MCMC for Joint Noise Reduction and Missing Data Treatment in Degraded Video

Anil C. Kokaram, *Member, IEEE*, and Simon J. Godsill, *Member, IEEE*

***Abstract*—Image sequence restoration has been steadily gaining importance with the increasing prevalence of visual digital media. Automated treatment of archived video material typically involves dealing with replacement noise in the form of "blotches" that have varying intensity levels and "grain" noise. In the case of replacement noise, the problem is essentially one of missing data that must be detected and then reconstructed based on surrounding spatio–temporal information, whereas the additive noise can be treated as a noise-reduction problem. It is typical to treat these problems as separate issues; however, it is clear that the presence of noise has an effect on the ability to detect missing data and vice versa. This paper therefore introduces a fully Bayesian specification for the problem that allows an algorithm to be designed that acknowledges and exploits the influences from each of the subprocesses, causing the observed degradation. Markov chain Monte Carlo (MCMC) methodology is applied to the joint detection and removal of both replacement and additive noise components. It can be seen that many of the previous processes presented for noise reduction and missing data treatment are special cases of the framework presented here.**

***Index Terms*—Autoregressive models, Bayesian inference, composition sampling, factored sampling, Gibbs sampling, image processing, marginalization, Markov chain Monte Carlo, missing data reconstruction, motion estimation, noise reduction, video processing.**

I. Introduction

WITHIN the last five years, there has been an explosion in the exploitation and availability of digital visual media. Digital television has been widely available in Europe for the last two years, and Internet usage continues to grow as does the availability of MPEG(1,2,4), AVI audio/video clips through the increasing use of streaming media. Digital video/versatile disk (DVD) usage is growing faster than CD audio usage did when it was first introduced. There is now growing interest in digital cinema, implying that the whole chain from "film" shooting to distribution/projection will be digital.

With all these available digital video "channels," it is amusing to note that the main concern for broadcasters is the relative unavailability of content. Holders of large video, film, and photograph archives, for instance, the British Broadcasting Company (BBC; U.K.), Institut National de L'Audiovisuel (INA; France), and Radio Televisão Portuguesa (RTP; Portugal), find that archive material is in increasing demand. However, the material is typically degraded due to physical problems in repeated projection or playback or simply the chemical decomposition of the original material. Typical problems with much of the archived film material have been increased level of noise, and dirt and sparkle due to the deposition of dust or the abrasion of the material. Of course, there are many more problems specific to the media, e.g., 2-in tape scratches affecting 2-in videotape and vinegar syndrome, which is a Moire pattern affecting film and the film scanning process.

In order to preserve and exploit this material, these defects must be removed so that the picture quality can be restored. Because of the large amount of data, manual retouching is impractical. Therefore, automated techniques have become important. Furthermore, it has been recognized that the reduction of *noise*, in particular before MPEG compression, allows a more efficient usage of the available digital bandwidth [3].

Hence, the area of automated restoration of image sequences has moved from being principally a signal processing research topic to one of more widespread significance. Projects such as the automated restoration of original film and video archives (AURORA) and broadcast restoration of archives by video analysis (BRAVA), which have been funded by the European Union, and recent companies that deal in automated restoration (DUST SA, MediaCleaner, and MTI) are all examples of this increased relevance.

This paper concentrates on two central issues in automated archive restoration: missing data detection and removal as well as noise reduction. It is typical to consider that the two problems are separate and can be treated independently. Thus, it is possible to cite much work on noise reduction for image sequences beginning with the early temporal recursive filtering of Dubois *et al.* [4] and continuing with various optimal schemes [5]–[8] and, more recently, with the 3-D wavelet approaches of Van Roosmalen *et al.* [9]. Work on missing data treatment for image sequences is less common [1], [2], [10]–[12].

Removal of both noise and missing data must rely on motion information in order to be able to remove degradation with any useful level of detail fidelity. However, it is clear that if data is missing in the first instance, then "blind" motion estimation will not be able to render the motion field of the underlying "true" image sequence. This will, of course, have some effect on the subsequent processing, most notably the inability to reconstruct the image data in the missing region. Typically, this results in the replacement of the missing area patch with another missing area patch that is equally disturbing.

Manuscript received January 31, 2001; revised October 5, 2001. This work was supported in part by the AURORA EU Project under Contract AC072 and the BRAVA EU project under Contract IST 1999 11628 from 1995 to 1998 and 2000, respectively. The associate editor coordinating the review of this paper and approving it for publication was Dr. Petar M. Djurić.

A. C. Kokaram is with the Electrical Engineering Department, Trinity College, Dublin, Ireland.

S. J. Godsill is with the Department of Engineering, University of Cambridge, Cambridge, U.K.

Publisher Item Identifier S 1053-587X(02)00568-8.

Similarly, missing data has an effect on the noise reduction step since it reduces the temporal correlation of the frames in the image sequence. In a spatio–temporal scheme, this would typically cause the noise reduction phase to ignore the temporal information and default to some kind of spatial noise reduction, which will reduce the noise-reduction level. Of course, the relative importance of these effects depends on the size of the area corrupted by missing data and the level of noise.

A. Interactions Between the Stages

Historically, various schemes have been developed that coped with the difficulties present with simultaneous estimation using a divide-and-conquer approach.

For instance, previous work has considered the removal of Blotches as a two-stage process: First, detect the missing locations [1], and then, reconstruct the underlying image data [2] using a spatiotemporal image sequence interpolation process. This latter process could be some form of optimal linear interpolation [13] or a variant of a 3-D median filter [12], [14]. Because of the problems with Blotches and motion estimation mentioned earlier, the reconstruction stage may be further specified as a motion reconstruction followed by an image reconstruction stage [15]. This works well, provided that the detection step has been successful.

B. Full Description and MCMC

As far as missing data interpolation is concerned, it is possible to pose the motion reconstruction and image interpolation process as a joint problem [16], but this is an interim step toward a full specification of the problem under one framework.

The first steps toward the full specification were introduced in [17]. A Bayesian framework was employed to present a joint detection and reconstruction methodology that linked the motion reconstruction as an integral part of the process of blotch treatment. In [13], there is an exhaustive discussion of the joint detection/interpolation scheme for blotches.

This paper unifies and extends these ideas by addressing the two issues of missing data and noise reduction in the most complete manner to date. The resulting system is certainly complex, but the solution is made tractable by the use of Bayesian inference in combination with marginalization, composition (or factored) sampling, and Gibbs sampling. By selecting carefully the sequence of estimation of the variables and combining both stochastic and deterministic schemes, the paper discusses a computationally tractable scheme for implementation.

The following sections introduce the concepts, presenting the various priors employed for the unknowns and Section VI-A, and then addresses the particular issue of stochastic solution of the system equations.

II. Quantifying the Problem

Assume that a degraded video signal has been digitized and stored as a sequence of observed image intensities $G_n(\vec{x})$, where n denotes a particular frame in the sequence, and $\vec{x}$ denotes a pixel location within that frame. As discussed above, our method explicitly models two common types of degradation in the image sequence: replacement noise (or "blotches") and random additive noise.

Replacement noise completely obliterates the underlying image pixels at certain pixel positions in the image sequence. The replacement process typically occurs in contiguous patches within a single video frame and will thus be referred to as the "blotch" process (the top and middle rows of Fig. 7 show a good example of blotches in a video sequence). The Blotch process at pixel location $\vec{x}$ in a particular frame is fully specified by random variables $b(\vec{x}) \in \{0, 1\}$ and $c(\vec{x}) \in \Re$. The first of these $[b(\vec{x})]$ is a switching process that determines whether replacement noise is present at $\vec{x}$ [set $b(\vec{x}) = 1$] or absent [set $b(\vec{x}) = 0$]. The second $[c(\vec{x})]$ gives the pixel intensity of the replacement noise at $\vec{x}$.

Random additive noise: Random additive noise $\mu(\vec{x})$, which is modeled here as a Gaussian i.i.d. process with $\mu(\cdot) \sim \mathcal{N}(0, \sigma_\mu^2)$, is also present in a typical frame of video.

The observed degraded pixel values may thus be modeled as

$$G_n(\vec{x}) = (1 - b(\vec{x}))I_n(\vec{x}) + b(\vec{x})c(\vec{x}) + \mu(\vec{x}) \qquad (1)$$

where $I_n(\vec{x})$ is the intensity of the uncorrupted image pixel at position $\vec{x}$ in the frame n. An interesting point to note about this representation of the degradation process is that $c(\vec{x})$ exists as a "hidden" background process even at pixel locations where there is no replacement noise [i.e., $b(\vec{x}) = 0$], and similarly, $I_n(\vec{x})$ exists as missing image data when replacement noise *is* present [i.e., $b(\vec{x}) = 1$]. This feature is an integral part of the MCMC sampling algorithms developed in the paper.

Two distinct (but interdependent) tasks can now be identified in the restoration problem. The missing data detection problem is that of estimating $b(\vec{x})$ at each pixel site. The noise-reduction problem is that of reducing $\mu(\vec{x})$ *without* affecting image details. The replacement model was employed within a nonprobabilistic framework by Kokaram *et al.* in [14] for image sequences and *implicitly* employed in a two-stage Bayesian framework for missing data detection and interpolation by Morris *et al.* [18]–[20].

The replacement noise expression developed here in (1) remains the most suitable form for the corruption caused by missing data simply because the Blotch obliterates the underlying image data (see Fig. 7).

III. Image Sequence Model

The original, uncorrupted image sequence is assumed to be generated from a causal spatio–temporal autoregressive (AR) process [1], [21], [22]. The image sequence can thus be expressed as follows:

$$I_n(\vec{x}) = \sum_{k=1}^{P} a_k I_{n+\vec{q}_k^t}\left(\vec{x} + \vec{q}_k^s + \vec{d}_{n,\, n+\vec{q}_k^t}(\vec{x})\right) + e(\vec{x}) \qquad (2)$$

where the pixel intensity $I_n(\vec{x})$ in frame n is predicted by a linear combination of P pixels within a spatio–temporal neighborhood around $I_n(\vec{x})$, suitably compensated for motion of objects between adjacent frames. The geometry of this neighborhood, or "support," is defined by P offset vectors $\vec{q}_k = [\vec{q}_k^s,\ q_k^t]$,

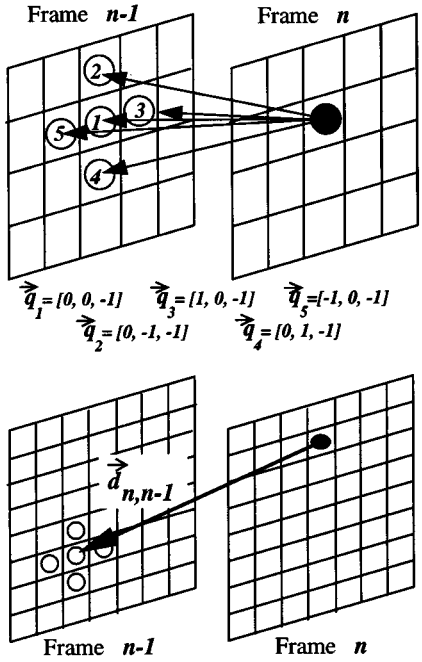

Fig. 1. Top: Spatiotemporal geometry for simple three-dimensional (3-D) AR model (3-D linear predictor) with five taps in the previous frame. Bottom: Geometry for same model incorporating motion. The dark circle shows the pixel to be predicted.

where $\vec{q}_k^s$ is the spatial offset within a particular frame, and q_k^t is the temporal offset in frames. The linear coefficients are the a_k, and $e(\vec{x}) \sim \mathcal{N}(0, \sigma_e^2)$ is an i.i.d. excitation or residual sequence. The motion offset between frames n and $n + \vec{q}_k^t$ is given by $\vec{d}_{n,\, n+\vec{q}_k^t}(\vec{x})$. Subsequently, in the paper, spatial variation in the AR model coefficients will be incorporated in order to account for spatial nonstationarity of the image data. In these cases, the coefficients will be denoted $a_k(\vec{x})$.

Fig. 1 illustrates these ideas using a simple five-tap model ($P = 5$) with $\vec{q}_k$ as indicated by the arrows in the upper half of the figure. This upper image shows the situation in the case of no relative motion offset between frames. The dark pixel in frame n is being predicted by a linear combination of the five sites in the previous frame at locations $\vec{x} + \vec{q}_k$. The intensity of this pixel is $I_n(\vec{x})$ in the model equation (2). The five support pixels are labeled 1 through 5 to correspond with the relevant offset vector $\vec{q}_1$, $\vec{q}_2$, etc. Each pixel in frame n is predicted in the same manner with this model framework. Of course, the model coefficients will not remain the same over the whole image, and in practice, the model coefficients are allowed to vary spatially between small sub-blocks of each image frame.

The lower image in Fig. 1 shows the situation when relative motion of objects between frames is incorporated. In this case, there is a displacement of $\vec{d}_{n,n-1} = [-1, -4]$. Therefore, to predict each pixel in frame n using the same five-tap model as discussed above, each member of the five-tap support must be offset by this motion vector. Hence, we have the term $\vec{x} + \vec{q}_k^s + \vec{d}_{n,\, n+\vec{q}_k^t}(\vec{x})$ in the model expression, which specifies the location of support pixels when motion is present between the frames.

The model expression above (2) is valid for any choice of spatiotemporal neighborhood vectors $\vec{q}_k$. For example, it is perfectly reasonable also to include terms from the current frame (i.e., having $q_k^t = 0$). The only constraint we impose for the work presented here is that the support is *causal*; in other words, all of the intensity values required to form the summation on the right-hand side of (2) are obtained from earlier frames or from the asymmetric half-plane above $\vec{x}$ (other causal structures are also permissible, such as the asymmetric half plane to the left/below/right of $\vec{x}$).

IV. Bayesian Framework

From the degradation model of (1), it can be seen that the principal unknown quantities in frame n are $I_n(\vec{x})$, $b(\vec{x})$, and $c(\vec{x})$. The Bayesian approach presented here infers these unknowns conditional on the corrupted data intensities from the current and surrounding frames $G_{n-1}(\vec{x})$, $G_n(\vec{x})$, and $G_{n+1}(\vec{x})$. In addition, note that the motion information $\vec{d}(\vec{x})$ is also an unknown since it is unavailable directly from the corrupted data.[1] The remaining "hidden" unknowns are the P AR model coefficients $\mathbf{a}(\vec{x}) = [a_1(\vec{x}), a_2(\vec{x}), \ldots a_k(\vec{x})]^T$, and $\sigma_e^2(\vec{x})$. It is assumed that σ_μ^2, which is the additive noise variance, is a user-defined parameter owing to the highly subjective nature of the noise reduction problem. Now, denote the collection of unknowns at pixel $\vec{x}$ in frame n as $\theta(\vec{x}) = [I_n(\vec{x}), b(\vec{x}), c(\vec{x}), \vec{d}(\vec{x}), \mathbf{a}(\vec{x}), \sigma_e^2(\vec{x})]^T$.

Suppose that at any given time, three frames of observed data are available (G_{n-1}, G_n, and G_{n+1}), where G_n denotes all of the observed pixel values in frame n.

In the modified Gibbs sampling solution proposed later, it is required to manipulate and draw samples from the posterior conditional distribution for $\boldsymbol{\theta}$, which is the collection of all $\theta(\vec{x})$ values for frame n of the sequence. Assume for the moment that the uncorrupted image data from surrounding frames I_{n-1} and I_{n+1} is directly available. We assume also that the maximum degree of temporal offset for the AR coefficients is one, i.e., prediction of a given pixel is in terms of image data that is at most one frame in the past.

Proceeding in a Bayesian fashion, the conditional may be written in terms of a product of a likelihood and a prior as follows:

$$p(\boldsymbol{\theta}|I_{n-1},\, G_n,\, I_{n+1}) \propto p(G_n|\boldsymbol{\theta},\, I_{n-1},\, I_{n+1})p(\boldsymbol{\theta}|I_{n-1},\, I_{n+1}). \quad (3)$$

This posterior may be expanded at the single pixel scale, exploiting conditional independence in the model, to yield

$$\begin{aligned} &p(\theta(\vec{x})|G_n(\vec{x}),\, I_{n-1},\, I_{n+1},\, \theta(-\vec{x})) \\ &\quad \propto p(G_n(\vec{x})|\theta(\vec{x}), I_{n-1},\, I_{n+1})p(\theta(\vec{x})|I_{n-1}, I_{n+1}, \theta(-\vec{x})) \\ &\quad = p(G_n(\vec{x})|I_n(\vec{x}),\, c(\vec{x}),\, b(\vec{x})) \\ &\qquad \times p(I_n(\vec{x})|I,\, \mathbf{a}(\vec{x}),\, \sigma_e(\vec{x})^2,\, \vec{d}(\vec{x}),\, I_{n-1},\, I_{n+1}) \\ &\qquad \times p(b(\vec{x})|B)p(c(\vec{x})|C)p\Big(\vec{d}(\vec{x})|D\Big)\, p(\mathbf{a}(\vec{x}))p(\sigma_e(\vec{x})^2) \quad (4) \end{aligned}$$

where $\theta(-\vec{x})$ denotes the collection of θ values in frame n with $\theta(\vec{x})$ omitted and B, C, D, and I denote local dependence

[1]The subscripts have been dropped in the notation for the motion vector $\vec{d}(\vec{x})$ for simplicity and to emphasize that the arguments apply to both backward and forward motion information.

neighborhoods around $\vec{x}$ (in frame n) for variables b c, d, and I_n, respectively. See the sections on prior distributions for details of these neighborhoods.

In the above general expressions, $\vec{d}$, $\mathbf{a}$, and σ_e are explicitly permitted to vary spatially within the frame. This is a desirable feature in general image sequences since they will contain spatial nonstationarity, if only because of the presence of different objects in the scene. In practice, these quantities are expected to be spatially quite smooth within regions corresponding to the same object within an image frame. Hence, in the algorithm implementation, these three parameters are constrained to be piecewise constant within small sub-blocks that form a regular grid over the image.

It is now necessary to assign precise functional forms to the various terms in the prior and likelihood expressions.

A. Corruption Likelihood

The first distribution on the right-hand side of (4) is derived from the model for degradation stated in (1). We have, for a single pixel site

$$p_l(G_n(\vec{x})|I_n(\vec{x}), c(\vec{x}), b(\vec{x})) = \mathcal{N}(G_n(\vec{x}) - (1 - b(\vec{x}))I_n(\vec{x}) - b(\vec{x})c(\vec{x}), \sigma_\mu^2). \quad (5)$$

Here, the notation $p_l(\cdot)$ is introduced to allow the reader to distinguish the form of this particular function when it is used in the subsequent text.

B. Original (Clean) Data Likelihood

The second term on the right of (4) is the likelihood of the original, clean image data in frame n, given clean image data from current and surrounding frames. This may be derived directly from the model statement of equation (2). Since the spatio–temporal AR models are constrained to have causal support both in space and time, the joint distribution of error or residual terms $e(\vec{x})$ can be constructed simply as a product of univariate Gaussians.

For any given pixel at site $\vec{x}$, the error term in (2) is a linear function of the pixel intensity $I_n(\vec{x})$ and those in its local causal support region $\{I_{n+q_k^t}(\vec{x} + \vec{q}_k^s + \vec{d}_{n,\, n+\vec{q}_k^t}(\vec{x})); k = 1, \ldots, P\}$. We can write this as

$$e(\vec{x}) = \boldsymbol{\alpha}(\vec{x})^T \mathbf{i} \quad (6)$$

where $\mathbf{i}$ is a column vector containing all the pixel intensities from frames $n-1$, n, and $n+1$, i.e., $\mathbf{i} = [\text{vec}(I_{n-1})^T, \text{vec}(I_n)^T, \text{vec}(I_{n+1})^T]^T$. $\boldsymbol{\alpha}(\vec{x})$ is a highly sparse column vector containing the elements of $\mathbf{a}(\vec{x})$ arranged to satisfy the AR prediction error equation, i.e.,

$$\boldsymbol{\alpha}(\vec{x})^T \mathbf{i} = I_n(\vec{x}) - \sum_{k=1}^{P} a_k(\vec{x}) I_{n+q_k^t}\left(\vec{x} + \vec{q}_k^s + \vec{d}_{n,\, n+\vec{q}_k^t}(\vec{x})\right).$$

To explain graphically, consider Fig. 2, and assume that any motion has already been compensated for, so that all $\vec{d}$ terms are equal to zero. The figure labels sites in a prediction error sequence (top) and in the image sequence (below) corresponding to three frames of 4×5 pixels. Suppose it is required to calculate

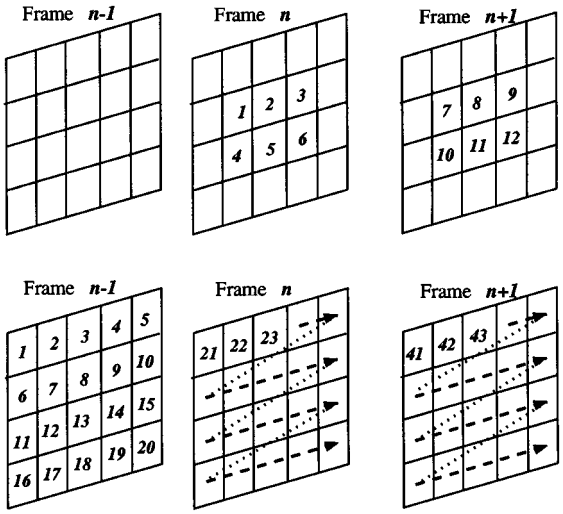

Fig. 2. Three frames of motion-compensated data and their relationship to the data vectors required given the five-tap model discussed in Section III. Top: Locations of sites at which prediction equations are set up. These 12 sites correspond to the excitation vector $\mathbf{e} = [e_1, e_2, e_3, e_4, e_5, e_6, \ldots, e_{12}]$. The sites are chosen so that the data required in support does not extend beyond the bounds of the available frames. Bottom: The sequence for raster scanning pixels into the data vector $\mathbf{i}$.

the prediction error at site 1 in the prediction error frames (which is denoted e_1 in this case). Assume a five-tap 3DAR model as described previously. The bottom part of Fig. 2 labels sites in the image, and for this model, e_1 is the difference between the intensity $I(27)$ and the sum of the pixel intensities at the sites labeled 7, 2, 8, 12, 6, weighted by the 3DAR model weights a_1, a_2, a_3, a_4, a_5, respectively. Creating the vector $\mathbf{i}$ by raster scanning all the data in the three frames into a single column vector allows e_1 to be written as

$$\begin{aligned} e_1 = &[0, -a_2, 0, 0, 0, -a_5, -a_1, -a_3, 0, 0, 0 \\ &-a_4, 0, \ldots, 0, 1, 0, 0, \ldots] \\ &\times [I(1), I(2), I(3), I(4), I(5) \\ &\ldots, I(58), I(59), I(60)]^T \end{aligned} \quad (7)$$

where the single "1" occurs at position 27 in the first vector. Hence, we get (6).

Now, in order to form the likelihood for frames n and $n+1$ conditional on frame $n-1$, stack all the error terms $e(\vec{x})$ corresponding to pixels in frames n and $n+1$ into a single column vector $\mathbf{e}$ and express the whole vector as

$$\mathbf{e} = \mathbf{A}\mathbf{i}$$

where the row of $\mathbf{A}$ corresponding to $e(\vec{x})$ is set equal to $\boldsymbol{\alpha}(\vec{x})^T$.

With reference to Fig. 2, the vector $\mathbf{e}$ is a vector of 12 prediction error elements with positions as shown in the top diagram. Each row of $\mathbf{A}$ then corresponds to one of the 12 prediction errors and is made up in accordance with the intensities required from the lower part of the figure in order to form that particular prediction error equation.

The joint distribution for these residuals in a block of size N pixels is

$$p(\mathbf{e}) = \frac{1}{\sqrt{2\pi\sigma_e^2}^N} \exp\left(-\frac{\mathbf{e}^T\mathbf{e}}{2\sigma_e^2}\right). \quad (8)$$

Note that to generate residuals at each of the N sites in a block, image data outside that block both in space and time would generally be necessary to provide the required support pixels. In Fig. 2, we have a "block" of $N = 12$ pixels, formed as two patches of 2×3 pixels centered in the middle of the 4×5 grid in both frames n and $n+1$.

Note also that in implementation, the motion information $\vec{d}_{n,\,n-1}$, $\vec{d}_{n,\,n+1}$ is used to shift the relevant image blocks prior to sampling of other parameters (such as the AR coefficients). By precompensating for the motion in this way, no motion parameter is explicitly required by the AR coefficient sampling routines.

Now, if we denote the set of pixels values corresponding to the sites used for the prediction errors (in the vector $\mathbf{e}$) as $\mathbf{i}_e$ and the remaining pixels in all three frames as $\mathbf{i}_{-e}$, then the joint conditional likelihood for these pixels is readily obtained as

$$p_i(\mathbf{i}_e|\mathbf{i}_{-e}, \mathbf{a}, \sigma_e^2, \mathbf{d}(\vec{x})) = \frac{1}{\sqrt{2\pi\sigma_e^2}^N} \exp\left(-\frac{\mathbf{i}^T\mathbf{A}^T\mathbf{A}\mathbf{i}}{2\sigma_e^2}\right) \quad (9)$$

where $\mathbf{a}$ and σ_e^2 denote the entire collection of AR coefficients and variances corresponding to the error terms $\mathbf{e}$. The conditional distribution for any subset of $\mathbf{i}$, say $\mathbf{i}_u$, is then, by the conditional probability formula, proportional to this expression. In particular, the single-site conditional distribution required above [see (4)] $p(I_n(\vec{x})|I, \mathbf{a}(\vec{x}), \sigma_e(\vec{x})^2, \vec{d}_{n,\,n-1}(\vec{x}), I_{n-1}, I_{n+1})$ is univariate Gaussian and directly proportional to $p_i(\mathbf{i}_e|\mathbf{i}_{-e}, \mathbf{a}, \sigma_e^2, \mathbf{d}(\vec{x}))$.

In Fig. 2, for example, we could choose to sample the patch of image data at sites $(27, 28, 29, 32, 33, 34)$ in frame n, in which case, simply set $\mathbf{i}_u = [I(27), I(28), I(29), I(32), I(33), I(34)]$. $p_i(\cdot)$ will be used later on to identify the particular probability function derived in this section.

V. Priors

The remaining distributions encode the prior belief about the values of the various unknowns. For simplicity, a uniform prior is assigned to $\mathbf{a}$. This removes $p(\mathbf{a}(\vec{x}))$ from (4). The variance σ_e^2 is assigned a noninformative prior $p(\sigma_e^2) \propto 1/\sigma_e^2$, following [23]. The remaining priors are more involved and deserve separate discussion.

A. Motion Prior

In practice for this application, it is sufficient to encode the notion of local motion smoothness in order to achieve implicit motion interpolation. The prior adopted for motion smoothness is a Gibbs energy prior, for instance, as introduced by Konrad and Dubois [24] and Stiller [25]. To reduce the complexity of the final solution, the motion field is block based, with one motion vector being employed for each specified block in the image.

The prior for $\vec{d}_{n,\,n-1}(\vec{x})$, which is the motion vector mapping the pixel at $\vec{x}$ in frame n into frame $n-1$, is as follows:

$$p_d\left(\vec{d}_{n,n-1}(\vec{x})|\vec{d}_{n,n-1}(-\vec{x}), \mathbf{S}_n(\vec{x})\right) \propto \exp - \left(\sum_{\vec{s}\in\mathbf{S}_n(\vec{x})} \lambda(\vec{s}) \left[\vec{d}_{n,n-1}(\vec{x}) - \vec{d}(\vec{s})\right]^2\right) \quad (10)$$

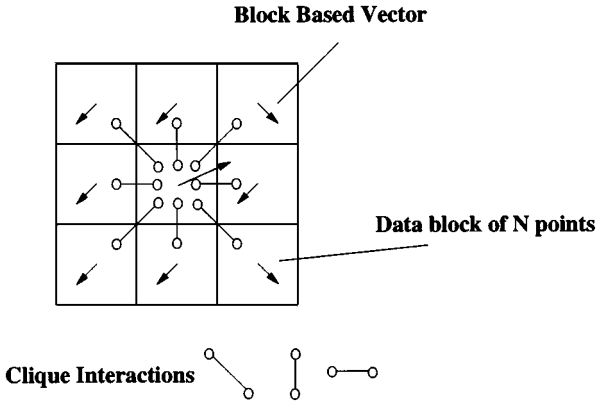

Fig. 3. Neighborhood and cliques for $p_d(\vec{d}_{n,\,n-1}(\vec{x})|\cdot)$.

where $\vec{s}$ is each motion vector in the neighborhood represented by $\mathbf{S}_n(\vec{x})$, and $\lambda(\vec{s})$ is the weight associated with each clique. The situation is illustrated in Fig. 3. The same prior is used for $\vec{d}_{n,\,n+1}(\vec{x})$. The neighborhood $\mathbf{S}_n(\vec{x})$ is the eight nearest neighbor blocks, as shown in the figure.

The driving force behind the specification of this prior is to penalize the creation of motion vector fields that have a high local gradient. Thus, in Fig. 3, the vector most likely to be appropriate is that which is aligned with the same directions as most of the other vectors. In the case shown in Fig. 3, that direction is to the left and down. Since the actual implementation of the final algorithm uses a candidate selection step for motion vectors, the effect of this prior is similar to that of a vector median operation, i.e., many of the vectors are caused to align in the same direction and only large discontinuities are allowed. This behavior follows that of typical observed motion fields since motion tends to be smooth (or the same) within a single object but then drastically different between objects.

In order to discourage "smoothness" over too large a range, $\lambda(\vec{s})$ is defined as $\lambda(\vec{s}) = \Lambda/|\vec{X}(\vec{s}) - \vec{x}_B|$, where $\vec{X}(\vec{s})$ is the location of the block (in "block" units) providing the neighborhood vector $\vec{s}$, and $\vec{x}_B$ is the central block location. $\Lambda = 2.0$ in the experiments presented later.

B. Priors for Corruption and Detection

Since blotches tend to be "convex" clumps of degradation, the prior for $b(\vec{x})$ should encourage contiguous areas of $b = 1$ to form. In practice, blotches tend to have fairly constant intensity (see Fig. 7). If a texture exists, it is certainly smoother than that in the original image. Thus, the prior for $c(\vec{x})$ should encourage smoothness of intensity, in much the same way that the prior for $b(\cdot)$ encourages smoothness in its binary configuration. Therefore, it is reasonable to place a similar energy prior on both the binary field $b(\vec{x})$ and the blotch value field $c(\vec{x})$.

It is found that acknowledging discontinuities in these fields leads to much better behavior. That is to say that blotches are smooth only *within* their boundaries but show a large contrast with the background. Edges in the c (corruption) and b (blotch) fields must correspond to edges in the image since the corrupted areas are generally well delineated from their surrounding by a marked grey-scale transition. Thus, a simple zero crossing edge detector employed on the degraded image will enable the rough

configuration of an edge field that can be used subsequently to define the priors on the c and b fields.[2]

The priors are therefore defined as follows.

$$p_c(c(\vec{x})|C) \propto \exp\left(-\sum_{k=1}^{8} \lambda_k^c(1-u(\vec{x},\vec{x}+\vec{v}_k))(c(\vec{x})-c(\vec{x}+\vec{v}_k)^2\right) \tag{11}$$

$$p_b(b(\vec{x})|B) \propto \exp\left(-\sum_{k=1}^{8} \lambda_k^b(1-u(\vec{x},\vec{x}+\vec{v}_k))|b(\vec{x})-b(\vec{x}+\vec{v}_k)|\right) \tag{12}$$

where the eight vectors $\vec{v}_k$ define the eight connected neighborhood of $\vec{x}$, and C, B represent sets of these values from the respective fields (as previously). $u(\vec{x}, \vec{x}+\vec{v}_k)$ is set to 1 if there is a significant zero crossing between the location $\vec{x}$ in the image and $\vec{x}+\vec{v}_k$ from which a neighborhood pixel is extracted. Thus, the smoothness constraint [i.e., the pairwise interaction denoted by $(c(\vec{x})-c(\vec{x}+\vec{v}_k))$ for instance] is turned off across significant edges in the image. Note that these priors are defined on the *pixel* resolution image grid, whereas the motion prior discussed previously is defined on a *block* grid. In the results shown later $u(\cdot)$, (the edge field) was configured using an edge detector employing difference of Gaussians (DOGs) [26] with the gradient threshold set at 5.0, the variance of the Gaussian filters was 1.0, 1.6, and the filter window sizes were 9×9.

Note that the Gaussian Markov random field (GMRF) prior used for $c(\cdot)$ is in fact *almost* identical to an autoregressive model for $c(\vec{x})$, except that it defines the conditional distribution only. The use of this prior allows the samples for c to be generated from the well-known Gaussian distribution and therefore results in a low computation step.

For both these priors, λ_k^c, λ_k^b are assigned values such that $\lambda_k^c = \Lambda^c/|\vec{v}_k|$, $\lambda_k^b = \Lambda^b/|\vec{v}_k|$. This makes the hyperparameter weighting circularly symmetric.

VI. Solving for the Unknowns

The solution is generated by manipulating $p(\boldsymbol{\theta}|I_{n-1}, G_n, I_{n+1})$. For instance, the MAP estimate is generated by maximizing the distribution with respect to the unknowns. Unfortunately, due to the nonlinear nature of the expression, a closed-form solution to the optimization problem is not available. Instead, the Gibbs sampler is used to generate random samples from the required distribution. These random samples can be manipulated numerically to yield the required estimate. For instance, the average of the samples yields the minimum mean-square error (MMSE) estimate.

A. Gibbs Sampler

The Gibbs sampler is an MCMC technique that decomposes the problem of generating a sample from a high-dimensional probability density function (pdf) into a series of draws from conditional pdfs of lower dimension. In this case, it is required ultimately to draw samples for the entire image field I_n and, therefore, all the associated variables for motion, etc. This draw can be decomposed into a series of draws from the conditional distribution for each variable on a block basis, thus reducing the dimensionality of the pdf to be manipulated.

Consider, therefore, that processing is performed on a block basis, and let $\mathbf{i}$ contain at least the pixels to be treated in a block and their immediate AR support. The corrupted pixels (noisy or missing) are denoted $\mathbf{i}_u$ (i.e., all the pixels to be treated in the current block) and the remaining pixels as $\mathbf{i}_k$. The vector $\mathbf{i}$ introduced previously, therefore, consists of data to be estimated (or unknown data) $\mathbf{i}_u$, and the remaining data $\mathbf{i}_k$. In the example given in Fig. 2, the data to be estimated is the central block of 2 $\times$ 3 pixels in frame n, and all other data constitutes $\mathbf{i}_k$.

The Gibbs sampler then operates iteratively with replacement, given some starting guess for the unknowns, by drawing random samples from the conditional posterior distribution at each block for each unknown in turn

$$\begin{aligned}
\mathbf{a} &\sim p(\mathbf{a}|\mathbf{i}, \sigma_e^2, \mathbf{b}, \mathbf{d}, \mathbf{c}, \mathbf{g}, \sigma_\mu^2)\\
\mathbf{b} &\sim p(\mathbf{b}|\mathbf{i}, \sigma_e^2, \mathbf{a}, \mathbf{d}, \mathbf{c}, \mathbf{g}, \sigma_\mu^2)\\
\mathbf{c} &\sim p(\mathbf{c}|\mathbf{i}, \sigma_e^2, \mathbf{a}, \mathbf{d}, \mathbf{b}, \mathbf{g}, \sigma_\mu^2)\\
\sigma_e^2 &\sim p(\sigma_e^2|\mathbf{a}, \mathbf{i}, \mathbf{b}, \mathbf{d}, \mathbf{c}, \mathbf{g}, \sigma_\mu^2)\\
\mathbf{d} &\sim p(\mathbf{d}|\mathbf{a}, \mathbf{i}, \mathbf{b}, \sigma_e^2, \mathbf{c}, \mathbf{g}, \sigma_\mu^2)\\
\mathbf{i_u} &\sim p(\mathbf{i}_u|\mathbf{a}, \mathbf{i}_k, \sigma_e^2, \mathbf{b}, \mathbf{d}, \mathbf{c}, \mathbf{g}, \sigma_\mu^2)
\end{aligned}$$

where $\mathbf{c}$, $\mathbf{b}$, etc., are vectors containing the relevant parameters, and the location argument $\vec{x}$ has been dropped for simplicity. These conditionals can be derived by manipulation of the joint posterior given in (4). This sampling procedure is repeated until convergence is reached, according to some suitable criterion. The process allows for MMSE, maximum *a posteriori* (MAP), or sampled estimates of $\mathbf{i}$ by manipulation of the sampled values following convergence.

Given the "noninformative" [27] prior distributions for $\mathbf{a}$, σ_e^2 introduced previously, the sampling operations for $\mathbf{a}$, σ_e^2, $\mathbf{i}_u$ involve simple random draws from well-known distributions. However, some practical considerations encourage an altered sampling strategy. In this strategy, the motion and image model variables are sampled jointly on a block basis, and the image data itself is sampled jointly with c and b on a pixel basis.

B. Adaptations to the Gibbs Sampler

The convergence of the Gibbs sampler is generally improved if several unknowns are sampled jointly [28]. This is possible using the method of composition [16], [23], . A random draw from $p(\mathbf{a}, \sigma_e^2, \mathbf{d}|\boldsymbol{\theta}_{-(\mathbf{a},\mathbf{d},\sigma_e)})$, for instance, is made possible by the decomposition

$$\begin{aligned}
&p(\mathbf{a}, \sigma_e^2, \mathbf{d}|\mathbf{b}, \mathbf{c}, \mathbf{i})\\
&\quad = p(\mathbf{a}|\mathbf{d}, \mathbf{b}, \mathbf{c}, \sigma_e^2, \mathbf{i})p(\sigma_e^2|\mathbf{d}, \mathbf{b}, \mathbf{c}, \mathbf{i})p(\mathbf{d}|\mathbf{b}, \mathbf{c}, \mathbf{i}).
\end{aligned} \tag{13}$$

Note that $\mathbf{a}$, σ_e^2, $\mathbf{d}$ are conditionally independent of $\mathbf{g}$, and $\boldsymbol{\theta}_{-(\mathbf{a},\mathbf{d},\sigma_e)}$ is a vector containing all the parameters in $\boldsymbol{\theta}$ except for $\mathbf{a}$, $\mathbf{d}$, σ_e. The various composition terms can be derived by successively *integrating out* $\mathbf{a}$ and then σ_e^2 from the posterior distribution.

[2]It is accepted that the incorporation of this type of information derived from the observed data affects the importance of the prior, but for all practical purposes, the result is effective.

1) Joint Estimates for Motion and AR Parameters: Random draws from $p(\mathbf{a}, \sigma_e^2, \mathbf{d}|\mathbf{b}, \mathbf{c}, \mathbf{i})$ can therefore be implemented by drawing from $p(\mathbf{d}|\mathbf{b}, \mathbf{c}, \mathbf{i})$ followed by a draw from $p(\sigma_e^2|\mathbf{d}, \mathbf{b}, \mathbf{c}, \mathbf{i})$ using the value of $\mathbf{d}$ drawn previously, and then similarly for $p(\mathbf{a}|\cdots)$, using the samples just generated for σ_e^2 and $\mathbf{d}$. In this manner, a joint draw for $\mathbf{b}, \mathbf{c}, \mathbf{i}$ is also achieved.

The expressions required to perform these joint draws can be derived [13], [16] by employing the vector $\mathbf{e}$ as previously described, containing all the excitation terms from (2) within a block of data and expressing it as $\mathbf{e} = \mathbf{i}_e - \mathbf{I}\mathbf{a}$. Here, $\mathbf{i}_e$ has the same meaning as before. $\mathbf{I}$ is created in a similar way to that of $\mathbf{A}$, which was created previously to result in the excitation vector $\mathbf{e} = \mathbf{A}\mathbf{i}$.

Thus, $\mathbf{I}$ is a matrix of pixels chosen from the frames such that the product of each row of $\mathbf{I}$ and the coefficient vector $\mathbf{a}$ results in each sample of the excitation vector $\mathbf{e}$ through (2). This product of the rows of $\mathbf{I}$ and $\mathbf{a}$ (or the rows of $\mathbf{A}$ and $\mathbf{i}$) is the convolution of the data with the linear prediction filter arising out of (2). Hence, this filter is defined by the coefficients $\mathbf{a}$ and the chosen filter geometry indicated by the offset vectors $\vec{q}_k$.

This expression then leads to the following distributions required in the joint draw from $p(\mathbf{a}, \sigma_e^2, \mathbf{d}|\mathbf{i})$:

$$\begin{aligned} p(\mathbf{a}|\mathbf{d}, \mathbf{i}, \sigma_e^2) &= \mathcal{N}_P(\hat{\mathbf{a}}, \sigma_e^2(\mathbf{I}^T\mathbf{I})^{-1}) \\ p(\sigma_e^2|\mathbf{d}, \mathbf{i}) &= \mathrm{IG}((N-P)/2, E(\hat{\mathbf{a}}, \mathbf{i}, \mathbf{d})/2) \\ p(\mathbf{d}|\mathbf{i}, D) &\propto \frac{E(\hat{\mathbf{a}}, \mathbf{i}, \mathbf{d})^{-(N-P)/2}}{|\mathbf{I}^T\mathbf{I}|^{1/2}}\, p(\mathbf{d}|D) \qquad (14) \\ \text{where } \hat{\mathbf{a}} &= (\mathbf{I}^T\mathbf{I})^{-1}\mathbf{I}^T\mathbf{i} \\ \text{and } E(\hat{\mathbf{a}}, \mathbf{i}, \mathbf{d}) &= \mathbf{e}^T\mathbf{e} \qquad (15) \end{aligned}$$

where N is the number of pixels in the image block, and D [which is shorthand for $\mathbf{S}_n(\vec{x})$] represents a neighborhood of vectors surrounding the sampled location. The derivation of these expressions can be found in [13].

2) Joint Estimates for $\mathbf{b}, \mathbf{c}, \mathbf{i}$*:* In practice, the draws for $\mathbf{b}, \mathbf{c}, \mathbf{i}$ are performed jointly on a pixel by pixel basis, sampling from the expression

$$p\left(b(\vec{x}), c(\vec{x}), I_n(\vec{x})|\mathbf{d}, \mathbf{a}, \sigma_e^2, B, C, \mathbf{i}_{-\mathbf{x}}, \mathbf{g}, \sigma_\mu^2\right) \qquad (16)$$

where $\mathbf{i}_{-\mathbf{x}}$ denotes all clean image data *not* at site $\mathbf{x}$ but required for all prediction equations [see (2)] involving site $\vec{x}$. This joint draw is unusual because of the switching process in the likelihood, but since $b(\cdot)$ is a binary field, a feasible sampling scheme results. The distributions required for the composition sampling can be derived (see Appendix A) by integrating the posterior to yield

$$p(b(\vec{x})|B, \ldots) = \begin{cases} \mathcal{F}_1 p_b(b=1|B), & \text{for } b(\vec{x}) = 1 \\ \mathcal{F}_2 p_b(b=0|B), & \text{for } b(\vec{x}) = 0 \end{cases} \qquad (17)$$

$$p(i_n(\vec{x})|b(\vec{x}), \ldots) = \begin{cases} \mathcal{N}\left(\hat{i}, \dfrac{\sigma_e^2}{\mathbf{a}_u^T\mathbf{a}_u}\right), & \text{for } b(\vec{x}) = 1 \\ \mathcal{N}\left(\bar{i}, \dfrac{\sigma_\mu^2\sigma_i^2}{\sigma_\mu^2 + \sigma_i^2}\right), & \text{for } b(\vec{x}) = 0 \end{cases} \qquad (18)$$

$$p(c(\vec{x})|b(\vec{x}), i_n(\vec{x}), \ldots) = \begin{cases} \mathcal{N}\left(\bar{c}, \dfrac{\sigma_\mu^2\sigma_c^2}{\sigma_\mu^2 + \sigma_c^2}\right), & \text{for } b(\vec{x}) = 1 \\ p_c(c(\vec{x})|C), & \text{for } b(\vec{x}) = 0 \end{cases} \qquad (19)$$

where B is the set of neighborhood of detection indicators $b(\cdot)$ surrounding the sampled location, i_n, g_n are the intensities of pixels at location $\vec{x}$ in the clean and dirty image, respectively, and

$$\begin{aligned} \sigma_c^2 &= 1 \Big/ \left(2\sum_k \Lambda_k^c\right) \\ \hat{i} &= \frac{\mathbf{a}_u^T\mathbf{A}_k\mathbf{i}_k}{\mathbf{a}_u^T\mathbf{a}_u} \\ \hat{c} &= \frac{\sum_k \Lambda_k^c c(\vec{x} + \vec{v}_k)}{\sum_k \lambda_k^c} \\ \sigma_i^2 &= \sigma_e^2/(\mathbf{a}_u^T\mathbf{a}_u) \\ \Lambda_k^c &= \lambda_k^c(1 - u(\vec{x}, \vec{x} + \vec{v}_k)) \\ \bar{c} &= \frac{\hat{c}\sigma_\mu^2 + g_n\sigma_c^2}{\sigma_\mu^2 + \sigma_c^2} \\ \bar{i} &= \frac{g_n\sigma_i^2 - \hat{i}\sigma_\mu^2}{\sigma_\mu^2 + \sigma_i^2} \\ \mathcal{F}_1 &= \frac{1}{\sqrt{2\pi\sigma_\mu^2}}\left[\exp - \left(\frac{(g_n - \bar{c})^2}{2\sigma_\mu^2}\right)\right] p_c(c = \bar{c}|C) \\ \mathcal{F}_2 &= \frac{1}{\sqrt{2\pi\sigma_\mu^2}}\left[\exp - \left(\frac{(g_n - \bar{i})^2}{2\sigma_\mu^2}\right)\right] p_i(i = \bar{i}|\cdot). \qquad (20) \end{aligned}$$

Note that it is necessary to decompose the coefficient matrix $\mathbf{A}$ into a matrix $\mathbf{A}_k$ and a column vector $\mathbf{a}_u$ so that the prediction error $(\mathbf{e})$ at the pixel sites whose model support overlaps with $\vec{x}$ may be expressed as $\mathbf{e} = \mathbf{A}_k\mathbf{i}_k + \mathbf{a}_u i(\vec{x})$. Here, $i(\vec{x})$ is the unknown image data at site $\vec{x}$. $\mathbf{i}_k$ are the current sampled values of the original image data that are located at sites in the region of $\vec{x}$, which include $\vec{x}$ in their model prediction support.

VII. Hybrid Options for Motion

The conditional distribution for motion in (14) is very difficult to sample from directly. It may be possible to produce a Gaussian approximation to the distribution that is simpler to sample from by linearizing the function [23]. However, motion is typically a spatially low-frequency signal (piecewise constant), and therefore, it is almost certain that the correct motion for a particular block exists elsewhere nearby. It is here, in particular, that one can make use of motion vectors generated from a prerun of a standard, deterministic motion-estimation process.

Using these initial estimates, it then becomes possible to draw samples from the numerically evaluated pdf for $\mathbf{d} \sim p(\mathbf{d}|\mathbf{i}, \sigma_e^2, \mathbf{a}, \mathbf{b}, \mathbf{c})$ in a local region around the current estimate. In essence, the actual procedure employed for sampling for $\mathbf{d}$ involves proposing eight candidate vectors from the neighborhood and an additional set of nine created by perturbing the current sampled vector at the site by ± 1

pixel. The motion sample is then drawn from this set by direct numerical evaluation of the probability distribution, assuming that the probability of all other samples is zero. This is an application of the *Griddy sampler* [29]. It is straightforward to incorporate this step into a Metropolis–Hastings scheme for rigorous sampling of the motion parameter.

Initial motion estimates can be taken from any number of motion estimators currently available. The multiresolution gradient-based technique discussed in [15] is employed here. This mixture of deterministic and stochastic techniques is extremely useful from both the computation and convergence property points of view.

VIII. JOMBANDI

This section presents a clear recipe for the overall procedure by outlining the chronology of the iterations. The process will be called the joint model-based noise reduction, detection, and interpolation (JOMBANDI) algorithm.

There are two major parts to the algorithm: The first part is the joint draw for $\mathbf{a}$, $\mathbf{d}$, σ_e^2, and the second is the draw for $\mathbf{b}$, $\mathbf{c}$, $\mathbf{i}_u$. The first joint draw is performed on a *block* basis since the motion and coefficient fields are expected to be generally smooth functions. The second draw is performed on a pixel basis in order to more carefully delineate the missing regions.

Each draw is performed for all the variables across the whole image before moving on to the draw for the next set of variables. It is also possible to make the draws in turn at each site before moving onto the next, but sweeping through the image in the fashion described here leads to a simpler implementation particularly with respect to computing issues like memory access, data caching, and pipelining. The description begins with a listing of the overall sequence of activities. The joint draw for $\mathbf{a}$, $\mathbf{d}$, σ_e^2 is identical to the pure blotch detection case, and exhaustive discussion can be found in [13] and [17].

First of all, it must be noted that the problem is to find $I_{n-1}(\cdot)$, $I_n(\cdot)$, $I_{n+1}(\cdot)$ given the observed, corrupted frames $G_{n-1}(\cdot)$, $G_n(\cdot)$, $G_{n+1}(\cdot)$. Although it may be possible to design a joint scheme for simultaneously restoring the frames, it is convenient to employ a higher level invocation of the Gibbs sampler so that each image can be treated in turn. Thus, the iterations may proceed as follows:

$$\begin{aligned}
I_n^0(\vec{x}) &\sim p(I_n(\vec{x})|G_n, G_{n-1}, G_{n+1}, I_{n-1}^0, I_{n+1}^0)\\
I_{n+1}^1(\vec{x}) &\sim p(I_{n+1}(\vec{x})|G_n, G_{n-1}, G_{n+1}, I_{n-1}^0, I_n^0)\\
I_{n-1}^1(\vec{x}) &\sim p(I_{n-1}(\vec{x})|G_n, G_{n-1}, G_{n+1}, I_n^0, I_{n+1}^1)\\
I_n^1(\vec{x}) &\sim p(I_n(\vec{x})|G_n, G_{n-1}, G_{n+1}, I_{n-1}^1, I_{n+1}^1)\\
&\vdots \qquad\qquad \vdots
\end{aligned}$$

Each iteration represents a complete sweep over each frame. Note that for any finite frame window in time, the conditionals employed in estimating the first and last frame are approximate since the relevant temporal support is incomplete.

A. Process Overview

1) The first step is to generate a *kick start* of estimates for the sampler. This is achieved using a simple technique for detection and interpolation of the missing data. Any one of the standard techniques can be employed [13], the combination of a gradient-based motion estimator [15] for estimating motion and the spike detection index with polarity (SDIp)[3] for detection is used here. The main point of this step is actually to generate reasonable estimates of motion in the areas that do not contain missing data.

   The SDIp initializes the $b(\vec{x})$ field as follows:

$$b(\vec{x}) = \begin{cases} 1, & \text{if } (|\Delta_f| > T) \text{ AND } (|\Delta_b| > T) \\ & \text{AND } (\text{sign}(\Delta_f) == \text{sign}(\Delta_b)) \\ 0, & \text{otherwise} \end{cases}$$

   where $\Delta_b = I_n(\vec{x}) - I_{n-1}\left(\vec{x} + \vec{d}_{n,n-1}(\vec{x})\right)$

   and $\Delta_f = I_n(\vec{x}) - I_{n+1}\left(\vec{x} + \vec{d}_{n,n+1}(\vec{x})\right)$.

   The SDIp simply flags a pixel as corrupt when *both* the forward and backward motion compensated frame differences are higher than some threshold T (user selected).

   Since the samples for the unknowns are generated using joint sampling strategies, it is found that using the degraded data as the start image for both the $c(\vec{x})$ and $I_n(\vec{x})$ fields is adequate.
2) The image is divided into blocks and in each block one sample each for $\vec{d}_{n,n-1}$, $\vec{d}_{n,n+1}$, $\mathbf{a}$, σ_e^2 is drawn jointly, in that order. All blocks are visited in a "checkerboard" fashion (see Besag [30]). This requires that the block-motion neighborhood employed for the next block processed must not overlap with the neighborhood of the current processed block. Using the eight nearest-neighborhood configuration presented above, the blocks are visited so that the next site is always at least two blocks away in one direction from the current site.
3) The image is now scanned *pixel* by *pixel*, again in a checkerboard fashion. At each site, a sample for $c(\vec{x})$, $I_n(\vec{x})$, $b(\vec{x})$ is drawn jointly. The samples are drawn based on the values of model coefficients, variance, and motion that have been estimated in step 2) for the block in which the current site lies. Some overlap between the blocks would therefore lead to a more smooth variation in the sampled model coefficients, motion, etc. The importance of the size of this overlap is not investigated here.
4) The last two steps are repeated until some convergence criterion is satisfied or until enough samples of $b(\vec{x})$, $I_n(\vec{x})$ have been collected so that it is deemed sufficient to determine some numerical estimate. Note that these samples must be collected after the chain has converged.

B. Joint Sampling for $c(\vec{x})$, $I_n(\vec{x})$, $b(\vec{x})$

The joint sample for c, I, b is achieved at each *pixel* site by the steps described as follows. Note that JOMBANDI requires just three hyperparameters—Λ^b, Λ^c, σ_μ^2—that are user defined. They control the connectivity of the fields $b(\cdot)$ and $c(\cdot)$ and the noise reduction level, respectively. High values for Λ bias JOMBANDI toward detection of flat regions only.

[3]related to early work by Storey [10].

1) The first step is to draw a sample for $b(\vec{x})$. This is done by evaluating (17) for both cases $b(\vec{x}) = 0, 1$. The sample is then drawn numerically. Drawing the sample requires knowledge of the normalizing constants Z_i, Z_c for the distributions $p_i(\cdot)$, $p_c(\cdot)$. Because $p_i(\cdot)$ is a univariate Gaussian distribution, determination of Z_i is straightforward [see (18)]. Depending on the choice for $p_c(\,)$, evaluation of Z_c can be difficult. If $p_c(\cdot)$ is chosen to be a GMRF as in this case, it is a univariate Gaussian, and the normalizing constant is simple to determine.
2) If $b(\vec{x}) = 1$, then a missing pixel has been detected, and $c(\vec{x})$ is drawn from a Gaussian distribution whose mean is $\overline{c}$ and variance as given in (19). $i(\vec{x})$ has to be interpolated, and a sample is drawn using (18) for $b = 1$. The mechanics of drawing both samples are straightforward. In the case of i, first generate the least squares estimate of the interpolant $\hat{i}$ and add to this Gaussian white noise of variance σ_i^2. This draw is the univariate equivalent of the multivariate Gaussian draw used for $\mathbf{a}$ in the separate joint draw for $\mathbf{a}$, $\mathbf{d}$, σ_e^2.
3) If $b(\vec{x}) = 0$, then the pixel is uncorrupted (by Blotches), and only noise reduction needs to be performed. The corrected intensity is therefore generated by adding to the least squares estimate $\overline{i}$ Gaussian noise of variance $(\sigma_\mu^2\sigma_i^2)/(\sigma_\mu^2 + \sigma_i^2)$. Because the pixel is uncorrupted by blotching, the value for $c(\vec{x})$ must be drawn from its prior. This draw is again from a univariate Gaussian as the prior for c is a GMRF.

IX. Relationships

It is difficult to compare this framework with previous systems since this is the first that treats missing data and noise jointly. However, it is educational to examine how systems that treat the noise and missing data problems separately are in fact special cases of *the deterministic aspects* of JOMBANDI.

A. Relationships With Noise Reducers

Viewing JOMBANDI purely from a noise-reduction standpoint, it can be seen that the temporal Wiener filter proposed for image sequences (see Katsagellos *et al.* [6]) is a special case of the estimate $\overline{i}$. This proposal estimates the intensity at a single pixel site $\vec{x}$, $\overline{i}_w$ as

$$\overline{i}_w = \frac{\sigma_g^2 - \sigma_\mu^2}{\sigma_g^2}(g_n - \overline{g}_n) + \overline{g}_n \tag{21}$$

where $\overline{g}_n$ is the mean of the observed (corrupted) image sequence data in a local spatio–temporal region around the site to be estimated, and g_n is the observed (corrupted) pixel value at that site. Noise reduction is thus achieved through the weighted average between this mean and the observed data. $\overline{g}_n$ was employed as an estimate of the underlying true signal.

Noting that $\sigma_g^2 = \sigma_i^2 + \sigma_\mu^2$ from (1) (in the absence of missing data), the above can be re-expressed as

$$\begin{aligned}\overline{i}_w &= \frac{(g_n - \overline{g}_n)\sigma_i^2 + \overline{g}_n(\sigma_i^2 + \sigma_\mu^2)}{\sigma_i^2 + \sigma_\mu^2} \\ &= \frac{g_n\sigma_i^2 + \overline{g}_n\sigma\mu^2}{\sigma_i^2 + \sigma_\mu^2}.\end{aligned} \tag{22}$$

This estimate is the same as $\overline{i}$, where $\overline{g}_n = -\hat{i}$. Thus, $\overline{i}$ in JOMBANDI is generalizing the idea proposed by Katsagellos *et al.* by allowing for an adaptive, nonstationary estimate for the underlying image data $\hat{i}$ that depends on a weighted average of local spatio–temporal pixel values. That weighted average in turn depends on local image details through the use of the 3-D AR model.

Furthermore, it is possible to separate the joint draws for the variables in JOMBANDI and to employ a draw for $\mathbf{i}$ separately from b, c. $\mathbf{i}$ can then be solved on a block basis in a separate Gibbs sampling step. In that case, it can be shown that $\overline{\mathbf{i}}$ (similar to $\overline{i}$) is related to the spatio–temporal FIR Wiener solution for noise reduction presented by Kokaram [13], [31]. By assuming a circularly symmetric correlation structure in each such block, and using the whole block as support for the AR model, the FFT can be used to solve for $\overline{\mathbf{i}}$ and, hence, a relationship with the 3-D IIR Wiener filter.

Purely temporally recursive or adaptive noise reduction systems like [4], frame averaging, and adaptive weighted averaging (AWA; see [32]) can be seen as special cases of the $\overline{i}$ estimate in JOMBANDI, when single tap, purely temporal AR models are used. In such a case, $\hat{i}$ simply becomes a copy of the motion-compensated pixel in the previous frame or an average of the same pixels in the previous and next frames. Then, $\overline{i}$ is a weighted combination of those pixels and the observed image data. This is identical to recursive noise reduction, provided the estimated image in each frame of JOMBANDI is then immediately reused in processing the next frame.

B. Relationships to Blotch Removers

In the task of blotch removal, JOMBANDI can be related to previous systems on two aspects: detection given by $b(\vec{x})$ and interpolation given by $\hat{i}$.

Assuming $b(\vec{x})$ and all other motion and model parameters are given, then all previous motion-compensated interpolation schemes based on linear models are special cases of the interpolation step in JOMBANDI [13]. This follows straightforwardly since all linear predictive models can be expressed in the form used in (2), and $p_i(i|\cdot)$ is independent of b, c. Therefore, $\hat{i}$ will result in the same form, regardless of the trappings of the rest of the algorithm. Cut-and-paste operations (the simplest of interpolators) can be derived by using a purely temporal AR model with one tap that is set to -1.

There are, however, more alternatives for blotch *detection*. Most of the available blotch detection processes are pixel based, but all are based on temporal motion-compensated frame differences, e.g., spike detection index a (SDIa), SDIp, and rank order detector (ROD) [11], [13], [33]. The basic tenet is to flag large motion-compensated frame differences when these occur in both the backward and forward frames. If the $b(\vec{x})$ field was split into two components [a field between frames n and $n-1$, b_{-1} with a similar field in the forward direction b_{+1} and a purely temporal AR model were used having one coefficient ($=-1$)], then these simple blotch detectors would be the same as the "least-squares" estimate for $b(\vec{x})$ used in JOMBANDI. These fields would denote discontinuities in time, and there would then be a four-state variable at each site, i.e., 00, 01, 10, and 11. There would be the need to introduce penalties in the priors for the

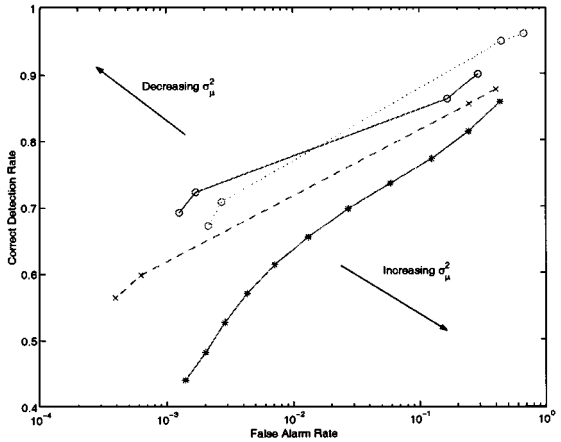

Fig. 4. Performance of JOMBANDI and SDIp at $\sigma_\mu^2 = 100$: JOMBANDI with five-tap 3-D AR (o—) with one-tap 3-D AR (×- -), SDIp (*—); JOMBANDI with five-tap 3-D AR and $\sigma_\mu^2 = 200$ (o . . .).

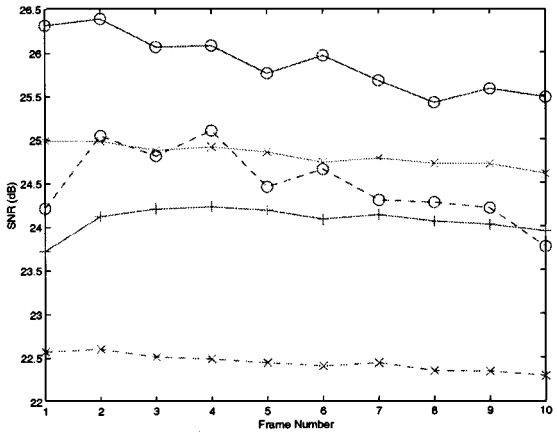

Fig. 5. Noise-reduction performance for $\sigma_\mu^2 = 100$. JOMBANDI, temporal Wiener (×—), and recursive filtering (+—). JOMBANDI settings are $\Lambda^c = 0.15$, $\Lambda^b = 1.0$ (o- -); $\Lambda^c = 0.15$, $\Lambda^b = 4.0$ (o—).

11 state. This latter framework is similar to that employed by Morris [1], [20].

X. Results

To assess the performance of JOMBANDI, a 256 × 256 subsection of the mobile and calendar sequence was corrupted with blotches that follow the prior for $c(\vec{x})$, and Gaussian noise of $\sigma_\mu^2 = 100, 200$ was added in keeping with the degradation model discussed here. A number of experiments were performed to evaluate the behavior as a blotch detector and as a noise reducer. Motion behavior is examined together with real degradation.

A. Blotch Detection Performance

Fig. 4 shows a receiver operating characteristic that compares the performance of JOMBANDI with the SDIp with respect to their blotch detection performance. To create the characteristics, the processes were run with a range of parameter settings. In the case of SDIp, $T = 5:5:55$ (Matlab notation), and the performance degrades as the threshold increases. The situation is more complicated with JOMBANDI since there are two parameters to be set. However, from top right to bottom left, the points on the curves shown correspond to the following values for (Λ^c, Λ^b): (0.15, 1.0), (0.1, 1.0), (0.15, 4.0), (0.1, 4.0).

JOMBANDI was run with two model settings. The five-tap 3-D AR model had support defined by $\vec{q}_k = [0\,0\,-1]$, $[1\,0\,-1]$, $[0\,1\,-1]$, $[-1\,0\,-1]$, $[0, -1, -1]$, and the one-tap model $\vec{q}_k = [0\,0\,-1]$. In both cases, a block size of 9 × 9 pixels was employed with a two-pixel overlap between blocks. The SDIp detector (see [13] for details) was used to initialize the $b(\vec{x})$ field for JOMBANDI, using a threshold of ten grey levels. The output images were created by averaging the last 25 samples from a 50-iteration run of the Gibbs sampler on each frame.

The correct detection rate is measured as the fraction of pixels (out of the total number of missing sites) correctly set to 1 in $b(\vec{x})$. The false alarm rate is measured as the fraction of pixels incorrectly flagged as missing out of all the possible uncorrupted sites. First of all, it is interesting to note that the performance of the five-tap 3-D AR version of JOMBANDI is better than the one-tap version. For $\sigma_\mu^2 = 100$ (roughly 25 dB SNR), at a correct detection rate of about 70% for instance, the five-tap model gives a false alarm rate of about 0.1%, and the one-tap model gives a rate of 1%, which is a factor-of–10 difference. This illustrates the increased ability of the model with more support to cope with noise. At this same detection rate, for this level of noise, the SDIp gives a false alarm rate of 3%, which is too high to be useful.

Fig. 6 shows the result of JOMBANDI on three frames from the corrupted sequence. The middle row shows $(I(\vec{x}) - G(\vec{x})) + 128$ and illustrates more clearly what has been removed from the dirty image. The combined blotch rejection and noise reduction features are clear. That the rotating ball is not damaged is also important. In addition, note that the corruption level in the test sequence is very high, and in fact, corruption at the same site in consecutive frames does occur.

B. Noise Reduction Performance

Fig. 5 shows the decibel improvement in SNR after processing with JOMBANDI (five-tap 3-D AR model), the temporal Wiener filter [6], and temporal recursive frame averaging [4]. To separate out the noise reduction component of JOMBANDI from the missing data treatment component, the measurement of SNR was made only in those regions not corrupted by missing data. This does not, however, totally separate the two components since blotches can have an effect on processing for some distance *outside* their area.

The lowest curve shows the SNR of the degraded sequence at about 22 dB, and the top curve shows that JOMBANDI at $\Lambda^c = 0.15$, $\Lambda^b = 4.0$ performs best, doing 1 dB better than the other processes. Changing Λ^b to 1.0 makes JOMBANDI perform somewhere between the two temporal filters as far as noise reduction goes. This is sensible since a reduction in Λ^b implies that it is expected that blotches are less "convex," which is not the case. One feature, which is not shown by the curves, is that the purely temporal filters are prone to the "dirty window" effect, whereas JOMBANDI does not show this feature to a great extent.

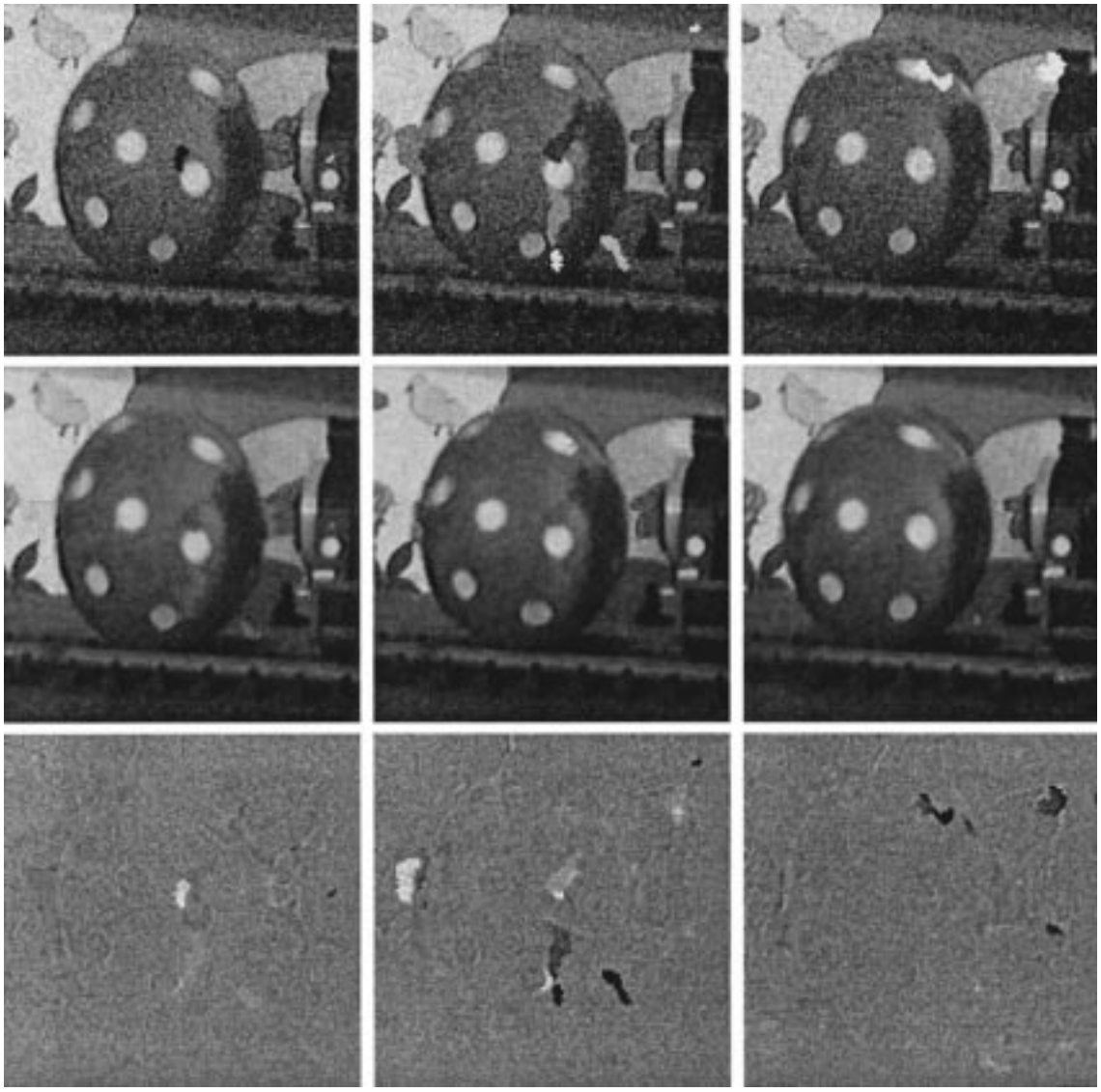

Fig. 6. Section of mobile and calendar sequence frames 6, 7, and 8. Top: Corrupted with blotches and $\sigma_\mu^2 = 100$. Middle: JOMBANDI result $\Lambda^c = 0.15$, $\Lambda^b = 1.0$. Bottom: Difference between top and middle frames offset by 128.

C. Real Example

Fig. 7 shows three frames from a dirty sequence supplied by RTP, Lisbon. Dark blotches can easily be identified as corrupted data by a human observer. The images have been brightened slightly to allow for better reproduction. There is visible grain noise on the images. The JOMBANDI algorithm was run for 50 iterations on the central frame (frame 8, top right of Fig. 7) using $\Lambda^c = 0.15$, $\Lambda^b = 4.0$, $\sigma_\mu^2 = 60.0$. The five-tap 3-D AR model was used, and otherwise, the same parameter settings as for the example in Fig. 6 were used.

The result of averaging the last 20 samples of the estimated image are shown as the right-hand image of the second row in Fig. 7. All the blotch artifacts are removed successfully, and the grain noise is substantially reduced, without excessive blurring. The bottom two images on the left in that figure show the detection result using SDIp and on the right, which is the last sample of $b(\vec{x})$ from JOMBANDI. The detection field has been configured successfully and has correctly rejected many of the false alarms of the deterministic process.

The top of Fig. 8 on the left shows the motion field used to kick start the JOMBANDI algorithm. One motion vector is shown per block, as well as the motion field mapping frame 8 into 7, and the images have been brightened to improve contrast for the vector icons. The image on the left is the dirty original frame 8, and on the right, the restored image is shown. The kick start motion field is noticeably distorted in regions of missing data, as expected and, in particular, appears to overestimate the motion in the upper part of the frame. After 50 iterations of JOMBANDI, however, this problem is much diminished (which is shown in right-hand image). The motion field appears to be more locally consistent and, even though the macro-rotational behavior in the top left-hand portion of the frame is wrong, the field is correct in the more highly textured areas where errors would cause more catastrophic effects in this case.

The middle two images on the left in Fig. 7 show the 50th sample of $c(\vec{x})$. Recall that the kick start for this sample is the original image. This explains the structure in the sample. The convergence of the overall algorithm is illustrated in the bottom

Fig. 7. Top row: Original frames 7 and 8 of degraded sequence 128 × 128. Middle row: Original frame 9, restored frame 8 using JOMBANDI. Bottom row: Detected blotches using SDIp (threshold = 10), JOMBANDI detection.

plot of number of pixels at which $b(\vec{x}) = 1$ in each iteration. By the last ten iterations, there is no visible change in the image.

The right-hand image on the middle row of Fig. 8 is extremely interesting. It shows $2(I(\vec{x}) - G(\vec{x})) + 128$ (magnified by 2.0 for better viewing). The removal of the grain noise is now more clearly apparent as is the remarkable removal of one line artifact. The removal of the blotches stands out as bright areas. What is very interesting is the structure of this difference image. There appears to be very little edge structure apparent (only at the top of the hat), implying that the noise reduction has not removed much image detail. Furthermore, the noise structure at large step edges is seen to be more correlated in a direction *parallel* to the edge (see bottom of image). This is a useful perceptual quality, in that damage caused by noise reduction is less visible at edges when the noise reduction is tuned parallel to the edge direction. This latter aspect of JOMBANDI is perhaps due to the ability of the process to tune itself to image details by adapting the model coefficients ($\mathbf{a}$) separately in each block.

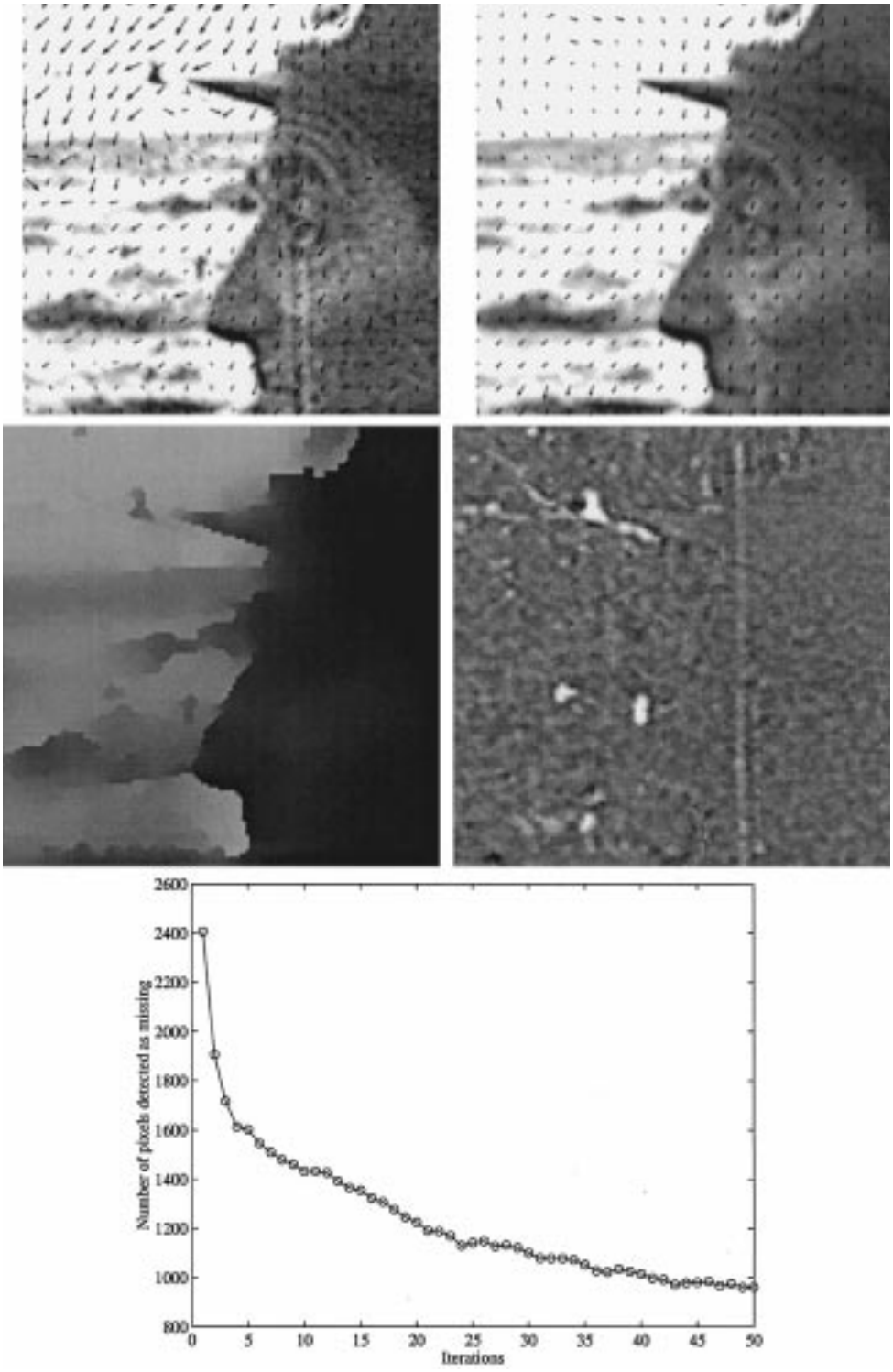

Fig. 8. Top row: Motion fields at iteration $=$ 0 and 50 of JOMBANDI. Middle row: Estimated $c(\vec{x})$. Difference between restored and original dirty images. Bottom row: Convergence of JOMBANDI.

Although the noise reduction here is effective, similar results can be obtained (as far as noise reduction performance is concerned) using alternative approaches, e.g., Wiener filters [5], [13] or temporally recursive filters [4]. It must be recognized that the power of JOMBANDI lies in its ability to draw information from all the different restoration and estimation process at the same time. There may be schemes that perform as well on each artifact separately, if it existed exclusively in the image, but this is the only process that has been designed specifically to treat all problems at the same time within a model-based framework.

XI. Computational Complexity

Running on a 400-MHz Pentium III and using a five-tap AR model as described above in the text, JOMBANDI takes about

10 s per iteration (of both joint sampling steps) on a 256×256 frame. This is, of course, without optimizing the code. This implies that one sweep of a full CCIR rec 601 video frame would be on the order of 1 min per iteration. Approximately 90% of the time is spent on the draw for $(\mathbf{a}, \sigma_e^2, \mathbf{d})$. This is understandable since this draw requires calculation and inversion of correlation matrices (order PN^2 operations, where the size of a block is $N \times N$ pixels) and sampling for motion (order $17PN^2$ operations[4]).

The latter step is more computationally intensive. The load could be reduced by periodically sampling for motion rather than sampling at every iteration. In addition, deterministic motion-estimation schemes are generally more robust to noise than missing data; therefore, it is possible to restrict the motion draw to those sites with detectable missing data.

A. Convergence

The matter of convergence and stopping criterion is typically problematic. In this case, because of the computational load of the algorithm, convergence as a stopping criterion takes second place to practical computing time for one frame. Running the algorithm for more than 50 iterations on an entire frame rapidly becomes totally impractical as the size of the frame increases. (See the previous discussion.)

Convergence can either be assessed by observing the samples from one of the variables or simply by looking at pictures. In practice, pictures from a wider range of material (than shown in this paper) are acceptable after about 30 iterations. Of course, looking at pictures is not viable in a real system using batch processing, and a workable alternative is to observe the number of pixels flagged as missing. When the mean of that number varies slowly over the last ten iterations, the algorithm is stopped since this indicates that the field $b(\vec{x})$ is not changing significantly. We do not have the space here to examine convergence and stopping criteria in detail.

In all the examples shown, a fixed number of iterations were employed, with a burn in of ten iterations. This allowed uncomplicated performance comparisons (in a reasonable compute time) across the wide range of artifact levels that were used. The real example employed the stopping criterion outlined above.

B. What Makes JOMBANDI Different?

Quite apart from its superior performance as a blotch detector in noisy conditions, JOMBANDI differs from all these other proposals in that it allows the dynamic interaction between the variables both in terms of their effect on the degradation and in terms of spatial correlation. In all previous work in blotch treatment in particular, decisions tend to be taken at sites without any acknowledgment of the correlation between those sites inherent in the degradation. Any spatial processing is typically done as a post-process. In JOMBANDI, however, this spatial interaction is explored during the iterative stages of the algorithm itself.

Furthermore, through the Gibbs sampler, JOMBANDI generates *samples* from the underlying textural probability distribution in each frame. This is very important for textural resynthesis, and it is well known that least-squares interpolants for missing data tend to look *dull* [13], [23], [34].

[4]Seventeen candidate motion vectors sampled per block.

XII. Final Comments

This paper has proposed a scheme for the joint "restoration" of image sequences, as far as noise reduction and missing data removal are concerned. Its ability to correct motion simultaneously with reconstructing the underlying image is fundamental to its robust behavior. It is acknowledged, however, that the absolute noise-reduction performance of the system is not vastly different from many other noise-reduction schemes in existence today, particularly the successful wavelet algorithms. This is principally because the pixel-wise update scheme would take a large number of iterations to take advantage of the same kind of information presented by explicit scale/frequency techniques. As outlined in previous sections, it is possible to repose JOMBANDI to perform noise reduction over a large block of pixels. In that case, elements of wavelet-based noise reduction can be brought into play. This is a matter for further work.

It is the way in which the Bayesian framework has allowed the coherent design of this joint process that is of interest here. To our knowledge, this is the only scheme that has quantitatively combined solutions to the noise reduction and missing data problems. Furthermore, the introduction of the joint sampling process using composition sampling is of pivotal importance in this work both from the point of view of computational simplicity and increased convergence of the iterative scheme. Finally, the use of *candidate selection* as part of the overall MCMC scheme (in the draw for motion) shows how good aspects from a deterministic scheme can be combined positively with MCMC.

Appendix A
Composition Sampling for $b(\vec{x})$, $c(\vec{x})$, $I_n(\vec{x})$

In the adapted Gibbs sampler, it is required to draw a *joint* sample for $b(\vec{x})$, $c(\vec{x})$, $I_n(\vec{x})$ from the conditional distribution for these variables, given the observed data and other parameters, e.g., $\vec{d}$, $\mathbf{a}$, using composition sampling. In what follows, these other (given) parameters are denoted by $(\cdot)$.

Using a pixel-wise draw allows the simplest implementation. Dropping the $\vec{x}$ arguments (since all these variables are at coincident sites in the image) and using $i = I_n(\vec{x})$; $g_n = G_n(\vec{x})$, the recipe for drawing a sample (b^0, i^0, c^0) is as follows:

$$\begin{aligned} \text{Using } \; p(b,i,c|\cdot) &= p_c(c|i,b,\cdot)p_i(i|b,\cdot)p_b(b|\cdot) \\ b^0 &\sim p_b(b|\cdot) \\ i^0 &\sim p_i(i|b^0,\cdot) \\ c^0 &\sim p_c(c|b^0,i^0,\cdot). \end{aligned}$$

There are other factorizations that could be used, but this one allows straightforward analysis.

The conditionals are derived by integrating out c first and then i from the posterior distribution for all the three variables. What complicates matters is that b is a binary variable appearing as a *switch* in the likelihood function, but this can be dealt with by treating the two cases of b as part of the integration process.

Consider first the derivation of the conditional $p(i|b,\cdot)$

$$p(i|b,\cdot)=\frac{p(i,b|\cdot)}{\int_i p(i,b|\cdot)\,di}=\frac{\int_c p(c,i,b|\cdot)\,dc}{\int_i p(i,b|\cdot)\,di}. \quad (23)$$

The term in the denominator is the typical normalization term, but this follows naturally once the integral in the numerator can be performed. This is discussed next.

A. Integrating Out c

At this point, it is useful to recall that

$$p(c,\,i,\,b|\cdot)\propto p_l(g_n|i,\,c,\,b,\,\cdot)p_i(i|\cdot)p_c(c|C)p_b(b|B) \quad (24)$$

where B, C denote the set of values of b, c at the eight nearest-neighbor sites of $\vec{x}$. Note that $p_i(i|\cdot)$ is always independent of c and b.

Considering the two cases of $b=0,\,1$, there are two numerator integrals to be evaluated in (23). For $b=1$, the integral is

$$\int_c p(c,i,b|\cdot)\,dc = \int_c p_l(g_n|c,b=1,\cdot)p_i(i|\cdot)p_c(c|C)p_b(b=1|B)\,dc. \quad (25)$$

For $b=0$, the integral is

$$\int_c p(c,i,b|\cdot)\,dc = \int_c p_l(g_n|i,b=0,\cdot)p_i(i|\cdot)p_c(c|C)p_b(b=0|B)\,dc. \quad (26)$$

In the second case, $b=0$, the only function involving c is $p_c(c|C)$, and since $\int_c p_c(c|C)dc=1$, the integration is straightforward.

In the first case, the integration is more involved since there are two functions involving c. Therefore

$$\int_c p_l(g_n|c,b=1,\cdot)p_i(i|c,b=1,\cdot)p_c(c|C)p_b(b=1|B)\,dc \propto \int_c p_l(g_n|c,b=1,\cdot)p_c(c|C)\,dc. \quad (27)$$

Continuing

$$\int_c p_l(g_n|c,b=1,\cdot)p_c(c|C)\,dc\propto \int_c\left[\exp\left(\frac{-(g_n-c)^2}{2\sigma_\mu^2}\right)\exp\left(-\sum_k\Lambda_k^c(c-c(k))^2\right)\right]dc \quad (28)$$

where $\Lambda_k^c=\lambda_k^c(1-u(\vec{x},\,\vec{x}+\vec{v}_k))$ from (11). Rearranging the right-hand side yields

$$\int_c \exp-\left[\frac{g_n^2-2cg_n+c^2}{2\sigma_\mu^2} +c^2\sum_k\Lambda_k^c-2c\sum_k\Lambda_k^c c(k)+\sum_k\Lambda_k^c(c(k))^2\right]dc = \int_c\exp-\left[c^2\left(\frac{1}{2\sigma_\mu^2}+\sum_k\Lambda_k^c\right) -2c\left(\frac{g_n}{2\sigma_\mu^2}+\sum_k\Lambda_k^c c(k)\right)+\frac{g_n^2}{2\sigma_\mu^2}+\sum_k\Lambda_k^c(c(k))^2\right]dc.$$

Completing the square with respect to c in the arguments of the exponentials, the *integrand* can be expressed as

$$\exp-\left[\left(\frac{1}{2\sigma_\mu^2}+\sum_k\Lambda_k^c\right)\left(c-\frac{\frac{g_n}{2\sigma_\mu^2}+\sum_k\Lambda_k^c c(k)}{\frac{1}{2\sigma_\mu^2}+\sum_k\Lambda_k^c}\right)^2\right]\times\exp-K_c \quad (29)$$

where K_c represents all the terms not including c. After some simplification, the integral can be expressed as

$$\int_c(\cdots)\,dc \propto\int_c\mathcal{N}\left(\frac{\hat{c}\sigma_\mu^2+g_n\sigma_c^2}{\sigma_\mu^2+\sigma_c^2},\,\frac{\sigma_\mu^2\sigma_c^2}{\sigma_\mu^2+\sigma_c^2}\right)\exp(-K_c)\,dc \quad (30)$$

where $\mathcal{N}$ is the Normal distribution, and the various introduced constants are as defined in (20).

Because $\int_c\mathcal{N}(\cdot)dc=1$, the result of the integration is therefore $\propto\exp-K_c$, where K_c depends on Λ_k^c, σ_μ^2, σ_c^2, $\hat{c}$, etc.

Defining

$$\overline{c}=\frac{\hat{c}\sigma_\mu^2+g_n\sigma_c^2}{\sigma_\mu^2+\sigma_c^2} \quad (31)$$

allows a much more easily evaluated form of the result [13], [23], [35] by simply substituting this value (which is the least squares estimate for c given all the other variables) into (29) and, hence, (28), as follows for $b=1$:

$$\int_c p_l(g_n|c,b=1,\cdot)p_i(i|\cdot)p_c(c|C)p_b(b=1|B)\,dc = p_l(g_n|c=\overline{c},b=1,\cdot)p_i(i|\cdot)p_c(c=\overline{c}|C)p_b(b=1|B). \quad (32)$$

Thus, we have, for $b=1,\,0$, respectively

$$p(i,b|\cdot)=\begin{cases}p_l(g_n|c=\overline{c},b=1,\cdot)p_i(i|\cdot)p_c(c=\overline{c}|C)p_b(b=1|B)\\ p_l(g_n|i,b=0,\cdot)p_i(i|\cdot)p_b(b=0|B).\end{cases} \quad (33)$$

From this, expression $p(i|b,\,\cdot)$ follows by substitution into (23). The actual forms for $p(i|b)$ are derived by completing the square where necessary w.r.t. i, to yield the expressions shown in (18).

Integrating out i from the expressions above will now allow the derivation of $p(b|\cdot)$.

B. Integrating Out i

In the case that $b=1$ [in (33)], it can be seen that there is just one expression involving i [$p_i(i|\cdot)$], and again, that integration is straightforward.

For the case $b=0$, the situation is more involved. To proceed, it is necessary to write $p_i(i|\cdot)$ using the pseudo-likelihood as follows:

$$p_i(i|\cdot)\propto\exp-\left(\frac{[i\mathbf{a}_u+\mathbf{A}_k\mathbf{i}_k]^T[i\mathbf{a}_u+\mathbf{A}_k\mathbf{i}_k]}{2\sigma_e^2}\right). \quad (34)$$

The vector $\mathbf{i}_k$ contains all the pixel sites that include i (at $\vec{x}$) in their 3-D AR support. If these sites are defined as $(\vec{x}_s)$, then $\mathbf{a}_u$, $\mathbf{A}_k$ are arranged so that $[i\mathbf{a}_u+\mathbf{A}_k\mathbf{i}_k]$ is a column vector

containing the residuals $e(\vec{x}_s)$ as well as the site $(\vec{x})$ at which i is to be estimated.

At this stage, it is useful to note that by completing the square w.r.t. i on the right-hand side of (34), it follows that the conditional expression is actually a Normal distribution as follows:

$$p_i(i|\cdot) \propto \exp - \left(\frac{i - \dfrac{\mathbf{a}_u^T \mathbf{A}_k \mathbf{i}_k}{\mathbf{a}_u^T \mathbf{a}_u}}{2\sigma_i^2} \right) \tag{35}$$

where $\sigma_i^2 = \sigma_e^2 / \mathbf{a}_u^T \mathbf{a}_u$, as in (20).

To proceed with integrating out i, the integral to be evaluated is

$$\int_i f(i) p_b(b=0|B)\, di = \int_i p_l(g_n|i, b=0, \cdot) p_i(i|\cdot) p_b(b=0|B)\, di. \tag{36}$$

Since $p_l(\cdot)$ is a Gaussian, the integration proceeds by substituting for $p_i(\cdot)$ from either (35) or (34) and then completing the square w.r.t. i. This is done in a similar manner to that shown previously for manipulating the expression for c. After some simplification, the following results:

$$f(i) = \mathcal{N}\left(\bar{i},\ \frac{\sigma_\mu^2 \sigma_i^2}{\sigma_\mu^2 + \sigma_i^2} \right) \exp - K_i \tag{37}$$

where the introduced constants have their values as shown in (20).

Again, following the same ideas that led to the solution for integrating out c, the final result (for $b = 0$) is best expressed as

$$\int_i f(i) p_b(b=0|B)\, di = p_l(g_n|i=\bar{i}) p_i(i=\bar{i}|\cdot) p_b(b=0|B). \tag{38}$$

Hence

$$p(b|\cdot) = \begin{cases} p_l(g_n|c=\bar{c}, b=1, \cdot) p_c(c=\bar{c}|C) p_b(b=1|B), & [b=1] \\ p_l(g_n|i=\bar{i}, b=0, \cdot) p_i(i=\bar{i}|\cdot) p_b(b=0|B), & [b=0] \end{cases}$$

as required, where again, variables are defined as in (20).

APPENDIX B
NOTE ON $p_c(c|C)$

From the prior for c [see (11)], we can assemble the form for the probability distribution $p(c|i,\ b,\ \cdot)$ (in the case that $b = 0$) by completing the square in terms of c in the prior. This form is given by $\mathcal{N}(\hat{c},\ \sigma_c^2)$, where the constants are defined in (20).

REFERENCES

[1] A. Kokaram, R. Morris, W. Fitzgerald, and P. Rayner, "Detection of missing data in image sequences," *IEEE Trans. Image Processing*, vol. 4, pp. 1496–1508, Nov. 1995.

[2] ——, "Interpolation of missing data in image sequences," *IEEE Trans. Image Processing*, vol. 4, pp. 1509–1519, Nov. 1995.

[3] P. V. M. Roosmalen, A. Kokaram, and J. Biemond, "Noise reduction of image sequences as preprocessing for mpeg2 encoding," in *Proc. Eur. Conf. Signal Process.*, vol. 4, Sept. 1998, pp. 2253–2256.

[4] E. Dubois and S. Sabri, "Noise reduction in image sequences using motion compensated temporal filtering," *IEEE Trans. Commun.*, vol. COMM-32, pp. 826–831, July 1984.

[5] A. Erdem, M. Sezan, and M. Özkan, "Motion-compensated multiframe Wiener restoration of blurred and noisy image sequences," in *Proc. IEEE ICASSP*, vol. 3, Mar. 1992, pp. 293–296.

[6] A. Katsagellos, J. Driessen, S. Efstratiadis, and R. Lagendijk, "Spatio–temporal motion compensated noise filtering of image sequences," *Proc. SPIE VCIP*, pp. 61–70, 1989.

[7] M. Özkan, A. Erdem, M. Sezan, and A. Tekalp, "Efficient multiframe Wiener restoration of blurred and noisy image sequences," *IEEE Trans. Image Processing*, vol. 1, pp. 453–476, Oct. 1992.

[8] R. Kleihorst, G. de Haan, R. Lagendijk, and J. Biemond, "Motion compensated noise filtering of image sequences," in *Signal Processing VI*. New York: Elsevier, 1992, pp. 1385–1388.

[9] P. V. M. Roosmalen, R. L. Lagendijk, and J. Biemond, "Noise reduction for image sequences using an oriented pyramid thresholding technique," in *Proc. IEEE Int. Conf. Image Process.*, vol. 1, Sept. 1996, pp. 275–378.

[10] R. Storey, "Electronic detection and concealment of film dirt," *SMPTE J.*, pp. 642–647, June 1985.

[11] M. J. Nadenau and S. K. Mitra, "Blotch and scratch detection in image sequences based on rank ordered differences," in *Proc. 5th Int. Workshop Time-Varying Image Process. Moving Object Recognit.*, Sept. 1996.

[12] G. R. Arce, "Multistage order statistic filters for image sequence processing," *IEEE Trans. Signal Processing*, vol. 39, pp. 1146–1161, May 1991.

[13] A. C. Kokaram, *Motion Picture Restoration: Digital Algorithms for Artifact Suppression in Degraded Motion Picture Film and Video*. New York: Springer Verlag, 1998.

[14] A. Kokaram and P. Rayner, "A system for the removal of impulsive noise in image sequences," *SPIE Visual Communications and Image Processing*, pp. 322–331, Nov. 1992.

[15] A. Kokaram and S. Godsill, "A system for reconstruction of missing data in image sequences using sampled 3D AR models and MRF motion priors," in *Proc. Eur. Conf. Comput. Vision*, Apr. 1996, pp. 613–624.

[16] S. Godsill and A. Kokaram, "Joint interpolation, motion and parameter estimation for degraded image sequences with missing data," *Signal Process. VIII*, vol. I, pp. 1–4, Sept. 1996.

[17] A. Kokaram and S. Godsill, "Joint detection, interpolation, motion and parameter estimation for image sequences with missing data," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 1997, pp. 191–194.

[18] R. D. Morris and W. J. Fitzgerald, "Detection and correction of speckle degradation in image sequences using a 3D markov random field," in *Proc. Int. Conf. Image Process.: Theory Appl.*, June 1993.

[19] ——, "Stochastic and deterministic methods in motion picture restoration," in *Proc. Int. Workshop Image Process.*, June 1994.

[20] ——, "Replacement noise in image sequences, detection and interpolation by motion field segmentation," in *Proc. IEEE Int. Conf. Acoust., Signal Processing (ICASSP)*, 1994.

[21] P. Strobach, "Quadtree-structured linear prediction models for image sequence processing," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 742–747, July 1989.

[22] S. Efstratiadis and A. Katsagellos, "A model based, pel-recursive motion estimation algorithm," in *Proc. IEEE ICASSP*, 1990, pp. 1973–1976.

[23] J. J. O. Ruanaidh and W. J. Fitzgerald, *Numerical Bayesian Methods Applied to Signal Processing*. New York: Springer-Verlag, 1996.

[24] J. Konrad and E. Dubois, "Bayesian estimation of motion vector fields," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, Sept. 1992.

[25] C. Stiller, "Motion-estimation for coding of moving video at 8kbit/sec with Gibbs modeled vectorfield smoothing," *SPIE VCIP*, vol. 1360, pp. 468–476, 1990.

[26] D. Marr, *Vision*. San Francisco, CA: W. H. Freeman, 1982.

[27] G. E. P. Box and G. C. Tiao, *Bayesian Inference in Statistical Analysis*. Reading, MA: Addison-Wesley, 1973.

[28] J. Liu, W. H. Wong, and A. Kong, "Covariance structure of the Gibbs sampler with applications to the comparison of estimators and augmentation schemes," *Biometrika*, vol. 81, pp. 27–40, 1994.

[29] M. A. Tanner, *Tools for Statistical Inference*. New York: Springer-Verlag, 1996.

[30] J. Besag, "On the statistical analysis of dirty pictures," *J. R. Stat. Soc. B*, vol. 48, pp. 259–302, 1986.

[31] A. Kokaram, "3D Wiener filtering for noise suppression in motion picture sequences using overlapped processing," *Signal Process. V, Theories Appl.*, pp. 1780–1783, Sept. 1994.

[32] M. Özkan, M. Sezan, and A. Tekalp, "Motion-adaptive weighted averaging for temporal filtering of noisy image sequences," *SPIE Image Process. Algorithms Techn. III*, pp. 201–212, Feb. 1992.

[33] E. Abreu, M. Lightstone, S. K. Mitra, and K. Arakawa, "A new efficient approach for the removal of impulsive noise from highly corrupted images," *IEEE Image Processing*, vol. 6, pp. 1012–1025, June 1996.
[34] A. Kokaram, "Detection and removal of line scratches in degraded motion picture sequences," *Signal Process. VIII*, vol. I, pp. 5–8, Sept. 1996.
[35] D. MacKay, "Bayesian methods for adaptive models," Ph.D. dissertation, Calif. Inst. Technol., Pasadena, 1992.

Anil C. Kokaram (M'92) was born in Trinidad. He received the B.S. degree electrical and information sciences from the University of Cambridge, Cambridge, U.K., in 1989. He then joined the Signal Processing Group, Cambridge University Engineering Department, and in 1993, he received the Ph.D. degree in engineering for his thesis entitled "Motion picture restoration."

He was then a Research Associate with that group, and later on, he was appointed to a Fellowship of Churchill College, Cambridge. During 1993 to 1998, he worked on the EU project "Automated restoration of original film and video archives" (AURORA). In 1998, he was appointed to a Lectureship in the Electronic and Electrical Engineering Department, Trinity College, Dublin, Ireland. He was recently awarded a Fellowship from that institution. His research interests lie principally in the area of image and video processing including motion estimation, texture synthesis, and video and film restoration/post-production in particular. His fundamental research areas include Bayesian methods in signal processing and fast (practical) algorithms for MCMC methodology used for video material. He has recently diversified into such application areas as multimedia over wireless and multimedia information retrieval. He is coordinator of the EU project "Models for unified multimedia information retrieval" (www.moumir.org). He has published over 40 papers in refereed journals and conference proceedings, including the book *Motion Picture Restoration* (New York: Springer Verlag, 1998).

Simon J. Godsill (M'95) received the B.S. degree in electrical and information sciences in 1988 and the Ph.D. degree in engineering for a thesis entitled "Digital audio restoration" in 1994 from the University of Cambridge, Cambridge, U.K.

He led the technical development team at the newly formed CEDAR Audio Ltd., researching digital signal processing algorithms for enhancement of degraded sound recordings. He is currently a director of CEDAR. In 1996, he was appointed as a University Lecturer in Information Engineering and Fellow of Corpus Christi College and, in 2001, to a Readership in statistical signal processing. Research specialties include Bayesian statistical methods in signal processing, Bayesian computational techniques (in particular, Markov chain methods and particle filters), audio and music signal analysis, image processing, and signal processing for nonlinear/non-Gaussian environments. He has published over 50 papers in refereed journals, conference proceedings, and books, including a research text, co-authored with P. Rayner, entitled *Digital Audio Restoration: A Statistical Model-Based Approach* (New York: Springer).