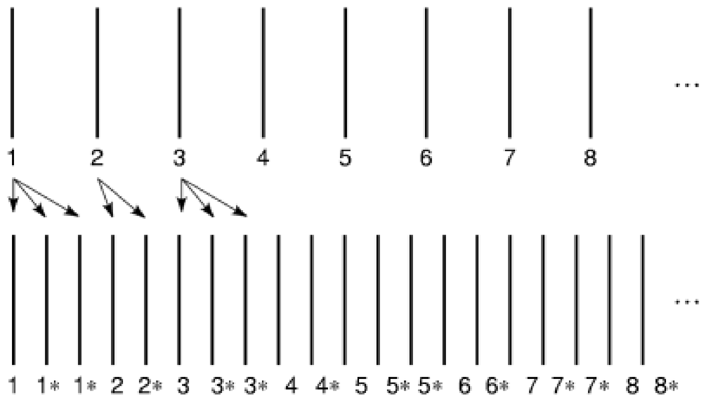

Fig. 12. Illustration of the 3:2 pull-down method.

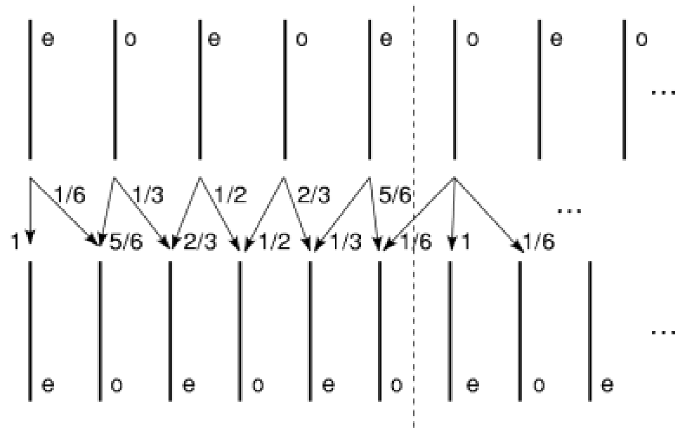

Fig. 13. Illustration of 50–60 Hz field rate conversion by linear filtering.

24 Hz input source. This is depicted in Fig. 12. The 3:2 pull-down method introduces temporal aliasing which results in jerky motion rendition. This is hardly visible at the spatiotemporal resolution provided by current NTSC receivers; however, with bigger displays and high-resolution video formats, more sophisticated frame /field rate conversion algorithms are needed.

The inverse of this procedure, which generates a 24 Hz noninterlaced source from a 60 fields/s interlaced video, has been adopted by the Moving Picture Experts Group (*MPEG*) as a preprocessing step for reduncy reduction in MPEG-2 compression of NTSC video sources which were converted from motion picture by the 3:2 pull-down method (39).

50–60 Hz Conversion. The NTSC standard employs 60 fields/s and 262.5 lines/field, whereas the PAL uses 50 fields/s and 312.5 lines/field. The conversion from NTSC to PAL may be achieved by dropping a complete frame (an even and an odd field) every six frames, and spatially interpolating for the missing lines. PAL to NTSC conversion requires dropping some extra lines per frame, and replicating a complete frame every five frames. Smoother results can be obtained by interpolating fields in time (rather than dropping or replicating them) to achieve the desired field rate. Weights of a commonly used two-tap linear interpolation filter for 50 to 60 Hz conversion are shown in Fig. 13.

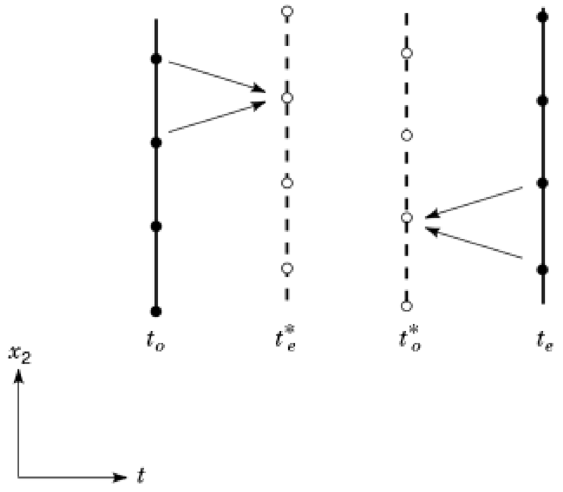

Fig. 14. Two-point line averaging.

Scan Rate Doubling. Doubling of the field rate has been adopted by several TV manufacturers in Europe for commercial 100-Hz receivers to improve visual quality. In digital TV receivers, it is very easy to replicate each field twice to achieve scan rate doubling. There exists more than one way to repeat the fields. For example, an odd field may be repeated to form the next even field, and an even field is repeated to form the next odd field. This method has reasonably good performance in moving scenes, but poor results in stationary regions is inevitable. Alternatively, one can repeat an even field to form the next even field, and an odd field to form the next odd field. This strategy is optimal for stationary scenes but fails for moving parts.

Yet another alternative is a line averaging filter, which is depicted in Fig. 14. The solid circles in Fig. 14 denote input pixels, and open circles pointed by the arrows denote the output pixels. The line-averaging filter provides reasonable performance in moving regions. However, it introduces blurring in stationary image parts because it is purely a spatial filter. Notice that while the repetition algorithms yield jagged edges, averaging algorithms provide blurred edges and may introduce ghost artifacts in moving regions. Obviously, none of the above algorithms alone is satisfactory for both stationary and moving regions of the scene, which motivates the need for motion-adaptive or motion-compensated filtering schemes.

Motion-Adaptive Field/Frame Rate Conversion. A shift-varying linear filter, with a three-pixel support as depicted in Fig. 15, where the filter impulse response is determined locally based on a motion detection function, provides the flexibility needed for obtaining improved results. For example, we can perform averaging only in the stationary image regions, and replicate pixels or compute a weighted average in the moving regions. The moving regions can be estimated by using a motion detection function, which may simply be the frame difference as in change detection (see also the section on Motion-Adaptive De-Interlacing). Because no optimal strategy exists to determine the filter weights in terms of the motion detection function in the moving areas, some researchers suggested the use of spatiotemporal median filtering.

Median filtering is known to be edge-preserving in intra-frame image processing. Considering the effect of motion as a temporal edge, spatiotemporal median filtering should provide motion adaptivity. Three-point median filtering has been used in prototype improved-definition TV receivers for field rate doubling. The support of the filter, for even and odd fields, is as shown in Fig. 15. Several modifications including a combination of averaging and median filters have been proposed for performance improvements (40).

Simple De-Interlacing. De-interlacing refers to up-conversion from an interlaced to a progressive sampling lattice at the same temporal rate.

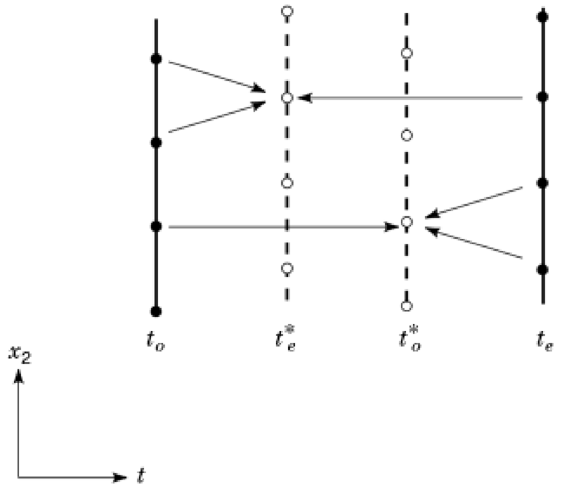

Fig. 15. Three-point filtering for field rate doubling.

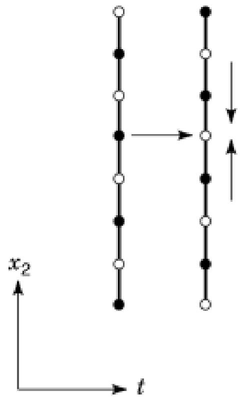

Fig. 16. Two-field filtering for de-interlacing.

Field Merging. The simplest de-interlacing method is merging the even and odd fields of a frame, that is, copying samples as shown by the horizontal arrow in Fig. 16. This method yields $N/2$ progressive frames from N fields. The resulting frames are known as *composite frames.* Composite frames provide perfect resolution in stationary image regions, but they may suffer from serious motion artifacts.

Linear Intrafield De-Interlacing. Motion artifacts can be avoided by interpolating each field individually. If only $N/2$ progressive frames are desired from N fields, all even or all odd fields may be dropped. Intraframe de-interlacing can employ line repetition, line averaging, or more sophisticated interpolation filtering. The projection of two frames of an interlaced video on the (x_2, t) coordinates is shown in Fig. 17, where each circle denotes the cross-section of a complete line of video. The shaded circles denote lines that are available, and the open circles show the lines to be interpolated. Similar to the case of frame/field rate conversion, the line repetition algorithm results in jagged edges, while the line-averaging algorithm causes undesired blurring.

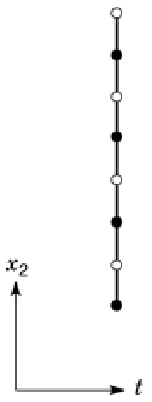

Fig. 17. Intrafield filtering.

Edge-Adaptive Intrafield De-Interlacing. Edge-adaptive spatial interpolation methods have been proposed to avoid blurring edges (41,42,43). In the edge-adaptive approach, each line of video in a given frame t_0 is modeled as a horizontally displaced version of the previous line in the same frame given by

$$s(x_1 - d/2, x_2 - 1, t_0) = s(x_1 + d/2, x_2 + 1, t_0) \tag{34}$$

where d denotes the horizontal displacement between two consecutive even or odd lines. This model suggests a 1-D motion compensation problem where x_2 takes the place of the time variable. The displacement d at each pixel can be estimated by either using symmetric line segment matching about (x_1, x_2) in order to minimize the summed absolute difference (*SAD*) given by (41)

$$\begin{aligned} SAD(d) = \sum_{j=-1}^{1} &|s(x_1 - d/2 + j, x_2 - 1, t_0) \\ &- s(x_1 + d/2 + j, x_2 + 1, t_0)| \end{aligned} \tag{35}$$

or through the relation (45,46)

$$d/2 \frac{\partial s(x_1, x_2, t_0)}{\partial x_1} + \frac{\partial s(x_1, x_2, t_0)}{\partial x_2} = 0 \tag{36}$$

which is the equivalent of the optical flow equation in this case. Then an edge-adaptive contour interpolation filter can be defined as

$$\begin{aligned} s(x_1, x_2, t_i) = \frac{1}{2}&[s(x_1 - d/2, x_2 - 1, t_i) \\ &+ s(x_1 + d/2, x_2 + 1, t_i)], \qquad i = e, o \end{aligned} \tag{37}$$

where e and o denote even and odd fields, respectively. This filter seeks those two pixels in the two neighboring lines that most likely belong to the same image structure, i.e., on the same side of the edge, and averages them. The fact that it is capable of preserving a 45 degree edge, unlike the linear averaging filter, is demonstrated in Fig. 18. The crucial step here is the accurate estimation of the local displacement values.

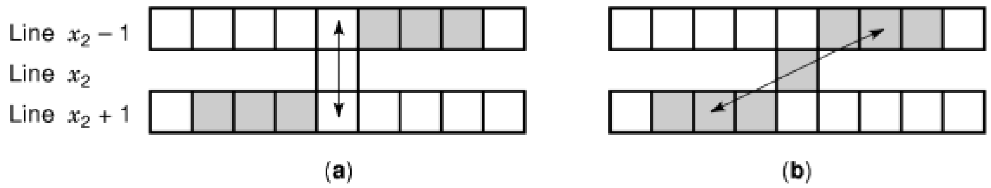

Fig. 18. Demonstration of (a) linear versus (b) edge-adaptive interpolation.

Intraframe filtering methods lead to simple hardware realizations. However, they are not well suited to de-interlacing in stationary regions, where spatial averaging usually causes blurring of image details; hence, the need for motion-adaptive de-interlacing.

Motion-Adaptive De-Interlacing. Motion-adaptive filtering refers to employing different filtering strategies in the presence and absence of motion without motion estimation. For example, in de-interlacing, in the absence of motion, the best strategy is to merge even and odd fields, which simply doubles the vertical resolution. However, in the presence of motion this technique would suffer from motion artifacts. Explicit schemes make use of a motion-detection function and apply different filters depending on the value of this function.

In order to obtain an acceptable performance in both the moving and stationary regions, we may consider motion-adaptive interframe filtering, which switches between merging and intraframe interpolation (41) or linearly blends them (44), based on a motion detection function. Two examples of a three-point motion-adaptive filter, whose support is depicted in Fig. 16, are the three-point weighted averaging filter,

$$\begin{aligned} s(x_1, x_2, t_i) = {} & \alpha\, s(x_1 - d/2, x_2 - 1, t_i) \\ & + (1-\alpha) s(x_1 + d/2, x_2 + 1, t_i) \\ & + \beta\, s(x_1, x_2, t_i - 1) \end{aligned} \tag{38}$$

and the three-point median filter (40)

$$\begin{aligned} s(x_1, x_2, t_i) = {} & \text{Med}\{s(x_1 - d/2, x_2 - 1, t_i), \\ & s(x_1 + d/2, x_2 + 1, t_i), s(x_1, x_2, t_i - 1)\} \end{aligned} \tag{39}$$

where α and β are determined based on the value of a motion detection function, and the parameter d, computed from Eq. (35) or Eq. (36), allows for edge-adaptive intraframe interpolation.

We can employ a three- or four-field motion detection function. The three-field motion detection function is obtained by thresholding the difference between two fields of the same polarity (even-even or odd-odd), whereas the four-field motion detection function takes the logic OR of the thresholded differences of the respective even-even and odd-odd fields. The coefficients α and β may be given by

$$\begin{aligned} &\alpha = 0.5, \quad \beta = 0 \quad \text{if motion is detected, and} \\ &\alpha = 0, \quad \beta = 1 \quad \text{if no motion is detected.} \end{aligned}$$

Motion adaptive methods provide satisfactory results, provided that the scene does not contain fast-moving objects. In the presence of fast-moving objects some sort of motion-compensated filtering, as described next, is needed for the best results.

Global-Motion-Compensated De-Interlacing. Motion-compensated filtering, in general, requires a different motion trajectory at each pixel. However, in practice, reliable estimation of these vectors at each pixel

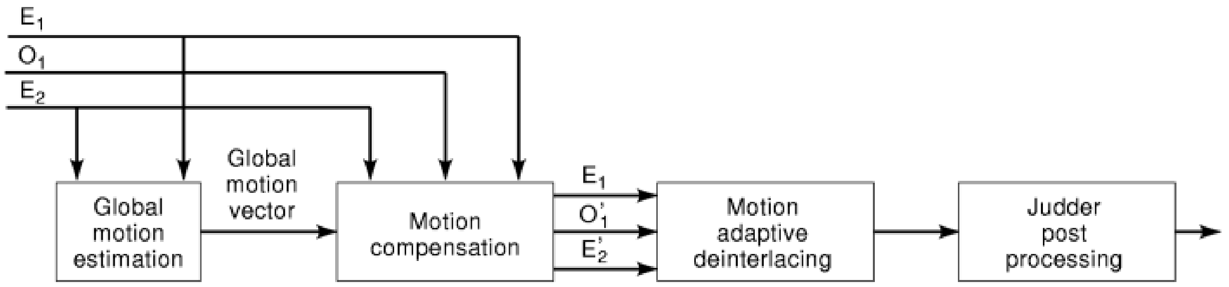

Fig. 19. Motion-compensated/adaptive de-interlacing.

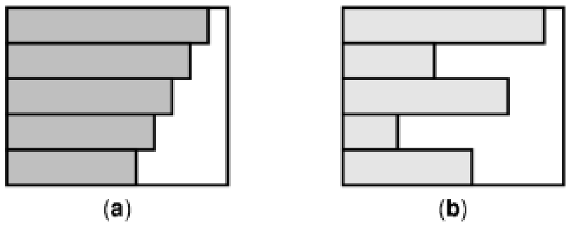

Fig. 20. Illustration of judder: (a) no judder; (b) judder present.

poses serious problems. To this effect, we propose a hybrid de-interlacing method, where we compensate for a single global motion, which is due to camera pan or shake, and then employ a motion-adaptive filter on the globally compensated image to account for any residual motion. The block diagram of a three-field hybrid de-interlacing filter is depicted in Fig. 19, where the three consecutive fields are assumed to be an even field E_1, an odd field O_1, and an even field E_2.

In the first stage, a global-motion vector between the fields E_1 and E_2 is estimated using the phase correlation method over four rectangular windows that are located near the borders of the fields, so that they are most likely affected by global motion only. Next, the fields O_1 and E_2 are motion-compensated with respect to E_1 to generate O'_1 and E'_2, respectively. The motion compensation step aims to create three consecutive fields, E_1, O'_1, and E'_2, that represent the interlaced video if no global motion were present. Subsequently, the three-field motion-adaptive weighted averaging filter, described above, is applied to the field sequence E_1, O'_1, and E'_2. Finally, a judder post-processing step, proposed by Zaccarin and Liu (45), has been included. Judder, illustrated in Fig. 20, refers to edge misalignment artifacts caused by incorrect motion vectors. In the postprocessing stage, the motion vectors at the pixels where judder is detected are deemed unreliable, and the corresponding pixels are replaced by spatially interpolated values.

Motion-Compensated Filtering. Motion-compensated filtering is the optimal standards up-conversion approach, provided that the motion trajectories can be accurately estimated. The basic concept of motion-compensated filtering is the same for frame/field rate up-conversion and de-interlacing; that is, to perform filtering along the motion trajectories passing through the missing pixels (46). The two problems differ only in the spatiotemporal locations of the missing samples. The procedure consists of: (1) motion estimation, (2) postprocessing of motion vectors, and (3) filter design. The reader is referred to (1) for filter design issues.

Motion Estimation. Several motion estimators have been proposed for use in standards conversion (47, 48). In motion-compensated up-conversion, we need to estimate motion trajectories that pass through missing pixel locations. A simple method that achieves this is the *symmetric block matching,* which is illustrated in Fig. 21(c). Unlike the forward or backward block matching [shown in Fig. 21(a) and 21(b)], in this scheme, blocks

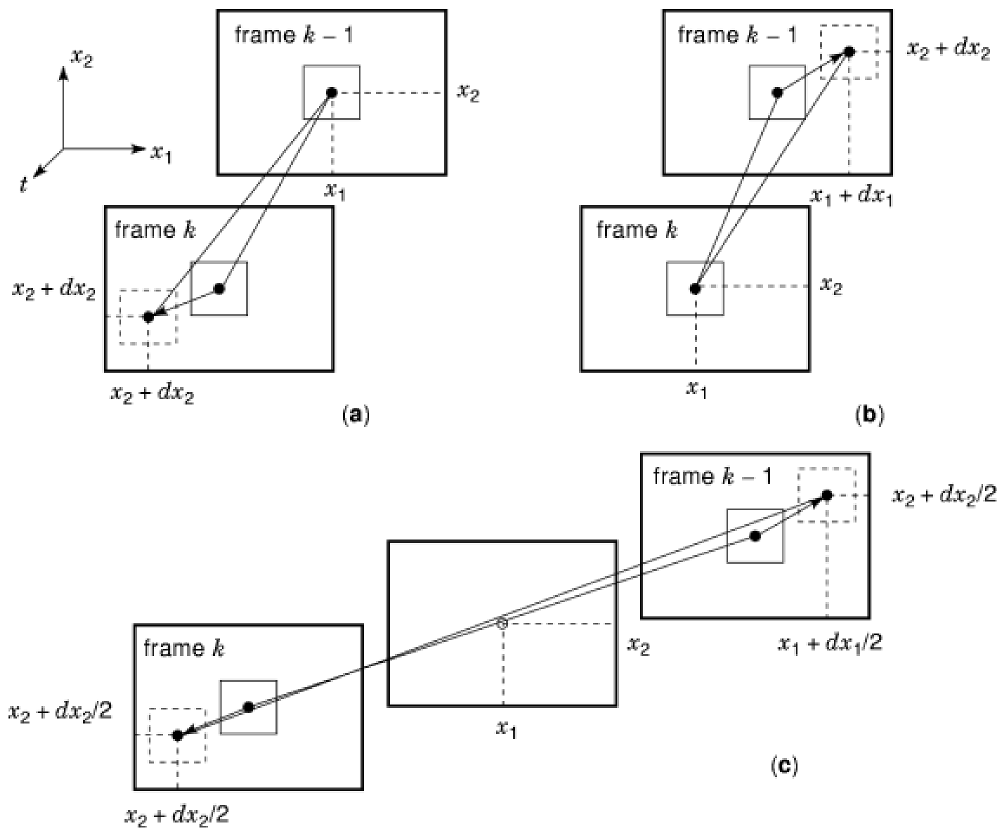

Fig. 21. (a) Forward, (b) backward, and (c) symmetric block matching.

in the two existing neighboring frames/fields k and $k-1$ are moved symmetrically so that the line connecting the centers of these two blocks always passes through the missing pixel of interest (x_1, x_2) (49).

Postprocessing of Motion Estimates. The accuracy of the motion estimates is probably the most important factor in the effectiveness of motion-compensated interpolation. Thus, some kind of postprocessing is usually applied to the estimated motion vectors to improve their accuracy. An easy method to detect unreliable motion vectors is to test the DFD between the frames k and $k+1$. All vectors yielding a DFD that is above a prespecified threshold are discarded. The unreliable motion vectors can be replaced by a set of candidate motion vectors if any of the candidate vectors yields a DFD that is less than the threshold. The candidate vectors can be determined based on the analysis of the histogram of the reliable motion vectors, where the dominant peaks of the histogram indicate background motion or the motion of large objects in the scene (50). If no reliable replacement motion vector can be found at a pixel, it is marked as a motion estimation failure, where a motion-adaptive or an intraframe filter is employed.

Image Sequence Estimation. Digital video recorded by consumer camcorders, still cameras, or even broadcast-quality video cameras and motion-picture cameras are degraded by some amount of noise. Noise sources include film grain, sensor, quantization, and compression noise. Image sequence estimation refers to estimating noise-free images by means of statistical modeling and filtering. It should be rather obvious that exact separation of variations in image intensity which are due to noise from genuine image detail is impossible. To this effect, both the image and noise will be characterized by statistical models. In general, the noise can be modeled as additive or multiplicative, signal-dependent or signal-independent, and white or colored. For example, photon noise and film-grain noise are signal-dependent, whereas CCD sensor noise and quantization noise are usually modeled as white, Gaussian distributed, and signal-independent. Here, we assume a simple additive noise model given by

$$g(n_1, n_2, k) = s(n_1, n_2, k) + v(n_1, n_2, k) \tag{40}$$

where $s(n_1, n_2, k)$ and $v(n_1, n_2, k)$ denote the ideal video and noise at frame k, respectively.

Noise filters can be classified as spatial (intraframe) and spatio-temporal (interframe) filters. In intraframe filtering, there is a tradeoff between noise reduction and spatial blurring of the image detail. Spatiotemporal filters are 3-D filters, which utilize not only the spatial correlations, but also the temporal correlations between the frames. We classify spatio-temporal filters as motion-adaptive and motion-compensated filters. Motion-adaptive filtering utilizes motion-detection, but does not require explicit estimation of interframe motion vectors. This is distinct from motion-compensated filtering schemes, which operate along motion trajectories, thus requiring exact knowledge of the trajectories at each pixel.

Estimation of the Noise Variance. Since we model the noise by a zero-mean, white Gaussian random process that is uncorrelated with the image, it is completely characterized by its variance. The variance of the noise is commonly estimated by the sample variance computed over a low-contrast local region of the observed image. As will be seen in the following, the noise variance plays an important role in defining constraints for image estimation and restoration algorithms.

Intraframe Filtering. We can classify intraframe noise filters into three categories: (1) linear, shift-invariant (*LSI*) filters, such as the weighted averaging filters and linear minimum mean square error (*LMMSE*) filters, also known as Wiener filters, (2) nonlinear filters, such as median filters and other order statistics filters, and (3) adaptive filters, such as directional smoothing filters and local (space-varying) LMMSE filters.

LMMSE (Wiener) Filter. The Wiener filter gives the minimum mean square error estimate of the ideal image among all linear filters, based on the model that the noise is image-independent, and image and noise are wide-sense stationary (that is, their means are constant and their correlation functions are shift-invariant). The frequency response of the Wiener filter is then given by (1)

$$H(f_1, f_2) = \frac{P_{ss}(f_1, f_2)}{P_{ss}(f_1, f_2) + P_{vv}(f_1, f_2)} \tag{41}$$

where $P_{ss}(f_1, f_2)$ and $P_{vv}(f_1, f_2)$ denote the power spectra of the image and noise, respectively. The mean of the image and noise are taken to be zero without loss of generality, since any nonzero mean can be removed prior to filtering. We observe that the Wiener noise filter is a low-pass filter, since the image power spectrum diminishes at high frequencies, which implies that the filter frequency response goes to zero at those frequencies. A realizable approximation to the filter (Eq. 41) can be obtained by a technique known as *frequency-sampling design,* where the filter frequency response $H(f_1, f_2)$ is sampled in the frequency domain using $N_1 \times N_2$ samples. The samples can be efficiently computed using a $N_1 \times N_2$ fast Fourier transform (*FFT*).

Adaptive (Local) LMMSE Filtering. Linear shift-invariant filters are limited in their ability to separate genuine image variations from noise, because they are based on wide-sense stationary (homogeneous) image models. In order to develop an easily implementable adaptive filter that preserves image detail, Kuan et al. (51) proposed a simple space-varying image model, where the local image characteristics are captured in a space-varying mean, and the residual after removing the local mean was modeled by a white Gaussian process.

The LMMSE estimator, based on this image model, can be expressed as (1)

$$\hat{s}(n_1, n_2) = \mu_s(n_1, n_2) + \frac{\sigma_s^2(n_1, n_2)}{\sigma_s^2(n_1, n_2) + \sigma_v^2}[g(n_1, n_2) - \mu_s(n_1, n_2)] \tag{42}$$

where $g(n_1, n_2)$ and $\hat{s}(n_1, n_2)$ denote the noisy observation and the estimate, $\mu_s(n_1, n_2)$ and $\sigma^2{}_s(n_1, n_2)$ are the local mean and variance of the original image, respectively, and $\sigma^2{}_v$ is the variance of the noise. The resulting filter, which has a predictor-corrector structure, is easy to implement, yet avoids excessive blurring in the vicinity of edges and other image detail.

Note that the adaptive LMMSE filter (Eq. 42) requires the estimation of the mean $\mu_s(n_1, n_2)$ and the variance $\sigma^2{}_s(n_1, n_2)$ at each pixel. We estimate the local sample mean and sample variance over an $M \times M$ window, W, as

$$\mu_s = \sum_{(n_1, n_2) \in W} g(n_1, n_2) \tag{43}$$

and

$$\sigma_g^2 = \sum_{(n_1, n_2) \in W} [g(n_1, n_2) - \mu_g(n_1, n_2)]^2 \tag{44}$$

We have

$$\sigma_s^2 = \max\{\sigma_g^2 - \sigma_v^2, 0\} \tag{45}$$

so that $\sigma^2{}_s$ is always nonnegative. It is assumed that the variance of the noise $\sigma^2{}_v$ is either known or can be estimated from a uniform image region.

It is interesting to note that when $\sigma^2{}_s$ is small, the second term in Eq. (42) is negligible, and the adaptive LMMSE filter approaches a direct averaging filter. On the other hand, when $\sigma^2{}_s$ is large compared with $\sigma^2{}_v$, the filter is turned off. Because large $\sigma^2{}_s$ usually indicates presence of edges, the adaptive LMMSE filter preserves edges by effectively turning the filter off across edges. Consequently, some noise is left in the vicinity of edges which may be visually disturbing.

Directional Filtering. Directional filtering is an edge-preserving de-noising approach that aims to reduce the noise also in the vicinity of the edges. This is achieved by filtering along the edges, but not across them. In particular, possible edge orientations are generally quantized into four: 0°, 45°, 90°, and 135°; and five FIR filter kernels, one for each orientation and one for nonedge regions, are defined. The supports of the edge-oriented FIR filters are depicted in Fig. 22.

There exist two approaches for directional filtering. The first is to select the most uniform support out of the five at each pixel according to a criterion of uniformity or by edge-detection (52). The variance of pixels within each support can be used as a selection criterion. Then, the edge-kernel with the lowest variance indicates the most likely edge orientation at that pixel. Filtering can be performed by averaging the pixels in the direction of the smallest variance. The second approach is to apply an edge-adaptive filter, such as the adaptive LMMSE filter, within each kernel at each pixel, and cascade the results (53). Recall that the local LMMSE filter is effectively off within those supports with a high variance. Thus, we expect that effective filtering is performed only over those kernels with a small variance. This method avoids an explicit support selection step. It offers satisfactory noise reduction around edges, since at least one of the filters should be active at every pixel.

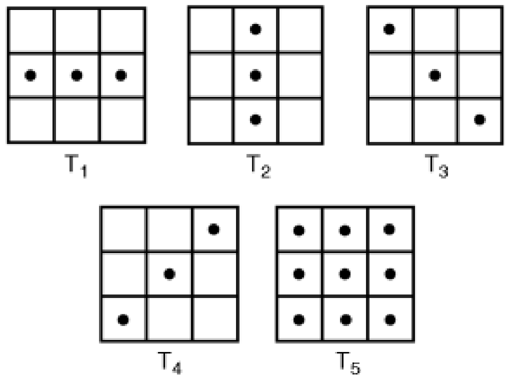

Fig. 22. Directional filtering kernels.

Median and Weighted Median Filtering. The median filter computes the median of pixel intensities within the support of the filter. The median filter is implicitly edge-preserving because it can reject outlier intensities; thus, avoiding blurring across edges (54). Fast algorithms for one- and two-dimensional (separable) median filtering exist for real-time implementations (55). In median filtering each sample in the filter support is given an equal emphasis. The weighted median filter is an extension of the median filter, where each sample (i_1, i_2) is assigned a weight w_{i_1,i_2}. The weighting is achieved by replicating the (i_1, i_2)th sample w_{i_1,i_2} times. The properties of the filter vary, depending on how the weights are assigned. The reader is referred to Ref. 56 and the references therein for other generalizations of median filtering, including order statistic and multistage order statistic filters.

Motion-Adaptive Filtering. Interframe noise filtering may provide several advantages over intraframe filtering, such as avoiding spatial blurring. In this section, we discuss motion-adaptive noise filters where there is no explicit motion estimation. These filters are applied over a fixed spatiotemporal support at each pixel. We start with direct filtering where the adaptivity is implicit in the filter design. Next, we cover filter structures, where some coefficients vary as a function of a so-called "motion-detection" signal.

Direct Filtering. The simplest form of direct filtering is frame averaging, where there is no motion adaptivity. Direct temporal averaging is well suited to stationary parts of the image, because averaging multiple observations of essentially the same pixel in different frames eliminates noise while resulting in no loss of spatial image resolution. In purely temporal filtering a large number of frames may be needed for effective noise reduction, which requires a large number of frame stores. Spatiotemporal filtering provides a compromise between the number of frame stores needed for effective noise reduction and the amount of spatial blurring introduced. Although direct temporal averaging serves well for stationary image regions, it may lead to smearing and chrominance separation in the moving areas. These degradations may be avoided if the filter makes use of interframe motion information.

Motion-adaptive filtering is the temporal counterpart of edge-preserving spatial filtering in that frame-to-frame motion gives rise to temporal edges. A fundamental question is how to distinguish the temporal variations due to motion from those due to noise. It follows that spatiotemporal noise filters that adapt to motion can be obtained by using structures similar to those of the edge-preserving filters. Examples of such filters include directional filters and order statistic filters, including median, weighted median, and multistage median filters (56). For instance, Martinez and Lim (57) proposed a cascade of five one-dimensional finite impulse response (*FIR*) linear minimum mean square error (LMMSE) estimators over a set of five hypothesized motion trajectories at each pixel. These trajectories correspond to no motion, motion in the $+x_1$ direction, motion in the $-x_1$ direction, motion in the $+x_2$ direction, and motion in the $-x_2$ direction. Due to the adaptive nature of the LMMSE estimator, filtering is effective only along hypothesized trajectories that are close to actual ones.

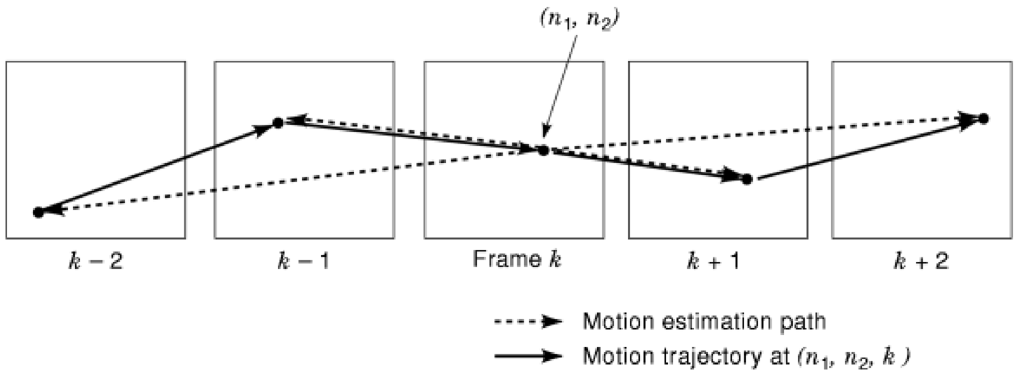

Fig. 23. Estimation of the motion trajectory ($n = 5$).

This approach has been reported to be successful in cases where one of the hypothesized motion trajectories is close to the actual one.

Motion-Detection Based Filtering. Motion-detection based noise filtering is analogous to motion-adaptive frame/scan rate conversion and de-interlacing algorithms, except that in standards conversion the "present pixel" does not have an observation and it needs to be interpolated from its neighbors, whereas in noise filtering the "present pixel" has a noisy observation and it needs to be estimated. Both FIR and IIR filter structures can be employed in motion adaptive filtering. The selected filter structure has parameters which can be tuned according to a motion-detection signal, such as the frame difference, which tends to turn the filtering off when a large motion is detected in an attempt to prevent artifacts. The FIR structure has limited noise-reduction ability, especially when used as a purely temporal filter with a small number of frames, because the reduction in the noise variance is proportional to the number of samples in the filter support. IIR filters are more effective in noise reduction, but they generally cause Fourier phase distortions. Several implementations of motion-adaptive noise filters have been proposed in the literature, which generally differ in the way they compute the motion-detection signal. Use of motion-adaptive IIR filters in practical systems have been demonstrated by McMann et al. (58) and Dennis (59).

Motion-Compensated Filtering. The motion-compensated approach is based on the assumption that the variation of the pixel gray levels over any motion trajectory is due mainly to noise. Thus, noise in both the stationary and moving areas of the image can effectively be reduced by low-pass filtering over the respective motion trajectory at each pixel. Motion-compensated filters differ according to (1) the motion estimation method, (2) the support of the filter, (e.g., temporal versus spatiotemporal) and (3) the filter structure (e.g., FIR versus IIR, adaptive versus nonadaptive).

The concept and estimation of a motion trajectory are illustrated in Fig. 23. Suppose we filter the kth frame of an image sequence using N frames $k - M, \ldots, k - 1, k, k + 1, \ldots, k + M$, where $N = 2M + 1$. The first step is to estimate the discrete motion trajectory at each pixel (n_1, n_2) of the kth frame. The discrete motion trajectory is depicted by the solid line in Fig. 23 for the case of $N = 5$. The displacement vectors are usually estimated in reference to the frame k as indicated by the dotted lines. The trajectory in general passes through subpixel locations, where the intensities can be determined via spatial or spatiotemporal interpolation. The support $S_{n_1,n_2,k}$ of a motion-compensated spatiotemporal filter is defined as the union of predetermined spatial neighborhoods (e.g., 3×3 regions) centered about the pixel (subpixel) locations along the motion trajectory. In temporal filtering, the filter support $S_{n_1,n_2,k}$ coincides with the motion trajectory. Clearly, the effectiveness of motion-compensated spatiotemporal filtering is strongly related to the accuracy of the motion estimates.

Various filtering techniques, ranging from averaging to more sophisticated adaptive filtering, can be employed given the motion-compensated filter support. In the ideal case, where the motion estimation is perfect, direct averaging of image intensities along a motion trajectory provides effective noise reduction (6,570). In practice, motion estimation is hardly ever perfect, due to noise and sudden scene changes, as well as changing camera views. As a result, image intensities over an estimated motion trajectory may not necessarily correspond to the same image structure, and direct temporal averaging may yield artifacts. In the case of spatiotemporal filtering, there may also be spatial variations within the support; hence the need for adaptive filter structures over the motion-compensated filter support. Here, we consider two adaptive filter structures: the adaptive LMMSE filter, which is aimed at reducing the amount of filtering whenever a nonuniformity is detected within the motion-compensated support, and the adaptive weighted averaging (AWA) filter, which is aimed at weighting down the effect of the outliers that create the nonuniformity and achieving effective filtering by concentrating on the remaining similar image intensities.

Spatio-Temporal Adaptive LMMSE Filtering. The motion-compensated adaptive LMMSE filter (61,62) is an extension of the edge-preserving spatial filter proposed by Lee (63) and Kuan et al. (51) to the spatiotemporal domain, where the local spatial statistics are replaced by their spatiotemporal counterparts. Then, the estimate of the pixel value at (n_1, n_2, k) is given by

$$\begin{aligned}\hat{s}(n_1, n_2, k) = {} & \frac{\hat{\sigma}_s^2(n_1, n_2, k)}{\hat{\sigma}_s^2(n_1, n_2, k) + \hat{\sigma}_v^2(n_1, n_2, k)} g(n_1, n_2, k) \\ & + \frac{\hat{\sigma}_v^2(n_1, n_2, k)}{\hat{\sigma}_s^2(n_1, n_2, k) + \hat{\sigma}_v^2(n_1, n_2, k)} \hat{\mu}_g(n_1, n_2, k)\end{aligned} \tag{46}$$

where the sample mean $\hat{\mu}_g(n_1, n_2, k)$ and variance $\hat{\sigma}^2{}_g(n_1, n_2, k)$ are computed within the support $S_{n_1,n_2,k}$ as

$$\hat{\mu}_g(n_1, n_2, k) = \frac{1}{L} \sum_{(i_1, i_2; \ell) \in \mathcal{S}_{n_1, n_2, k}} g(i_1, i_2; \ell) \tag{47}$$

and

$$\hat{\sigma}_g^2(n_1, n_2, k) = \frac{1}{L} \sum_{(i_1, i_2; \ell) \in \mathcal{S}_{n_1, n_2, k}} [g(i_1, i_2; \ell) - \hat{\mu}_g(n_1, n_2, k)]^2 \tag{48}$$

and L is the number of pixels in $S_{n_1,n_2,k}$. Then

$$\hat{\sigma}_s^2(n_1, n_2, k) = \max[0, \hat{\sigma}_g^2(n_1, n_2, k) - \hat{\sigma}_v^2(n_1, n_2, k)] \tag{49}$$

in order to avoid the possibility of a negative variance estimate. Depending on whether we use spatiotemporal or temporal statistics, this filter will be referred to as the LMMSE-ST or the LMMSE-T filter, respectively.

The adaptive nature of the filter can be observed from Eq. (46). When the spatiotemporal signal variance is much smaller than the noise variance, $\hat{\sigma}{>}^2{}_s(n_1, n_2, k) \approx 0$, that is, the support $S_{n_1,n_2,k}$ is uniform, the estimate approaches the spatio-temporal mean, $\hat{\mu}_g = \hat{\mu}_s$. In the extreme, when the spatio-temporal signal variance is much larger than the noise variance, $\hat{\sigma}^2{}_s(n_1, n_2, k) \gg \hat{\sigma}^2{}_v(n_1, n_2, k)$, due to poor motion estimation or the presence of sharp spatial edges in $S_{n_1,n_2,k}$, the estimate approaches the noisy image value to avoid blurring.

A drawback of the adaptive LMMSE filter is that it turns the filtering down even if there is a single outlier pixel in the filter support, thus often leaving noisy spots in the filtered image. An alternative implementation, called the *switched LMMSE filter* (*LMMSE-SW*), may maintain the advantages of both the spatiotemporal and the temporal LMMSE filtering by switching between a selected set of temporal and spatiotemporal supports at each pixel, depending on which support is the most uniform (64). If the variance $\hat{\sigma}^2{}_g(n_1, n_2, k)$ computed over $S_{n_1,n_2,k}$ is less than the noise variance, then the filtering is performed using this support; otherwise, the largest support over which $\hat{\sigma}^2{}_g(n_1, n_2, k)$ is less than the noise variance is selected. In the next section, we introduce an adaptive weighted averaging (*AWA*) filter, which employs an implicit mechanism for selecting the most uniform subset within $S_{n_1,n_2,k}$ for filtering.

Adaptive Weighted Averaging Filter. The adaptive weighted averaging (AWA) filter computes a weighted average of the image values within the spatiotemporal support along the motion trajectory. The weights are determined by optimizing a criterion functional, and they vary with the accuracy of motion estimation as well as the spatial uniformity of the region around the motion trajectory. In the case of sufficiently accurate motion estimation across the entire trajectory and spatial uniformity, image values within the spatiotemporal filter support attain equal weights, and the AWA filter performs direct spatiotemporal averaging. When the value of a certain pixel within the spatiotemporal support deviates from the value of the pixel to be filtered by more than a threshold, its weight decreases, shifting the emphasis to the remaining image values within the support that better matches the pixel of interest. The AWA filter is therefore particularly well suited for efficient filtering of sequences containing segments with varying scene contents due, for example, to rapid zooming and changes in the view of the camera.

The AWA filter can be defined by

$$\hat{s}(n_1, n_2, k) = \sum_{(i_1, i_2; \ell) \in \mathcal{S}_{n_1, n_2, k}} w(i_1, i_2: \ell) g(i_1, i_2: \ell) \qquad (50)$$

where

$$w(i_1, i_2: \ell) \doteq \frac{K(n_1, n_2, k)}{1 + a \max\{\epsilon^2, [g(n_1, n_2, k) - g(i_1, i_2: \ell)]^2\}} \qquad (51)$$

are the weights within the support $S_{n_1,n_2,k}$ and $K(n_1, n_2, k)$ is a normalization constant, given by

$$K(n_1, n_2, k) = \left(\sum_{(i_1, i_2; \ell) \in \mathcal{S}_{n_1, n_2, k}} \times \frac{1}{1 + a \max\{\epsilon^2, [g(n_1, n_2, k) - g(i_1, i_2: \ell)]^2\}} \right)^{-1} \qquad (52)$$

The quantities a ($a > 0$) and ϵ are the parameters of the filter. These parameters are determined according to the following principles:

(1) When the differences in the intensities of pixels within the spatiotemporal support are merely due to noise, it is desirable that the weighted averaging reduces to direct averaging. This can be achieved by appropriately selecting the parameter ϵ^2. Note that, if the square of the differences are less than ϵ^2, then all the weights attain the same value $K/(1 + a\epsilon^2) = 1/L$, and $\hat{s}(n_1, n_2, k)$ reduces to direct averaging. We set the value of ϵ^2 equal to two times the value of the noise variance, i.e., the expected value of the square of the difference between two image values that differ due to the presence of noise only.

(2) If the square of the difference between the values $g(n_1, n_2, k)$ and $g(i_1, i_2; l)$ for a particular $(i_1, i_2, l) \in S_{n_1,n_2,k}$ is larger than ϵ^2, then the contribution of $g(i_1, i_2; l)$ is weighted down by $w(i_1, i_2; l) < w(n_1, n_2, k) = K/(1 + a\epsilon^2)$. The parameter a is a "penalty" parameter that determines the sensitivity of the weight to the squared difference $[g(n_1, n_2, k) - g(i_1, i_2; l)]^2$. The "penalty" parameter a is usually set equal to unity.

The effect of the penalty parameter a on the performance of the AWA filter can be best visualized considering a special case, where one of the frames within the filter support is substantially different from the rest. In the extreme, when $a = 0$, all weights are equal. That is, there is no penalty for a mismatch, and the AWA filter performs direct averaging. However, for a large, the weights for the $2M$ "matching" frames are equal, whereas the weight of the nonmatching frame approaches zero. Generally speaking, the AWA filter takes the form of a "limited-amplitude averager," where those pixels whose intensities differ from that of the center pixel by not more than $\pm\epsilon$ are averaged. A similar algorithm is K nearest neighbor averaging (65), where the average of K pixels within a certain window whose values are closest to the value of the pixel of interest are computed.

Image Sequence Restoration. In addition to being contaminated by noise, digital images and video may also be blurred due to out-of-focus imaging systems, relative motion between the scene and camera, atmospheric turbulence, and so on. Filtering for image enhancement and restoration has received significant attention over the last three decades. Image enhancement generally refers to methods that aim to obtain visually pleasing images, including elimination of noise and sharpening of image detail, without mathematical modeling of the image formation and degradation processes. The reader is referred to Ref. 43 for a detailed discussion of image-enhancement techniques such as contrast adjustment by histogram equalization and unsharp masking, and edge-detection methods.

The goal of image sequence restoration is to estimate each image (frame or field) as it would appear without any degradations, by first modeling the degradation process, and then applying an inverse procedure. Unlike image-enhancement algorithms, every image-restoration/superresolution algorithm is based on an observation model, which relates the observed degraded image(s) to the desired "ideal" image, and possibly a regularization model, which conveys the available a priori information about the ideal image. The success of image restoration depends on how good the assumed mathematical models fits the actual application. Image-restoration algorithms can also be classified as intraframe and multiframe restoration methods. A concise discussion of early results on intraframe image restoration can be found in the books by Andrews and Hunt (66) and Gonzalez and Woods (67). More recent developments are summarized in the review papers by Meinel (68), Demoment (69), Sezan and Tekalp (70), and Kaufman and Tekalp (71). For most cases, intraframe/intrafield restoration may be sufficient, although there are cases where multiframe methods provide some advantages. The following covers a summary of most popular restoration methods.

Observation Model. The observed blurred and noisy image can be modeled as

$$g = s(Df) + v \tag{53}$$

where $\boldsymbol{g}$, $\boldsymbol{f}$, and $\boldsymbol{v}$ denote vectors representing lexicographical ordering of the samples of the observed image, ideal image, and a particular realization of the additive (random) noise process, respectively. The operator $\boldsymbol{D}$ is called the *blur operator.* The response of the image sensor to light intensity is represented by the memoryless mapping $s(\cdot)$, which is, in general, nonlinear. This nonlinearity has often been ignored in the literature for algorithm development. The noise is generally approximated by a zero-mean, white Gaussian random field, which is additive and independent of the image signal. In fact, it has been widely acknowledged that more sophisticated noise models do not, in general, lead to significantly improved restorations.

The blur may be space-invariant or space-variant. For space-invariant blurs, $\boldsymbol{D}$ becomes a convolution operator, which has block-Toeplitz structure; and Eq. (53) can be expressed, in scalar form, as

$$g(n_1, n_2) = s\left(\sum_{(m_1,m_2)\in S_d} d(m_1, m_2) f(n_1 - m_1, n_2 - m_2)\right) + v(n_1, n_2) \tag{54}$$

where $d(m_1, m_2)$ and S_d denote the kernel and support of the operator $\boldsymbol{D}$, respectively. The kernel $d(m_1, m_2)$ is the impulse response of the blurring system, often called the *point spread function* (*PSF*). In case of space-variant blurs, the operator $\boldsymbol{D}$ does not have a particular structure; and the observation equation can be expressed as a superposition summation

$$g(n_1, n_2) = s\left(\sum_{(m_1,m_2)\in S_d(n_1,n_2)} d(n_1, n_2{:}\, m_1, m_2) f(m_1, m_2)\right) + v(n_1, n_2) \tag{55}$$

where $S_d(n_1, n_2)$ denotes the support of the PSF at the pixel location (n_1, n_2).

Multiframe observation models can be obtained by stacking a number of single frame/field observation vectors. The resulting multiframe blur operators generally have block-diagonal structures, which facilitate development of fast multiframe restoration algorithms (72).

Blur Identification. Blur identification, which refers to estimation of both the support and parameters of the PSF $\{d(n_1, n_2) : (n_1, n_2) \in S_d\}$ from one or more frames of the blurred video, is often the first step in image restoration, since this information is generally unknown. It is also a crucial step, because quality of restored images are highly sensitive to the accuracy of the estimated PSF (73). An early approach to blur identification has been based on the assumption that the original scene contains an ideal point source, and that its spread (hence the PSF) can be determined from the observed image (74). These approaches are of limited use in practice, because a scene may not contain an ideal point or line source and the observation noise may not allow the measurement of a useful spread.

Physical modeling allows certain types of PSF (e.g., out-of-focus and motion blur PSF), be represented by parametric models with a few parameters. Further, these parameters can be easily identified in the Fourier domain. Spectral and cepstral (Fourier transform of the logarithm of the power spectrum) analysis have been successfully applied in many cases to identify out-of-focus and motion blurs from the location of the zero-crossings of the power spectrum of a blurred image or negative spikes in the cepstrum (75,76). Alternatively, Chang et al. (77) proposed a bispectrum analysis method, which is motivated by the fact that bispectrum is not affected, in principle, by the observation noise. However, the bispectral method requires much more data than the method based on the power spectrum. Note that PSFs which do not have zero crossings in the frequency domain (e.g., Gaussian PSF modeling atmospheric turbulance) cannot be identified by these techniques. An alternative approach is the maximum likelihood (*ML*) PSF estimation. The ML approach aims to find those parameter values (including, in principle, the observation noise variance) which have most likely resulted in the observed image(s). Different implementations of the ML image and blur identification are discussed under a unifying framework (78). Later, Pavlović and Tekalp (79) proposed a practical method to find the ML estimates of the parameters of a PSF, based on a continuous domain image formation model.

In multi-frame image restoration, blur identification using more than one frame at a time becomes possible. For example, the PSF of a possibly space-varying motion blur can be computed at each pixel from an estimate of the frame-to-frame motion vector at that pixel, provided that the shutter speed of the camera is known (80).

Intraframe Space-Invariant Restoration. When the mapping $s(.)$ is ignored, it is evident from Eq. (53) that image restoration reduces to solving a set of simultaneous linear equations. If the matrix $\boldsymbol{D}$ is nonsingular (i.e., $\boldsymbol{D}^{-1}$ exists) and the vector $\boldsymbol{g}$ lies in the column space of $\boldsymbol{D}$ (i.e., there is no observation noise), then there exists a unique solution which can be found by direct inversion (also known as inverse filtering). In practice, however, we almost always have an underdetermined (due to boundary truncation problem) (73) and inconsistent (due to observation noise) set of equations. In this case, we report to a minimum-norm least-squares solution. A least-squares (*LS*) solution (not unique when the columns of $\boldsymbol{D}$ are linearly dependent) minimizes the norm-square of the residual

$$J_{LS}(f) \doteq \|g - Df\|^2 \tag{56}$$

Least-squares solution(s) with the minimum norm (energy) is (are) generally known as *pseudo-inverse solution(s)* (*PIS*).

Restoration by pseudo-inversion is often ill-posed, owing to the presence of observation noise (73). This follows because the pseudo-inverse operator usually has some very large eigenvalues. For example, a typical blur transfer function has zeros; and thus, its pseudo-inverse attains very large magnitudes near these singularities as well as at high frequencies. This results in excessive amplification at these frequencies in the sensor noise. Regularized inversion techniques [e.g., the singular value decomposition method (1)], attempt to roll-off the transfer function of the pseudo-inverse filter at these frequencies to limit noise amplification. It follows that the regularized inverse deviates from the pseudo-inverse at these frequencies, which leads to other types of artifacts, generally known as *regularization artifacts* (73). Popular strategies for regularized inversion (and how to achieve the right amount of regularization) are summarized in the following.

Iterative Filtering (Landweber Iterations). Several image-restoration algorithms are based on variations of the so-called Landweber iterations (81,82,83,84,85,86).

$$\boldsymbol{f}_{k+1} = \boldsymbol{f}_k + \boldsymbol{R}\boldsymbol{D}^T(\boldsymbol{g} - \boldsymbol{D}\boldsymbol{f}_k) \tag{57}$$

where $\boldsymbol{R}$ is a matrix that controls the rate of convergence of the iterations. There is no general way to select the best $\boldsymbol{C}$ matrix. If the system Eq. (53) is nonsingular and consistent (hardly ever the case), the iterations Eq. (57) will converge to the solution. If, on the other hand, Eq. (53) is underdetermined and/or inconsistent, then Eq. (57) converges to a minimum-norm least squares solution, also known as the pseudo-inverse solution (*PIS*). The theory of this and other closely related algorithms are discussed by Sanz and Huang (82) and Tom et al. (83). Kawata and Ichioka (84) are among the first to apply the Landweber-type iterations to image restoration, which they refer as "reblurring" method.

Landweber-type iterative restoration methods can be regularized by appropriately terminating the iterations before convergence, since the closer we are to the pseudo-inverse, the more the noise amplification. A termination rule can be defined on the basis of the norm of the residual image signal (87). Alternatively, soft and/or hard constraints can be incorporated into iterations to achieve regularization. The constrained iterations can be written as (86,88)

$$\boldsymbol{f}_{k+1} = \boldsymbol{C}[\boldsymbol{f}_k + \boldsymbol{R}\boldsymbol{D}^T(\boldsymbol{g} - \boldsymbol{D}\boldsymbol{f}_k)] \tag{58}$$

where $\boldsymbol{C}$ is a nonexpansive constraint operator, i.e., $\|\boldsymbol{C}(\boldsymbol{f}_1) - \boldsymbol{C}(\boldsymbol{f}_2)\| \le \|\boldsymbol{f}_1 - \boldsymbol{f}_2\|$, to guarantee the convergence of the iterations. Application of Eq. (58) to image restoration has been extensively studied (see Refs. 85 and 86, and the references therein).

Constrained Least-Squares Filtering. Regularized image restoration can be formulated as a constrained optimization problem, where a functional $\|\boldsymbol{Q}(\boldsymbol{f})\|^2$ of the image is minimized subject to the constraint $\|\boldsymbol{g} - \boldsymbol{Df}\|^2 = \sigma^2$. Here σ^2 is a constant, which is usually set equal to the variance of the observation noise. The constrained least squares (*CLS*) estimate minimizes the Lagrangian (89)

$$J_{CLS}(\boldsymbol{f}) = \|\boldsymbol{Q}(\boldsymbol{f})\|^2 + \alpha(\|\boldsymbol{g} - \boldsymbol{Df}\|^2 - \sigma^2) \qquad (59)$$

where α is the Lagrange multiplier. The operator $\boldsymbol{Q}$ is chosen such that the minimization of Eq. (59) enforces some desired property of the ideal image. For instance, if $\boldsymbol{Q}$ is selected as the Laplacian operator, smoothness of the restored image is enforced. The CLS estimate can be expressed, by taking the derivative of Eq. (59) and setting it equal to zero, as (66)

$$\hat{\boldsymbol{f}} = (\boldsymbol{D}^H\boldsymbol{D} + \gamma\boldsymbol{Q}^H\boldsymbol{Q})^{-1}\boldsymbol{D}^H\boldsymbol{g} \qquad (60)$$

where ^{H} stands for Hermitian (i.e., complex-conjugate and transpose). The parameter $\gamma = 1/\alpha$ (the regularization parameter) must be such that the constraint $\|\boldsymbol{g} - \boldsymbol{Df}\|^2 = \sigma^2$ is satisfied. It is often computed iteratively (70). A sufficient condition for the uniqueness of the CLS solution is that $\boldsymbol{Q}^{-1}$ exists. For space-invariant blurs, the CLS solution can be expressed in the frequency domain as (89)

$$\hat{F}(u,v) = \frac{D^*(u,v)}{|D(u,v)|^2 + \gamma|L(u,v)|^2}G(u,v) \qquad (61)$$

where $*$ denotes complex conjugation. Another well-known approach that employs constrained optimization is the maximum entropy method (90,91,92,93).

Statistical (Wiener) Filtering. Statistical estimation methods, such as linear minimum mean square error (LMMSE) or maximum a posteriori probability (*MAP*) estimation, have also been applied to image restoration. The LMMSE method finds the linear estimate which minimizes the mean square error between the estimate and ideal image, using up to second-order statistics of the ideal image. Assuming that the ideal image can be modeled by a zero-mean homogeneous random field and the blur is space-invariant, the LMMSE estimate, in the frequency domain, is given by (1)

$$\hat{F}(u,v) = \frac{D^*(u,v)}{|D(u,v)|^2 + \sigma_v^2/|P(u,v)|^2}G(u,v) \qquad (62)$$

where $\sigma^2{}_v$ is the variance of the observation noise (assumed white) and $|P(u,v)|^2$ stands for the power spectrum of the ideal image. This estimator is commonly known as the *Wiener filter.* The power spectrum of the ideal image is usually estimated from a prototype. It can be easily seen that the CLS estimate Eq. (61) reduces to the Wiener estimate by setting $|L(u,v)|^2 = \sigma^2{}_v/|P(u,v)|^2$ and $\gamma = 1$.

A Kalman filter determines the causal (up to a fixed lag) LMMSE estimate recursively. It is based on a state-space representation of the image and observation models. In the first step of Kalman filtering, a prediction of the present state is formed using an autoregressive (*AR*) image model and the previous state of the system. In the second step, the predictions are updated on the basis of the observed image data to form the estimate of the present state. Application of 2-D Kalman filtering to image restoration has been studied in Refs. 71 and 94,95,96.

The MAP restoration, maximizes the a posteriori probability density function, i.e., the likelihood of a given estimate being the same as the ideal image given the observed degraded image data. Trussel and Hunt (97) used nonstationary a priori pdf models, and proposed a modified form of the Picard iteration to solve the nonlinear maximization problem. They suggested using the variance of the residual signal as a criterion for convergence. Geman and Geman (98) proposed using a Gibbs random field model for the a priori pdf of the ideal image. They used simulated annealing procedures for the maximization. It should be noted that the MAP procedures usually require significantly more computation compared with, for example, the CLS or Wiener solutions.

Set-Theoretic Methods. In set-theoretic methods, first, a number of "constraint sets" are defined such that their members are consistent with the observations and/or some a priori information about the ideal image. A set-theoretic estimate of the ideal image is, then, defined as a feasible solution satisfying all constraints, i.e., any member of the intersection of the constraint sets.

Set-theoretic methods vary according to the mathematical properties of the constraint sets. In the method of projections onto convex sets (*POCS*), the constraint sets C_i are closed and convex in an appropriate Hilbert space H. Given the sets C_i, $i = 1, \ldots, M$, and their respective projection operators $\mathbf{P}_i$, a feasible solution is found by performing successive projections as

$$f_{k+1} = \mathbf{P}_M \mathbf{P}_{M-1} \ldots \mathbf{P}_1 f_k; \qquad k = 0, 1, \ldots \tag{63}$$

where $\boldsymbol{f}_0$ is the initial estimate (a point in H). The projection operators are usually found by solving constrained optimization problems. In finite-dimensional problems (which is the case for digital image restoration), the iterations converge to a feasible solution in the intersection set (99,100,101). It should be noted that the convergence point is affected by the choice of the initialization. However, as the size of the intersection set gets smaller, the differences between the convergence points obtained by different initializations become smaller. Trussell and Civanlar (102) applied POCS to image restoration. For examples of convex constraint sets that are used in image restoration see Ref. 103. A relationship between the POCS and Landweber iterations were developed in Ref. 97. A special case of POCS is the Gerchberg-Papoulis type algorithms, where the constraint sets are either linear subspaces or linear varieties (104).

Restoration of Images Recorded by Nonlinear Sensors. Image sensors and media may have nonlinear characteristics that can be modeled by a pointwise (memoryless) nonlinearity $s(.)$. Common examples are photographic film and paper, where the nonlinear relationship between the exposure (intensity) and the silver density deposited on the film or paper is specified by a "$d - \log e$" curve. The modeling of sensor nonlinearities was first addressed by Andrews and Hunt (66). However, it was not generally recognized that results obtained by taking the sensor nonlinearity into account may be far more superior to those obtained by ignoring the sensor nonlinearity, until the experimental work of Tekalp and Pavlovic (105,106). Their results show that accounting for the sensor nonlinearity may dramatically improve restoration results (105,106).

Intraframe Restoration of Space-Varying Blurred Images. Implementation of some of the above algorithms becomes computationally formidable in the case of space-varying PSFs, because Fourier transforms cannot be used to simplify large matrix operations (such as inversion or singular value decomposition). There are three practical approaches for space-variant image restoration: (1) sectioning; (2) coordinate transformation; and (3) adaptive filtering.

Sectioning methods assume that a blurred image with a space-varying PSF can be restored by applying space-invariant filters to predefined local image regions (80,107). A drawback of sectioning methods is generation of artifacts at the region boundaries. Overlapping of regions somewhat reduces these artifacts, but does not completely avoid them. Most space-varying PSF vary continuously from pixel to pixel; thus, violating the basic premise of the sectioning methods. To this effect, Robbins and Huang (108) and Sawchuck (109) proposed a coordinate transformation (*CTR*) method such that the blur PSF in the transformed coordinates is space-invariant. Then, the transformed image can be restored by a space-invariant filter and then transformed back to obtain the final restored image. However, the CTR method is applicable to a limited class of space-varying blurs.

The lack of generality of sectioning and CTR methods motivates adaptive approaches, such as iterative methods, Kalman filtering, and set-theoretic methods, which can be applied to space-varying de-blurring in a computationally feasible manner. Angel and Jain (110) propose solving the superposition Eq. (55) iteratively, using a conjugate gradient method. Application of constrained iterative methods was discussed in Ref. 88. More recently, Ozkan et al. (111) developed a robust POCS algorithm for space-varying image restoration, where they defined a closed, convex constraint set for each observed blurred image pixel (n_1, n_2), given by:

$$C_{n_1,n_2} = \{y: |r^{(y)}(n_1, n_2)| \le \delta_0\} \qquad (64)$$

and

$$r^{(y)}(n_1, n_2) \doteq g(n_1, n_2) - \sum_{(m_1,m_2)\in \mathcal{S}_d(n_1,n_2)} d(n_1, n_2: m_1, m_2) y(m_1, m_2) \qquad (65)$$

is the residual at pixel (n_1, n_2) associated with $\boldsymbol{y}$, which denotes an arbitrary member of the set. The quantity δ_0 is an a priori bound reflecting the statistical confidence with which the actual image is a member of the set C_{n_1,n_2}. Since $r^{(f)}(n_1, n_2) = v(n_1, n_2)$, the bound δ_0 is determined from the statistics of the noise process so that the ideal image is a member of the set within a certain statistical confidence. The collection of bounded residual constraints over all pixels (n_1, n_2) enforce the estimate to be consistent with the observed image. The projection operator onto this set can be found in Ref. 111. The algorithm starts with an arbitrary initial estimate, and successively projects onto each C_{n_1,n_2} until convergence. Additional constraints, such as bounded energy, amplitude, and limited support, can be utilized to improve the results.

Multiframe Image Restoration. The sequential nature of images in a video source can be used to better estimate the PSF parameters, regularization terms, and the restored image. The first multiframe restoration filter was the motion-compensated multiframe Wiener filter (*MCMF*) proposed by Ozkan et al. (72), who considered the case of frame-to-frame global translations. Then, the auto power spectra of all frames are the same and the cross spectra are related by a phase factor, which can be estimated from the motion information. Multiframe restoration of a group of pictures provides a significant improvement over intraframe restoration of individual frames sequentially if the blur PSF varies from frame to frame while the scene content stays relatively unchanged (except for small displacements). This is because each PSF has then slightly different zero-crossing locations, and frequencies which are zeroed-out in a frame can be reconstructed by those of other frames.

High-Resolution Filtering. Most electronic cameras have limited spatial resolution determined by the characteristics of the sensor array, resulting in blurring and/or aliasing. Recall that the highest spatial frequency that can be represented is one half of the sampling (Nyquist) frequency. Superresolution refers to estimating an image at a resolution higher than that of the imaging sensor. Recently, printing high-quality still images from video sources has become an important application for multi-frame restoration and superresolution methods.

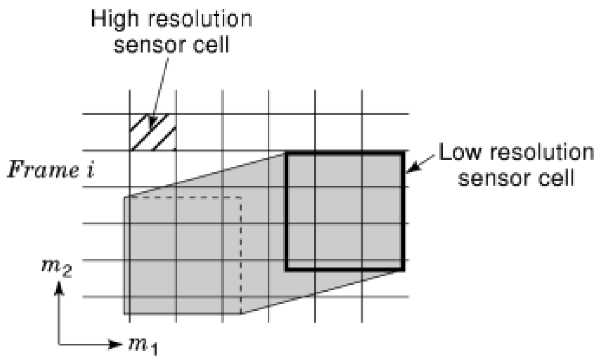

Fig. 24. Illustration of the discrete system PSF.

Modeling. Each observed low-resolution image (frame or field) can be related to the desired high-resolution ideal still-image through the superposition summation (1)

$$g_k(n_1, n_2) = s\left(\sum_{(m_1,m_2)\in\mathcal{S}_d(n_1,n_2;k)} d_k(n_1, n_2 : m_1, m_2) f(m_1, m_2)\right) + v_k(n_1, n_2) \quad (66)$$

where the support of the summation over the high-resolution grid (m_1, m_2) at a particular observed pixel (n_1, n_2; k) depends on the motion trajectory connecting the pixel (n_1, n_2; k) to the ideal image, the size of the support of the low-resolution sensor PSF $h_a(x_1, x_2)$ with respect to the high-resolution grid, and whether there is additional optical (out-of-focus, motion, etc.) blur. Because the relative positions of low- and high-resolution pixels in general vary from pixel to pixel, the discrete sensor PSF is space-varying. The support of the space-varying PSF is indicated by the shaded area in Fig. 24, where the rectangle depicted by solid lines shows the support of a low-resolution pixel over the high-resolution sensor array. The shaded region corresponds to the area swept by the low-resolution pixel due to motion during the aperture time (1).

In summary, the model states that each observed pixel (n_1, n_2; k) can be expressed as a linear combination of several desired high-resolution pixels (m_1, m_2), such that (n_1, n_2; k) is connected to (m_1, m_2) by a sub-pixel motion trajectory. Note that this model is invalid in case of occlusion. We assume that occlusion regions can be detected a priori using a proper motion estimation/segmentation algorithm.

The Principle. When the interframe motion is subpixel, each frame contains some “new” information, that can be utilized to achieve superresolution. Superresolution refers to high-resolution image expansion, which aims to remove aliasing artifacts, blurring due to sensor PSF, and optical blurring given the observation model Eq. (66). Provided that enough frames with accurate subpixel registration are available, the observation model becomes invertible. It can be easily seen, however, that superresolution from a single observed image is ill-posed, since we have more unknowns than equations, and there exist infinitely many expanded images which are consistent with the model Eq. (66). Therefore, single-frame nonlinear interpolation (also called *image expansion* and *digital zooming*) methods for improved definition image expansion employ additional regularization criteria, such as edge-preserving smoothness constraints (112,113). (It is well known that no new high-frequency information can be generated by LSI interpolation techniques, including ideal band-limited interpolation, hence the need for nonlinear methods.)

Methods. Motion-compensated (multiframe) superresolution methods that are based on the model Eq. (66) can be classified as those which aim to eliminate (1) aliasing only, (2) aliasing and LSI blurs, and (3) aliasing and space-varying blurs. In addition, some of these methods are designed for global translational motion only, while others can handle space-varying motion fields with occlusion. Multiframe superresolution was first introduced by Tsai and Huang (114), who exploited the relationship between the continuous and discrete Fourier transforms of the undersampled frames to remove aliasing errors, in the special case of global motion. Their formulation has been extended by Kim et al. (115) to take into acocunt noise and blur in the low-resolution images, by posing the problem in the least-squares sense. A further refinement by Kim and Su (116) allowed blurs that are different for each frame of low-resolution data, by using a Tikhonov regularization. However, the resulting algorithm did not treat the formation of blur due to motion or sensor size, and suffers from convergence problems.

Inspection of the model Eq. (66) suggests that the superresolution problem can be stated in the spatio-temporal domain as the solution of a set of simultaneous linear equations. Suppose that the desired high-resolution frames are $M \times M$, and we have L low-resolution observations, each $N \times N$. Then, from Eq. (66), we can set up at most $L \times N \times N$ equations in M^2 unknowns to reconstruct a particular high-resolution frame. These equations are linearly independent, provided that all displacements between the successive frames are at subpixel amounts. (Clearly, the number of equations will be reduced by the number of occlusion labels encountered along the respective motion trajectories.) In general, it is desirable to set up an overdetermined system of equations, i.e., $L > R^2 = M^2/N^2$, to obtain a more robust solution in the presence of observation noise. Because the impulse response coefficients $h_{ik}(n_1, n_2; m_1, m_2)$ are spatially varying, and hence the system matrix is not block-Toeplitz, fast methods to solve them are not available. Stark and Oskoui (117) proposed a POCS method to compute a high-resolution image from observations obtained by translating and/or rotating an image with respect to a CCD array. Irani et al. (22,23,118) employed iterative methods. Patti et al. (119) extended the POCS formulation to include sensor noise and space-varying blurs. Bayesian approaches were also employed for superresolution (120). The extension of the POCS method with space-varying blurs is explained in the following.

The POCS solution described here addresses the most general form of the superresolution problem based on the model Eq. (66). The formulation is quite similar to the POCS approach presented for intraframe restoration of space-varying blurred images. In this case, we define a different closed, convex set for each observed low-resolution pixel (n_1, n_2, k) (which can be connected to the desired frame i by a motion trajectory) as

$$C_{n_1,n_2;i,k} = \{x_i(m_1, m_2): |r_k^{(\mathbf{x}_i)}(n_1, n_2)| \le \delta_0\}, \qquad 0 \le n_1, n_2 \le N-1, \qquad k = 1, \ldots, L \quad (67)$$

where

$$r_k^{(\mathbf{x}_i)}(n_1, n_2) \doteq g_k(n_1, n_2) - \sum_{m_1=0}^{M-1} \sum_{m_2=0}^{M-1} x_i(m_1, m_2) h_{ik}(m_1, m_2: n_1, n_2)$$

and δ_0 represents the confidence that we have in the observation and is set equal to $c\sigma_v$, where σ_v is the standard deviation of the noise and $c \ge 0$ is determined by an appropriate statistical confidence bound. These sets define high-resolution images which are consistent with the observed low-resolution frames within a confidence bound that is proportional to the variance of the observation noise. The projection operator which projects onto $C_{n_1,n_2;i,k}$ can be deduced from Eq. (67) (1). Additional constraints, such as amplitude and/or finite support constraints, can be utilized to improve the results. Excellent reconstructions have been reported using this procedure (72,119).

A few observations about the POCS method are in order: (1) While certain similarities exist between the POCS iterations and the Landweber-type iterations (22,118), the POCS method can adapt to the amount of the observation noise, while the latter generally cannot. (2) The POCS method finds a feasible solution, that is, a solution consistent with all available low-resolution observations. Clearly, the more observations (more frames with reliable motion estimation) we have, the better the high-resolution reconstructed image $\hat{s}(m_1, m_2)$ will be. In general, it is desirable that $L > M^2/N^2$. Note however that, the POCS method generates a reconstructed image with any number L of available frames. The number L is just an indicator of how large the feasible set of solutions will be. Of course, the size of the feasible set can be further reduced by employing other closed, convex constraints in the form of statistical or structural image models.

Acknowledgments

The author acknowledges his colleagues M. I. Sezan and A. T. Erdem, and his former Ph.D. students G. Pavlovic, M. K. Ozkan, A. Patti, M. Chang, Y. Altunbasak, and C. Toklu for their influence on this work. Prof. Tekalp was supported by grants from NSF, New York State Science and Technology Foundation, and Eastman Kodak Co. during the course of this work.

BIBLIOGRAPHY

1. A. M. Tekalp *Digital Video Processing*, Upper Saddle River, NJ: Prentice Hall, 1995,
2. A. N. Tikhonov V. Y. Arsenin *Solutions of Ill-Posed Problems*, Washington, DC: V. H. Winston and Sons 1977.
3. B. K. P. Horn B. G. Schunck Determining optical flow, *Artif. Intell.*, **17**: 185–203, 1981.
4. E. Dubois J. Konrad Estimation of 2-D motion fields from image sequences with application to motion-compensated processing, in M. I. Sezan and R. L. Lagendijk (eds.), *Motion Analysis and Image Sequence Processing*, Norwell, MA: Kluwer, 1993.
5. P. Anandan *et al.* Hierarchical model-based motion estimation, in M. I. Sezan and R. L. Lagendijk (eds.), *Motion Analysis and Image Sequence Processing*, Norwell, MA: Kluwer, 1993.
6. J. K. Aggarwal N. Nandhakumar On the computation of motion from sequences of images, *Proc. IEEE*, **76**: 917–935, 1988.
7. B. D. Lucas T. Kanade An iterative image registration technique with an application to stereo vision, *Proc. DARPA Image Underst. Workshop*, 1981, pp. 121–130.
8. J. D. Robbins A. N. Netravali Recursive motion compensation: A review, in T. S. Huang (ed.), *Image Sequence Processing and Dynamic Scene Analysis*, Berlin: Springer-Verlag, 1983, pp. 76–103.
9. D. R. Walker K. R. Rao Improved pel-recursive motion compensation, *IEEE Trans. Commun.*, **COM-32**: 1128–1134, 1984.
10. C. Cafforio F. Rocca The differential method for image motion estimation, in T. S. Huang (ed.), *Image Sequence Processing and Dynamic Scene Analysis*, Berlin: Springer-Verlag, 1983, pp. 104–124.
11. V. Seferidis M. Ghanbari General approach to block-matching motion estimation, *Opt. Eng.*, **32**: 1464–1474, 1993.
12. C. Toklu *et al.* Tracking motion and intensity-variations using hierarchical 2-D mesh modeling for synthetic object transfiguration, *Graphical Models Image Process.*, **58** (6): 553–573, 1996.
13. Y. Altunbasak A. M. Tekalp Closed-form connectivity-preserving solutions for motion compensation using 2-D meshes, *IEEE Trans. Image Process.*, 1997.
14. Y. Nakaya H. Harashima Motion compensation based on spatial transformations, *IEEE Trans. Circuits Syst. Video Technol.*, **4**: 339–356, 1994.
15. D. J. Heeger Optical flow using spatiotemporal filters, *Int. J. Comput. Vision*, **1**: 279–302, 1988.

16. J. L. Barron D. J. Fleet S. S. Beauchemin Systems and experiment: Performance of optical flow techniques, *Int. J. Comput. Vision*, **12** (1): 43–77, 1994.
17. D. J. Fleet A. D. Jepson Computation of component image velocity from local phase information, *Int. J. Comput. Vision*, **5**: 77–104, 1990.
18. M. Bierling Displacement estimation by hierarchical block-matching, *Proc. SPIE*, **1001**: 942–951, 1988.
19. T. Aach A. Kaup R. Mester Statistical model-based change detection in moving video, *Signal Process.*, **31**: 165–180, 1993.
20. M. Hoetter R. Thoma Image segmentation based on object oriented mapping parameter estimation, *Signal Process.*, **15**: 315–334, 1988.
21. J. R. Bergen *et al.* Dynamic multiple-motion computation, *Artif. Intell. and Comput. Vision*, 1991.
22. M. Irani S. Peleg Motion analysis for image enhancement: Resolution, occlusion and transparency, *J. Visual Commun. Image Rep.* **4**: 324–335, 1993.
23. M. Irani *et al.* Efficient representation of video sequences and their applications, *Signal Process.: Image Commun.*, **8**: 327–351, 1996.
24. J. Y. A. Wang E. Adelson Representing moving images with layers, *IEEE Trans. Image Process.*, **3**: 625–638, 1994.
25. G. Adiv Determining three-dimensional motion and structure from optical flow generated by several moving objects, *IEEE Trans. Pattern Anal. Mach. Intell.*, **7**: 384–401, 1985.
26. M. Bober J. Kittler On combining the Hough transform and multiresolution MRF's for the robust analysis of complex motion, *Proc. Asian Conf. Comput. Vision* (ACCV), 1995.
27. D. W. Murray B. F. Buxton Scene segmentation from visual motion using global optimization, *IEEE Trans. Pattern Anal. Mach. Intell.*, **9**: 220–228, 1987.
28. Y. Weiss E. H. Adelson A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models, *Proc. IEEE Int. Conf. Comput. Vision Pattern Recognition*, 1996.
29. P. Salembier Morphological multiscale segmentation for image coding, *Signal Process.*, **38** (3): 359–386, 1994.
30. C. Gu T. Ebrahimi M. Kunt Morphological moving object segmentation and tracking for content-based video coding, *Proc. Int. Symp. Multimedia Commun. Video Coding*, New York, 1995.
31. M. M. Chang M. I. Sezan A. M. Tekalp An algorithm for simultaneous motion estimation and scene segmentation, *Proc. ICASSP*, Adelaide, Australia, 1994.
32. F. Dufaux F. Moscheni A. Lippman Spatio-temporal segmentation based on motion and static segmentation, *Proc. IEEE Int. Conf. Image Process.*, **1**: 306–309, 1995.
33. B. Bascle *et al.* Tracking complex primitives in an image sequence, *Int. Conf. Pattern Recognition*, Jerusalem, Israel, 1994, pp. 426–431.
34. M. Kass A. Witkin D. Terzopoulos Snakes: Active contour models, *Int. J. Comput. Vision*, **1** (4): 321–331, 1988.
35. C. Kervrann F. Heitz Robust tracking of stochastic deformable models in long image sequences, *Proc. IEEE Int. Conf. Image Process.*, Austin, TX, 1994.
36. Y. Y. Tang C. Y. Suen New algorithms for fixed and elastic geometric transformation models, *IEEE Trans. Image Process.*, **3**: 355–366, 1994.
37. F. G. Meyer P. Bouthemy Region-based tracking using affine motion models in long image sequences, *CVGIP: Image Understanding*, 60: 119–140, 1994.
38. C. Toklu A. T. Erdem A. M. Tekalp 2-D mesh-based synthetic transfiguration of an object with occlusion, *Proc. IEEE ICASSP*, Munich, 1997.
39. B. G. Haskell A. Puri A. N. Netravali *Digital Video: An Introduction to MPEG-2*, New York: Chapman and Hall, 1997.
40. P. Haavisto Y. Neuvo Motion adaptive scan rate up-conversion, *Multidim. Syst. Signal. Process.*, **3**: 113–130, 1992.
41. M. A. Isnardi *Modeling the television process*, Ph.D. thesis, MIT, Cambridge, MA, 1986.
42. D. M. Martinez *Model-based motion interpolation and its application to restoration and interpolation of motion pictures*, Ph.D. thesis, MIT, Cambridge, MA, 1986.
43. J. S. Lim *Two-Dimensional Signal and Image Processing*, Englewood Cliffs, NJ: Prentice Hall, 1990.
44. G. Schamel Pre- and post-filtering of HDTV signals for sampling rate reduction and display up-conversion, *IEEE Trans. Circuits Syst.*, **34**: 1432–1439, 1987.
45. A. Zaccarin B. Liu Block motion compensated coding of interlaced sequences using adaptively de-interlaced fields, *Signal Process.: Image Commun.*, **5**: 473–485, 1993.
46. T. Reuter Standards conversion using motion compensation, *Signal Process.*, **16**: 73–82, 1989.

47. M. Bierling R. Thoma Motion compensating field interpolation using a hierarchically structured displacement estimator, *Signal Process.*, **11**: 387–404, 1986.
48. S. Tubaro F. Rocca Motion field estimators and their application to image interpolation, in M. I. Sezan and R. L. Lagendijk (eds)., *Motion Analysis and Image Sequence Processing*, Norwell, MA: Kluwer, 1993.
49. R. Thoma M. Bierling Motion compensating interpolation considering covered and uncovered background, *Signal Process.: Image Commun.*, **1**: 191–212, 1989.
50. R. L. Lagendijk M. I. Sezan Motion compensated frame rate conversion of motion pictures, *IEEE Int. Conf. Acoust. Speech, Signal. Process.*, San Francisco, 1992.
51. D. T. Kuan *et al.* Adaptive noise smoothing filter for images with signal-dependent noise, *IEEE Trans. Pattern. Anal. Mach. Intell.*, **PAMI-7**: 165–177, 1985.
52. L. S. Davis A survey of edge-detection techniques, *Comput. Graphics Image Process.*, **4**: 248–270, 1975.
53. P. Chan J. S. Lim One-dimensional processing for adaptive image restoration, *IEEE Trans. Acoust. Speech Signal Process.*, **33**: 117–126, 1985.
54. G. R. Arce N. C. Gallagher T. A. Nodes Median filters: Theory for one or two dimensional filters, *Adv. Comput. Vision Image Process.*, 1986.
55. E. Ataman V. K. Aatre K. M. Wong A fast method for real-time median filtering, *IEEE Trans. Acoust. Speech Signal Process.*, **28**: 415–421, 1980.
56. G. R. Arce Multistage order statistic filters for image sequence processing, *IEEE Trans. Signal Process.*, **39**: 1146–1163, 1991.
57. D. Martinez J. S. Lim Implicit motion compensated noise reduction of motion video scenes, *Proc. IEEE ICASSP*, Tampa, FL, 1985, pp. 375–378.
58. R. H. McMann *et al.* A digital noise reducer for encoded NTSC signals, *SMPTE J.*, **87**: 129–133, 1979.
59. T. J. Dennis Non-linear temporal filter for television picture noise reduction, *IEE Proc.*, **127G**: 52–56, 1980.
60. T. S. Huang Y. P. Hsu Image sequence enhancement, in T. S. Huang (ed.), *Image Sequence Analysis*, Berlin: Springer-Verlag, 1981.
61. R. Samy An adaptive image sequence filtering scheme based on motion detection, *SPIE*, **596**: 135–144, 1985.
62. M. I. Sezan M. K. Ozkan S. V. Fogel Temporally adaptive filtering of noisy image sequences using a robust motion estimation algorithm, *Proc. IEEE Int. Conf. Acoust. Speech, Signal Process.*, Toronto, Can., 1991, pp. 2429–2432.
63. J. S. Lee Digital image enhancement and noise filtering by use of local statistics, *IEEE Trans. Pattern. Anal. Mach. Intell.*, **PAMI-2**: 165–168, 1980.
64. M. K. Ozkan M. I. Sezan A. M. Tekalp Adaptive motion-compensated filtering of noisy image sequences, *IEEE Trans. Circuits Syst. Video Technol.*, **3**: 277–290, 1993.
65. L. S. Davis A. Rosenfeld Noise cleaning by iterated local averaging, *IEEE Trans. Syst. Man Cybern.*, **8**: 705–710, 1978.
66. H. C. Andrews B. R. Hunt *Digital Image Restoration*, Englewood Cliffs, NJ: Prentice Hall, 1977.
67. R. C. Gonzalez R. E. Woods *Digital Image Processing*, Reading, MA: Addison-Wesley, 1992.
68. E. S. Meinel Origins of linear and nonlinear recursive restoration algorithms, *J. Opt. Soc. Amer.* **A-3** (6): 787–799, 1986.
69. G. Demoment Image reconstruction and restoration: Overview of common estimation structures and problems, *IEEE Trans. Acoust. Speech Signal Process.* **37**: 2024–2036, 1989.
70. M. I. Sezan A. M. Tekalp Survey of recent developments in digital image restoration, *Opt. Eng.*, **29**: 393–404, 1990.
71. H. Kaufman A. M. Tekalp Survey of estimation techniques in image restoration, *IEEE Control Syst. Mag.*, **11**: 16–24, 1991.
72. M. K. Ozkan *et al.* Efficient multiframe Wiener restoration of blurred and noisy image sequences, *IEEE Trans. Image Process.*, **1**: 453–476, 1992.
73. A. M. Tekalp M. I. Sezan Quantitative analysis of artifacts in linear space-invariant image restoration, *Multidimens. Syst. Signal Proc.*, **1** (1): 143–177, 1990.
74. A. Rosenfeld A. C. Kak *Digital Picture Processing*, New York: Academic Press, 1982, Vol. 2.
75. D. B. Gennery Determination of optical transfer function by inspection of frequency-domain plot, *J. Opt. Soc. Amer.*, **63** (12): 1571–1577, 1973.
76. M. Cannon Blind deconvolution of spatially invariant image blurs with phase, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-24**: 58–63, 1976.

77. M. M. Chang A. M. Tekalp A. T. Erdem Blur identification using the bispectrum, *IEEE Trans. Signal Process.*, **ASSP-39**: 2323–2325, 1991.
78. R. L. Lagendijk A. M. Tekalp J. Biemond Maximum likelihood image and blur identification: A unifying approach, *Opt. Eng.*, **29** (5): 422–435, 1990.
79. G. Pavlović A. M. Tekalp Maximum likelihood parametric blur identification based on a continuous spatial domain model, *IEEE Trans. Image Process.*, **1**: 496–504, 1992.
80. H. J. Trussell S. Fogel Identification and restoration of spatially variant motion blurs in sequential images, *IEEE Trans. Image Process.*, **1**: 123–126, 1992.
81. H. J. Trussell M. R. Civanlar The Landweber iteration and projection onto convex sets, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-33**: 1632–1634, 1985.
82. J. L. C. Sanz T. S. Huang Unified Hilbert space approach to iterative least-squares linear signal restoration, *J. Opt. Soc. Amer.*, **73** (11): 1455–1465, 1983.
83. V. T. Tom *et al.* Convergence of iterative nonexpansive signal reconstruction algorithms, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-29**: 1052–1058, 1981.
84. S. Kawata Y. Ichioka Iterative image restoration for linearly degraded images. II. Reblurring, *J. Opt. Soc. Amer.*, **70**: 768–772, 1980.
85. A. K. Katsaggelos Iterative image restoration algorithms, *Opt. Eng.*, **28** (7): 735–748, 1989.
86. J. Biemond R. L. Lagendijk R. M. Mersereau Iterative methods for image deblurring, *Proc. IEEE*, 78: 856–883, 1990.
87. H. J. Trussell Convergence criteria for iterative restoration methods, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-31**: 129–136, 1983.
88. R. W. Schafer R. M. Mersereau M. A. Richards Constrained iterative restoration algorithms, *Proc. IEEE*, **69**: 432–450, 1981.
89. B. R. Hunt The application of constrained least squares estimation to image restoration by digital computer, *IEEE Trans. Comput.*, **C-22**: 805–812, 1973.
90. B. R. Frieden Restoring with maximum likelihood and maximum entropy, *J. Opt. Soc. Amer.*, **62** (4): 511–518, 1972.
91. S. F. Gull G. J. Daniell Image reconstruction from incomplete and noisy data, *Nature*, **272**: 686–690, 1978.
92. S. F. Burch S. F. Gull J. Skilling Image restoration by a powerful maximum entropy method, *Comput. Vision Graphics Image Process.*, **23**: 113–128, 1983.
93. R. A. Gonsalves H-M. Kao Entropy-based algorithm for reducing artifacts in image restoration, *Opt. Eng.*, **26** (7): 617–622, 1987.
94. J. W. Woods V. K. Ingle Kalman filtering in two-dimensions-further results, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-29**: 188–197, 1981.
95. D. L. Angwin H. Kaufman Image restoration using reduced order models, *Signal Process.*, **16**: 21–28, 1988.
96. A. M. Tekalp H. Kaufman J. W. Woods Edge-adaptive Kalman filtering for image restoration with ringing suppression, *IEEE Trans. Acoust. Speech Signal Process.*, **37**: 892–899, 1989.
97. H. J. Trussell B. R. Hunt Improved methods of maximum a posteriori restoration, *IEEE Trans. Comput.*, **C-27**: 57–62, 1979.
98. S. Geman D. Geman Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, *IEEE Trans. Pattern Anal. Mach. Intell.*, **6**: 721–741, 1984.
99. D. C. Youla H. Webb Image restoration by the method of convex projections: Part 1. Theory, *IEEE Trans. Med. Imag.*, **MI-1**: 81–94, 1982.
100. M. I. Sezan An overview of convex projections theory and its applications to image recovery problems, *Ultramicroscopy*, **40**: 55–67, 1992.
101. P. L. Combettes The foundations of set-theoretic estimation, *Proc. IEEE*, **81**: 182–208, 1993.
102. H. J. Trussell M. R. Civanlar Feasible solution in signal restoration, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-32**: 201–212, 1984.
103. M. I. Sezan H. J. Trussell Prototype image constraints for set-theoretic image restoration, *IEEE Trans. Signal Process.*, **39**: 2275–2285, 1991.
104. D. C. Youla Generalized image restoration by the method of alternating orthogonal projections, *IEEE Trans. Circuits Syst.*, **CAS-25**: 694–702, 1978.
105. A. M. Tekalp G. Pavlović Image restoration with multiplicative noise: Incorporating the sensor nonlinearity, *IEEE Trans. Signal Process.*, **39**: 2132–2136, 1991.

106. A. M. Tekalp G. Pavlović Digital restoration of images scanned from photographic paper, *J. Electron. Imag.*, **2**: 19–27, 1993.
107. H. J. Trussell B. R. Hunt Image restoration of space-variant blurs by sectioned methods, *IEEE Trans. Acoust. Speech Signal Process.*, **ASSP-26**: 608–609, 1978.
108. G. M. Robbins T. S. Huang Inverse filtering for linear shift-variant imaging systems, *Proc. IEEE*, **60**: 1972.
109. A. A. Sawchuck Space-variant image restoration by coordinate transformations, *J. Opt. Soc. Amer.*, **64** (2): 138–144, 1974.
110. E. S. Angel A. K. Jain Restoration of images degraded by spatially varying point spread functions by a conjugate gradient method, *Appl. Opt.*, **17**: 2186–2190, 1978.
111. M. K. Ozkan A. M. Tekalp M. I. Sezan POCS-based restoration of space-varying blurred images, *IEEE Trans. Image Process.*, **3**: 450–454, 1994.
112. Y. Wang S. K. Mitra Motion/pattern adaptive interpolation of interlaced video sequences, *Proc. IEEE ICASSP*, Toronto, Can., pp. 2829–2832, 1991.
113. R. R. Schultz R. L. Stevenson A Bayesian approach to image expansion for improved definition, *IEEE Trans. Image Process.*, **3**: 233–242, 1994.
114. R. Y. Tsai T. S. Huang Multiframe image restoration and registration, *Adv. Comput. Vision Image Process.*, **1**: 317–339, 1984.
115. S. P. Kim N. K. Bose H. M. Valenzuela Recursive reconstruction of high-resolution image from noisy undersampled frames, *IEEE Trans. Acoust., Speech Signal Process.* **ASSP-38**: 1013–1027, 1990.
116. S. P. Kim W.-Y. Su Recursive high-resolution reconstruction of blurred multiframe images, *IEEE Trans. Image Process.*, **2**: 534–539, 1993.
117. H. Stark P. Oskoui High-resolution image recovery from image plane arrays using convex projections, *J. Opt. Soc. Amer.*, **A6**: 1715–1726, 1989.
118. M. Irani S. Peleg Improving resolution by image registration, *CVGIP: Graphical Models Image Process.*, **53**: 231–239, 1991.
119. A. Patti M. I. Sezan A. M. Tekalp High-resolution image reconstruction from a low-resolution image sequence in the presence of time-varying motion blur, *Proc. IEEE Int. Conf. Image Process.*, Austin, TX, 1994.
120. R. R. Schultz R. L. Stevenson Improved definition video frame enhancement, *Proc. IEEE ICASSP '95*, Detroit, MI, 1995, pp. 2169–2172.

A. MURAT TEKALP
University of Rochester