

# Temporal Resolution Enhancement in Compressed Video Sequences

Mark A. Robertson and Robert L. Stevenson

**ABSTRACT** — Compressed video may possess a number of artifacts, both spatial and temporal. Spatial compression artifacts arise as a result of quantization of the transform-domain coefficients, and are often manifested as blocking and ringing artifacts. Temporal limitations in compressed video occur when the encoder, in an effort to reduce bandwidth, drops frames. Omitting frames decreases the reconstructed frame rate, which can cause motion to appear jerky and uneven. This paper discusses a method to increase the frame rate of video compressed with the DCT by inserting images between received frames of the sequence. The Bayesian formulation of the restoration prevents spatial compression artifacts in the received frames from propagating to the reconstructed frames.

**Keywords** — Temporal Interpolation, DCT Quantization Noise, Compressed Video, Postprocessing

## 1 INTRODUCTION

Two of the most important limitations of compressed video are spatial compression artifacts and missing frames. Spatial compression artifacts commonly take the form of blocking and ringing, and are annoying in their own right. Spatial compression artifact removal has been well-researched, a few examples of which can be found in [10, 11, 17, 18]. On the other hand, temporal limitations in compressed video are often caused by missing frames, and can result in sequence motion appearing uneven.

In a complete scheme for post-processing compressed video, missing frames should be estimated *and* received frames should be enhanced. However, the focus here is the *temporal* aspect, and this paper deals exclusively with the restoration of the missing frames at the decoder (see [11] for treatment of both received and inserted frames). This paper does, however, explicitly consider the spatial compression artifacts in the received frames so that their effect on the reconstructed missing frames is minimized.

The problem at hand can be stated as follows: An encoder has compressed a sequence of video frames. However, due to bandwidth or storage limitations, the encoder may have had to drop frames prior to encoding, e.g., dropping two out of every three frames to go from 30 fps to 10 fps. In scenes with moderate to high motion, the dropped frames may cause motion to appear jerky and uneven. Furthermore, the frames that are transmitted are

compressed with lossy transform coding, which is taken to incorporate the block discrete cosine transform in this paper. Lossy compression of frames can result in poor received image quality. The goal for temporal restoration of compressed video is to reconstruct the missing frames of the sequence given the noisy received frames, and, for the example just given, go from a 10-fps sequence to a 30-fps sequence. Naturally, one desires the reconstructions to be free of the compression artifacts that may plague the received frames.

All effective forms of restoring missing video frames take scene motion into account, and for this reason the problem will be referred to as Frame Rate Adjustment by Motion Estimation, or *FRAMEstimation* for short. A number of *FRAMEstimation* schemes have been reported in the literature; a few examples are [1, 2, 14–16]. However, none of the methods appear to address the quality of the *received* frames, which seems odd because low bit rate video is one of the target applications of many *FRAMEstimation* procedures (due to the low frame rates of heavily compressed video), and low bit rate video typically also has spatial compression artifacts. If not properly considered, these spatial compression artifacts can propagate to the inserted frames of a *FRAMEstimation* procedure. The formulation presented here explicitly considers the compression noise that is present in the received frames.

Restoration of dropped frames of video is quite different than restoration of received frames. In the latter case, compressed versions of the frames under consideration are available, and the goal is to eliminate the spatial compression artifacts in the received image. For dropped frames, however, the missing images must be restored based only on other surrounding images in the sequence.

A maximum *a posteriori* (MAP) approach is taken here to estimate the missing frames. However, since received data are not available for the frame being restored, the observations of *different* frames must be related to the inserted frame. The MAP criterion in this case leads to

$$\hat{\mathbf{z}} = \arg \max_{\mathbf{z}} p(\mathbf{z} | \mathbf{z}_q^k, k \in \mathcal{K}), \quad (1)$$

$$= \arg \max_{\mathbf{z}} p(\mathbf{z}) p(\mathbf{z}_q^k, k \in \mathcal{K} | \mathbf{z}), \quad (2)$$

where  $\mathbf{z}$  is the frame to be inserted and  $\mathbf{z}_q^k$  is the  $k^{\text{th}}$  received frame, which contains quantization noise. (Note that throughout this paper, two-dimensional images are represented as vectors by using a stacked-column notation.) The index set  $\mathcal{K}$  includes all frame indices of the received sequence that are being used to reconstruct the missing image. In this work, only the received frames immediately before and immediately after the missing image are used, but the framework allows for inclusion of

---

Manuscript submitted 27 July 2001, revised 27 September 2001  
The authors are with the Laboratory for Image and Signal Analysis (LISA) at the University of Notre Dame, Notre Dame, Indiana, 46556 U.S.A.

Contact author: Robert L. Stevenson, rls@nd.edu

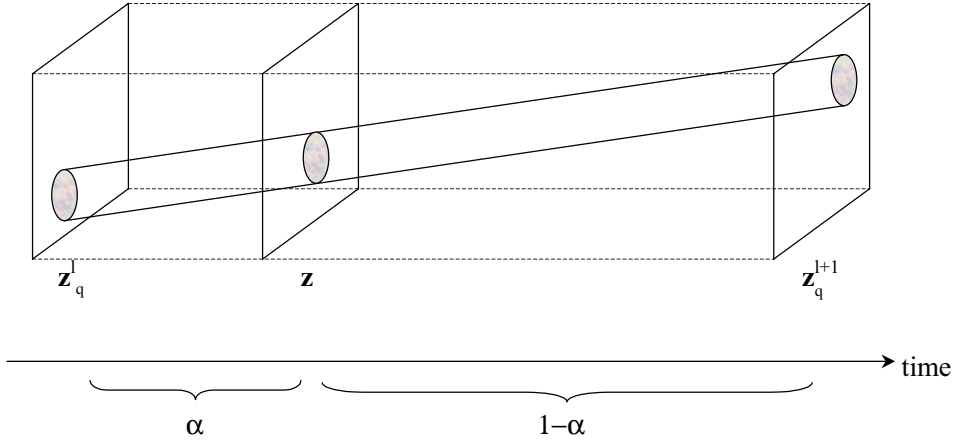


Figure 1: Configuration of the received frames  $\mathbf{z}_q^l$  and  $\mathbf{z}_q^{l+1}$  with respect to the inserted frame  $\mathbf{z}$ .

as many received frames as desired. Figure 1 shows the configuration of the missing frame with respect to the received frames.

The first term of (2) is the *a priori* image model, and is discussed in the next section. The latter term of (2) is the observation model that relates observations to the data being estimated, and consists of two elements—the quantization noise model, discussed in Section 3, and the temporal observation model, discussed in Section 4. When the terms from the prior image model and the observation model are determined, the MAP criterion of (2) results in a convex optimization problem, the solution to which is discussed in Section 5 along with experimental results. Section 6 concludes the paper.

## 2 PRIOR IMAGE MODEL

The prior image model is chosen as a Huber Markov Random Field (HMRF), which has been used extensively in image and video processing [10, 13]. The Huber function is a convex function that has edge-preserving properties relative to a simple quadratic. Details of the HMRF can be found in the papers just cited, and is given here without excessive discussion,

$$p(\mathbf{z}) = \frac{1}{G} \exp \left\{ -\lambda_F \sum_{c \in \mathcal{C}} \rho_T(\mathbf{d}_c^t \mathbf{z}) \right\}, \quad (3)$$

where  $G$  is a normalizing constant,  $\lambda_F$  is a regularization parameter, and the Huber function  $\rho_T(\cdot)$  is defined as

$$\rho_T(u) = \begin{cases} u^2, & |u| \leq T, \\ T^2 + 2T(|u| - T), & |u| > T. \end{cases} \quad (4)$$

The  $c$  of (3) are local groups of pixels called cliques, and  $\mathcal{C}$  is the set of all such cliques, which depends on the neighborhood structure of the MRF. Here, the vectors  $\mathbf{d}_c$  are chosen to extract the differences between a pixel and its neighbors, such that (3) simplifies to

$$p(\mathbf{z}) = \frac{1}{G} \exp \left\{ -\lambda_F \sum_{n=0}^{N-1} \sum_{m \in \mathcal{N}_n} \rho_T(z[n] - z[m]) \right\}, \quad (5)$$

where  $\mathcal{N}_n$  is the index set of neighbors for the  $n^{\text{th}}$  pixel, and  $N$  is the number of pixels in the image. The inner summation in (5) is over each pixel in the neighborhood of the  $n^{\text{th}}$  pixel. A neighborhood consisting of a pixel's eight nearest neighbors is used here.

Using the HMRF encourages smoothness in the final image reconstruction, since the probability in (3) is higher for smoother images. The Huber function penalizes differences less than  $T$  quadratically; however, differences larger than  $T$  are only penalized linearly, which helps prevent oversmoothing image edges. In this work, the HMRF smooths compression artifacts introduced from the received frames, preventing their presence in the estimates of missing frames. The prior image model also fills in gaps in the inserted image that may result when no motion information is present, which is discussed further in Section 4.

## 3 DCT QUANTIZATION NOISE

It is important to characterize the compression noise in received frames, because the received frames are the only observations from which the missing frames can be estimated. After compression and transmission, the receiver decodes the observed image  $\mathbf{z}_q^k$  at time  $k$ , where the “ $q$ ” subscript denotes that  $\mathbf{z}_q^k$  has quantization error relative to the original noise-free image  $\mathbf{z}^k$ ,

$$\mathbf{z}_q^k = \mathbf{z}^k + \mathbf{e}_z^k. \quad (6)$$

Some researchers [8, 9] assume this model with an independent and identically-distributed (IID) Gaussian assumption for the noise terms  $\mathbf{e}_z^k$ . However, a spatially-invariant noise term does not realistically model the quantization error, and a more sophisticated and accurate quantization noise model is used here instead.

Compression noise is derived here explicitly for compression methods that use the discrete cosine transform (DCT), due to the DCT's popularity in image and video compression standards. To determine statistics of the spatial-domain quantization error, begin by considering

the DCT prior to quantization,

$$\mathbf{y}^k = \mathcal{D}(\mathbf{z}^k - \mathbf{z}_{mc}^k), \quad (7)$$

where  $\mathbf{y}^k$  is the frequency-domain representation of  $\mathbf{z}^k$ ,  $\mathcal{D}$  represents the matrix that performs the two-dimensional block DCT, and  $\mathbf{z}_{mc}^k$  is the motion-compensating prediction for the video frame. Note that the DCT is applied on a block-by-block basis, so  $\mathcal{D}$  is a block-diagonal matrix.

The DCT coefficients are quantized as

$$\mathbf{y}_q^k = Q^k[\mathbf{y}^k], \quad (8)$$

where  $Q^k[\cdot]$  is the quantization operator for the  $k^{\text{th}}$  frame. The decoded image is the motion-compensated prediction added to the inverse DCT (IDCT) of the quantized frequency values,

$$\mathbf{z}_q^k = \mathcal{D}^t \mathbf{y}_q^k + \mathbf{z}_{mc}^k, \quad (9)$$

where  $\mathcal{D}^t$  is the transpose of  $\mathcal{D}$ , which is also the inverse of  $\mathcal{D}$  due to the unitary property of the DCT. Note that the spatial-domain quantization error can be expressed as

$$\mathbf{e}_z^k = \mathbf{z}_q^k - \mathbf{z}^k = \mathcal{D}^t(\mathbf{y}_q^k - \mathbf{y}^k). \quad (10)$$

Of primary importance here are the statistics of the noise. The autocovariance matrix for the noise is easily shown to be

$$\mathbf{K}_{\mathbf{e}_z^k} = E[\mathbf{e}_z^k \mathbf{e}_z^{k,t}] = \mathcal{D}^t \mathbf{K}_{\mathbf{e}_y^k} \mathcal{D}, \quad (11)$$

where  $\mathbf{K}_{\mathbf{e}_y^k}$  is the autocovariance matrix of the DCT-domain noise. Due to the decorrelating properties of the DCT for typical images [7], the autocovariance matrix of  $\mathbf{y}^k$  is approximately diagonal; along with some symmetry conditions on the distributions of  $\mathbf{y}^k$ , the DCT-domain quantization noise autocovariance matrix  $\mathbf{K}_{\mathbf{e}_y^k}$  can be shown to be diagonal as well. The diagonal elements of  $\mathbf{K}_{\mathbf{e}_y^k}$  represent the variances of the frequency-domain quantization errors, and can be found by considering the quantization intervals: If a particular observed DCT value  $y_q[m]$  lies in the quantization interval  $[q_i, q_{i+1})$  defined by the quantizer, then barring any prior knowledge on  $y[m]$  one concludes that  $y[m]$  given the observation is uniformly distributed in  $[q_i, q_{i+1})$ , and the variance of the frequency-domain quantization error is easily seen to be  $\frac{1}{12}(q_{i+1} - q_i)^2$ . Thus the matrix  $\mathbf{K}_{\mathbf{e}_y^k}$  is easily constructed based on the quantization limits defined by the quantizer. Note from (11) that in general the spatial-domain noise terms are *not* IID, but are correlated.

From (10), each spatial-domain noise term is a linear combination of independently distributed uniform random variables (64 of them, when using the  $8 \times 8$  block DCT), allowing the spatial-domain quantization noise to be approximated as a  $\mathbf{0}$ -mean Gaussian random vector with autocovariance matrix  $\mathbf{K}_{\mathbf{e}_z^k}$ . From (6),

$$p(\mathbf{z}_q^k | \mathbf{z}^k) = \frac{(2\pi)^{-\frac{N}{2}}}{|\mathbf{K}_{\mathbf{e}_z^k}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{z}_q^k - \mathbf{z}^k)^t \mathbf{K}_{\mathbf{e}_z^k}^{-1} (\mathbf{z}_q^k - \mathbf{z}^k)\right\}. \quad (12)$$

#### 4 TEMPORAL OBSERVATION MODEL

The DCT quantization noise model of the previous section is one component of the overall observation model. This section provides the final component: The temporal observation model. The goal is to relate the image to be inserted to the observations, which is not a trivial task due to motion of objects between frames—the location of objects in the observed frames must be related to the location of objects in the unknown frame to be inserted. It thus seems obvious that knowledge of *motion* is necessary. A regularized block-based motion model is used in this work, with  $4 \times 4$  block sizes. Motion estimation is performed on the received frames *after* they have been enhanced with a post-processing algorithm; details of both the motion-estimation procedure and the post-processing algorithm can be found in [11]. However, the primary focus of this paper is not motion estimation, and the FRAMEstimation algorithm discussed here is not dependent on the type of motion estimation performed. For these reasons, it will be assumed that motion vectors are already known between received frames; these motion vectors will then be used to establish motion to the inserted frame, as shown in the following.

Assume that the unavailable frame for which motion information is desired is located between frames  $l$  and  $l+1$ , the temporal distance between which is normalized to unity. Assume that the intermediate frame  $\mathbf{z}$  is situated such that the distance between the frame and frame  $l$  is  $\alpha$ , the distance between the frame and frame  $l+1$  is  $1-\alpha$ , and  $0 < \alpha < 1$ , as shown in Fig. 1. The observation model will be described for the  $l^{\text{th}}$  frame only, since the observation model for frame  $l+1$  is completely analogous to that of frame  $l$ .

Suppose that motion information for each pixel at frame  $l$  with respect to frame  $l+1$  has already been determined from a motion-estimation algorithm. Note that some of the motion vectors from  $l$  to  $l+1$  may point off the screen, which is perfectly fine, because some of these off-screen motion vectors from  $l$  to  $l+1$  may result in legitimate motion (not off-screen) for the intermediate frame.

If the 2-D motion vector from  $l$  to  $l+1$  for pixel  $i$  is  $\mathbf{v}_i^{l,l+1} = [u_i^{l,l+1}, v_i^{l,l+1}]^t$ , then the 2-D motion vector from  $l$  to the intermediate frame is found simply as

$$\mathbf{v}_i^l = \alpha \mathbf{v}_i^{l,l+1} = \alpha [u_i^{l,l+1}, v_i^{l,l+1}]^t. \quad (13)$$

This motion model assumes constant velocity of objects between frames  $l$  and  $l+1$ , or a “linear motion model.” This assumption is obviously not strictly true, because real-world object velocities between two frames will almost never be exactly constant. However, due to the relatively small temporal sampling interval for most video sequences, the linear velocity model is adequate.<sup>1</sup> Note that

<sup>1</sup>Note that more accurate motion models, for example *constant acceleration* instead of constant velocity, require motion information for more than two frames, which can become unwieldy. Most temporal interpolation schemes use the same linear motion model used here.

the motion elements in (13) are in general non-integer in value.

The motion vectors are used to model the motion compensation relationship explicitly as

$$\mathbf{z}^l = \mathbf{A}^l \mathbf{z} + \mathbf{n}_m^l, \quad (14)$$

where  $\mathbf{n}_m^l$  is the noise, or error, that accounts for the uncertainty in estimating  $\mathbf{z}^l$  with the motion-compensated  $\mathbf{z}$ . Note that (14) relates *original* images in the sequence. Since the motion to the intermediate frame is in general non-integer, the motion-compensating matrix  $\mathbf{A}^l$  is formed based on a bi-linear interpolation model; see [11] for details on construction of such a matrix.

Note that the motion relationship in (14) relates each pixel in frame  $l$  to a pixel in the missing frame. Due to the motion vector determination of (13), it is quite likely that there will be pixels in the missing frame that have no corresponding observation, and hence no information from surrounding frames is available with which to estimate these pixels. For such regions of pixels the HMRF prior image model “fills the holes”—the prior smoothness assumptions cause the algorithm to smooth over the regions with no observations, masking their appearance. In such regions, the HMRF performs quite similar to the error concealment algorithm of Salama et al. [12].

Due to the DCT quantization noise model, it is known that  $\mathbf{z}_q^l = \mathbf{z}^l + \mathbf{e}_z^l$ , where  $\mathbf{e}_z^l$  is the DCT quantization noise for frame  $l$  with autocovariance  $\mathbf{K}_{\mathbf{e}_z^l} = \mathcal{D}^t \mathbf{K}_{\mathbf{e}_y^l} \mathcal{D}$ . Combining the quantization noise model with (14),

$$\mathbf{z}_q^l = \mathbf{z}^l + \mathbf{e}_z^l, \quad (15)$$

$$= \mathbf{A}^l \mathbf{z} + \mathbf{n}_m^l + \mathbf{e}_z^l, \quad (16)$$

$$= \mathbf{A}^l \mathbf{z} + \mathbf{n}^l, \quad (17)$$

where  $\mathbf{n}^l$  is the error, or noise, between  $\mathbf{z}_q^l$  and its motion compensation from  $\mathbf{z}$ , and consists of both motion-compensation noise and DCT quantization noise. Characteristics of the DCT quantization noise have already been covered in some detail, and the motion noise must now be characterized to continue. It is assumed that the motion noise  $\mathbf{n}_m^l$  is Gaussian with autocovariance matrix  $\mathbf{K}_m^l$ . The Gaussian assumption has been used before, for example by Schultz and Stevenson in [13]. However, in [13] the authors assume IID Gaussian noise, which for the case at hand is rather limited—IID noise implies that each observation is equally reliable, and does not take into consideration spatially-varying errors in motion estimation such as appearing/disappearing objects, lighting changes, or incorrect motion vectors.

Here it is proposed that a spatially-varying motion noise model be used. Suppose first that the autocovariance of the motion-compensation noise  $\mathbf{K}_m^l$  is diagonalized by the transformation matrix  $\mathcal{U}$ , such that

$$\mathbf{K}_m^l = \mathcal{U}^t \mathbf{K}_d^l \mathcal{U}, \quad (18)$$

where  $\mathbf{K}_d^l$  is a diagonal matrix. Thus, with knowledge of  $\mathcal{U}$  the autocovariance of the motion-compensating noise is represented by the  $N$  elements of  $\mathbf{K}_d^l$  rather than the  $N^2$

elements of  $\mathbf{K}_m^l$ , where  $N$  is the number of pixels in the image.

Probability theory says that the transformation matrix  $\mathcal{U}$  is the Karhunen-Loève Transform (KLT), but to determine  $\mathcal{U}$  using the KLT requires prior knowledge of the matrix  $\mathbf{K}_m^l$ , and would result in a prohibitively complex FRAMEstimation algorithm. Rather than rely on the KLT, a simplifying assumption is used to allow construction of  $\mathbf{K}_m^l$ . Two-dimensional signals that are well-modeled by first-order prediction models with high one-step correlation parameters have autocovariance matrices that are approximately diagonalized by the DCT [7]. Assuming such a signal leads to the approximation

$$\mathbf{K}_m^l = \mathcal{D}^t \mathbf{K}_d^l \mathcal{D}, \quad (19)$$

where  $\mathbf{K}_d^l$  is diagonal, and  $\mathcal{D}$  is the BDCT, which is the DCT applied on a block-by-block basis. Such an approximation relies on the BDCT approximating the KLT for motion compensation error, and requires some further justification. The BDCT *does* do a reasonable job of decorrelating signals with high correlation parameters, as evidenced by its popular use in image and video compression standards [3–5]. The following points provide justification for assuming that the motion-compensation noise is highly correlated, and hence has an autocovariance matrix that is approximately diagonalized by the BDCT:

- **Appearing/Disappearing Objects.** If an object is present in one frame but is covered up in the next frame, then whatever motion vectors are found for the object will be pointing to an incorrect object. The error in such a case is the difference between the two distinct objects, and will be highly correlated provided that the two objects are independent (a reasonable assumption) and that each is relatively smooth (a common assumption for images, at least at the local level).
- **Off-screen Motion Vectors.** If the motion in a scene results in a motion vector pointing off the screen (i.e., pointing outside of the reference image), then the situation is similar to the case of appearing/disappearing objects above.
- **Lighting Changes.** If the light incident upon an object changes (due to, for example, some change in ambient light or a change in an object’s orientation with respect to a light source), even with perfect motion vectors there will be considerable motion-compensation error. Since the lighting change will probably apply to relatively large regions, it is reasonable to assume that the error will be highly correlated.
- **Motion Vector Errors.** Errors in motion vectors arise from a variety of sources. One major source of error is due to limitations in the motion model used in motion estimation—perhaps the true motion exceeded the range of motion searched in the estimation procedure; or maybe the motion model assumes a purely

translational motion, and the actual motion consisted of rotation or deformation; or maybe the motion-estimation algorithm simply found the wrong vector for some unknown reason. In any case, the resulting motion vector is pointing to a *wrong* object, and the discussion for appearing/disappearing objects applies here as well.

- **Experimental Tests.** The motion-estimation algorithm used in this work was applied to the *foreman* sequence at 15 fps to examine the motion-compensation error. Values of the one-step correlation coefficients in the horizontal and vertical directions ( $\rho_1$  and  $\rho_2$ ) were estimated on a frame-by-frame basis, and averaged out to approximately 0.4 in both the horizontal and vertical dimensions. While 0.4 is certainly not approximately 1.0, the BDCT still performs a certain amount of de-correlation—just not as much as if the correlation coefficients were approximately 1.0. Indeed, the BDCT is used on a wide class of images, many of which have correlation coefficients considerably less than 1.0.

While the above points provide justification for using the BDCT to de-correlate the motion-compensation noise, there is at least one strong argument against:

- When none of the situations presented above apply—i.e., there are no appearing/disappearing objects, no off-screen motion vectors, no lighting changes, and the motion vectors are determined perfectly—then the motion-compensation error does *not* seem to be highly correlated. In the experimental tests mentioned above, during the few areas of the *foreman* sequence that had almost no motion at all, the estimated correlation coefficients varied between 0.0 (i.e., not correlated at all) and 0.2.

While perhaps not perfectly de-correlating the motion-compensation noise, it is argued that for most situations using the BDCT provides a more accurate description of the noise than an IID assumption. Furthermore, when performing FRAMEstimation for sequences at relatively low bit rates, motion-compensation errors that are not approximately de-correlated by the BDCT—where the motion error is quite small—will be dominated by the BDCT quantization noise, and the slight mid-modeling has little negative impact.

The variances  $\sigma_i^2[i]$ ,  $i = 0, \dots, N - 1$ , that compose the diagonal matrix  $\mathbf{K}_d^l$  must be determined. Since the actual frames  $\mathbf{z}$  and  $\mathbf{z}^l$  are unavailable, the variances must be estimated from the available frames at times  $l$  and  $l + 1$  instead. As mentioned previously in this section, motion estimation is performed with the processed images  $\hat{\mathbf{z}}^l$  and  $\hat{\mathbf{z}}^{l+1}$  rather than the received frames  $\mathbf{z}_q^l$  and  $\mathbf{z}_q^{l+1}$  to prevent compression artifacts from interfering with the process. To determine motion compensation variances here, the processed frames are used as well. Assuming that the motion-compensation variances are proportional to the distance between frames, as done in [13], the re-

quired terms are found here as

$$\sigma_i^2[i] = \alpha \left( [\mathcal{D}(\hat{\mathbf{z}}^l - \mathbf{A}^{l,l+1}\hat{\mathbf{z}}^{l+1})]_i \right)^2, \quad (20)$$

where  $\mathbf{A}^{l,l+1}$  is the motion-compensation matrix from frame  $l$  to frame  $l + 1$ . As can be seen, the variance estimate of (20) is computed based on only a single sample. The variance of this estimate is thus quite high, but is unavoidable because there is only the single observation of motion-compensation error available.

From (17),  $\mathbf{z}_q^l | \mathbf{z}$  is Gaussian with mean  $\mathbf{A}^l \mathbf{z}$  and autocovariance  $\mathbf{K}^l = \mathbf{K}_{e_z} + \mathbf{K}_m^l$ . Similarly,  $\mathbf{z}_q^{l+1} | \mathbf{z}$  can be shown to be Gaussian with mean  $\mathbf{A}^{l+1} \mathbf{z}$  and autocovariance  $\mathbf{K}^{l+1} = \mathbf{K}_{e_z} + \mathbf{K}_m^{l+1}$ . Assuming conditional independence,

$$p(\mathbf{z}_q^k, k \in \mathcal{K} | \mathbf{z}) = \prod_{k \in \mathcal{K}} p(\mathbf{z}_q^k | \mathbf{z}), \quad (21)$$

where the individual probabilities on the right-hand side are now known from (17).

## 5 SOLUTION AND EXPERIMENTAL RESULTS

With both terms from (2) now determined, the final optimization problem to be solved becomes

$$\hat{\mathbf{z}} = \arg \min_{\mathbf{z}} [\lambda_F u(\mathbf{z}) + v(\mathbf{z})], \quad (22)$$

where

$$u(\mathbf{z}) = \sum_{c \in \mathcal{C}} \rho_T(\mathbf{d}_c^t \mathbf{z}), \quad (23)$$

$$v(\mathbf{z}) = \frac{1}{2} \sum_{k \in \mathcal{K}} (\mathbf{A}^k \mathbf{z} - \mathbf{z}_q^k)^t \mathbf{K}^{k-1} (\mathbf{A}^k \mathbf{z} - \mathbf{z}_q^k). \quad (24)$$

The convex optimization problem in (22) is solved using a gradient descent algorithm [6]. After substituting for  $\mathbf{K}^k$ , the gradients of the individual terms are given as

$$\nabla u(\mathbf{z}) = \sum_{c \in \mathcal{C}} \mathbf{d}_c \rho_T'(\mathbf{d}_c^t \mathbf{z}), \quad (25)$$

$$\nabla v(\mathbf{z}) = \sum_{k \in \mathcal{K}} \mathbf{A}^{k,t} \mathcal{D}^t \left[ \mathbf{K}_{e_y} + \mathbf{K}_d^k \right]^{-1} \mathcal{D}(\mathbf{A}^k \mathbf{z} - \mathbf{z}_q^k). \quad (26)$$

Denoting the estimate of  $\mathbf{z}$  at iteration  $w$  as  $\mathbf{z}^{(w)}$ , and  $\mathbf{g}(\mathbf{z}) = \lambda_F \nabla u(\mathbf{z}) + \nabla v(\mathbf{z})$ , the gradient descent algorithm forms the new estimate as

$$\mathbf{z}^{(w+1)} = \mathbf{z}^{(w)} - \tau^{(w)} \mathbf{g}(\mathbf{z}^{(w)}), \quad (27)$$

where  $\tau^{(w)}$  is a step size that ideally reduces the objective function by as much as possible. The step size is determined by a simple one-dimensional search algorithm that finds the best  $\tau^{(w)}$  in a pre-defined search range, where the search is performed at a pre-determined resolution. Iterations of (27) are continued until improvements in the objective function fall below a small threshold.

The FRAMEstimation algorithm has been implemented in an H.263 decoder, and PSNR plots of the inserted frames are shown in Figs. 2 and 3 for various values

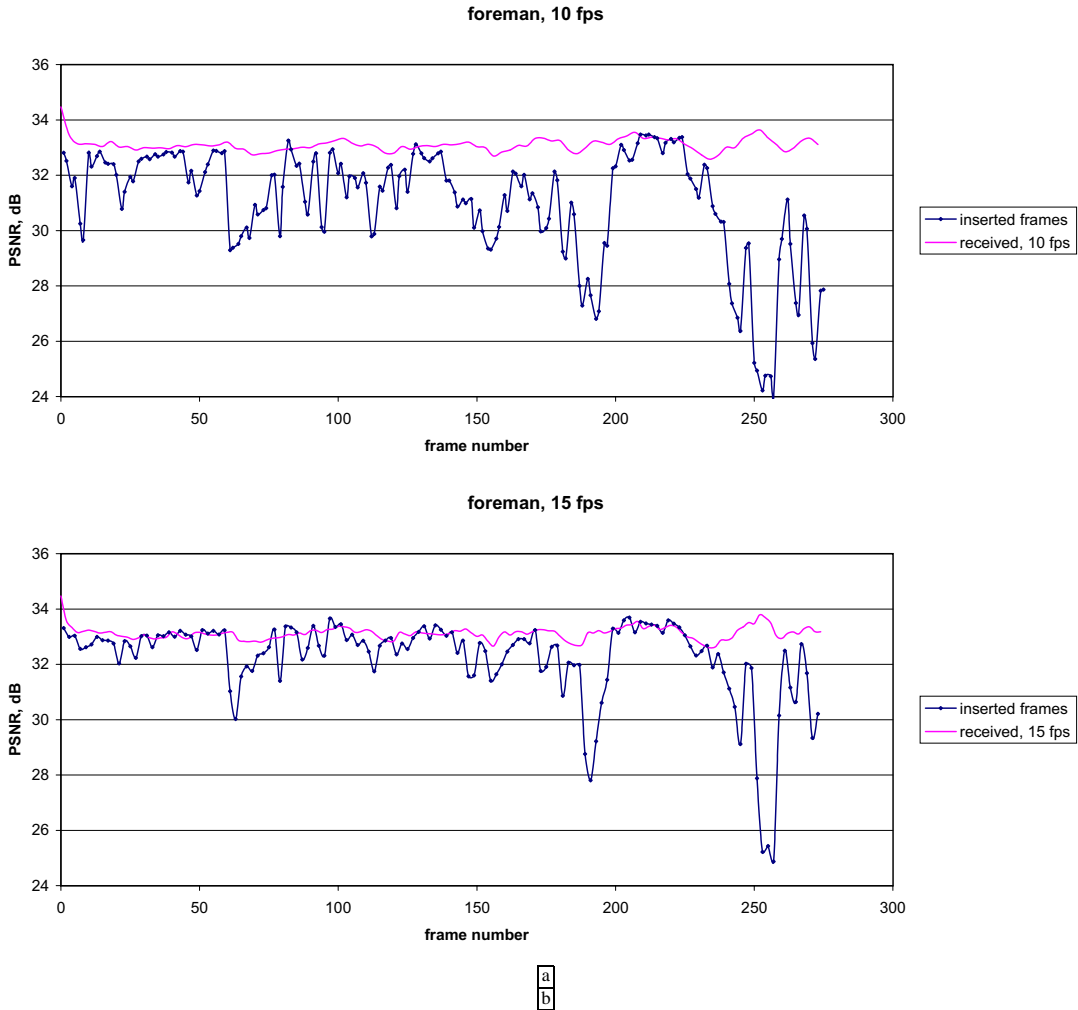


Figure 2: PSNR plots for inserted frames in the *foreman* sequence. (a) 10 fps, QP = 8,  $\lambda_F = 0.00075$ , inserting two frames between each pair of received frames to achieve 30 fps; (b) 15 fps, QP = 8,  $\lambda_F = 0.00075$ , inserting one frame between each pair of received frames to achieve 30 fps. The Huber parameter is  $T = 10.0$ .

of quantization parameter (QP). For each of these cases, the original sequence was compressed using inter-coded frames for all but the first frame. Frame rates of the received sequences varied between 7.5, 10.0, and 15.0 fps, implying that 3, 2, and 1 frame or frames needed to be inserted between each pair of received frames in order to achieve 30 fps. Plots of the received frames' PSNR are shown in these figures as well, and are provided as a baseline with which to compare the PSNR of the inserted frames. Note that these figures only present PSNR for the inserted frames, and do not include PSNR results for any enhancements done to the received frames. Close-ups of various frames from these sequences are shown in Figs. 4, 5, and 6, and demonstrate that the proposed FRAMEstimation scheme effectively inserts missing frames, while preventing the propagation of compression artifacts from the received frames.

At first glance, the FRAMEstimation PSNR plots of Figs. 2 and 3 might seem to suggest that the proposed algorithm performs poorly. However, although most of the inserted frames have noticeably lower PSNR than the surrounding received frames, the visual results presented

in Figs. 4 through 6 suggest quite the opposite—in each of the three cases, noticeable compression artifacts are present in the received frames, while the inserted frames have had these compression artifacts effectively removed. The low values from the PSNR charts can be explained as follows: If the estimated motion information to the inserted frame is incorrect by even a single pixel, then objects placed in the inserted frame will not be located in the exact proper location, which will result in poor PSNR due to misalignment of edges between the true image and the inserted image. Small errors in motion vectors can arise from a variety of sources, the most prevalent of which is perhaps the linear motion model (recall that velocities are assumed to be constant between frames). However, when the sequences are viewed at 30 fps, these small misalignments of edges have no subjective effect, and the inserted frames typically appear to have better quality than the surrounding received frames. This provides evidence of the inadequacy of PSNR as a metric for video quality.

However, there are areas of significant motion where the proposed FRAMEstimation algorithms fails, in particular the two steep drops in the PSNR curves for the

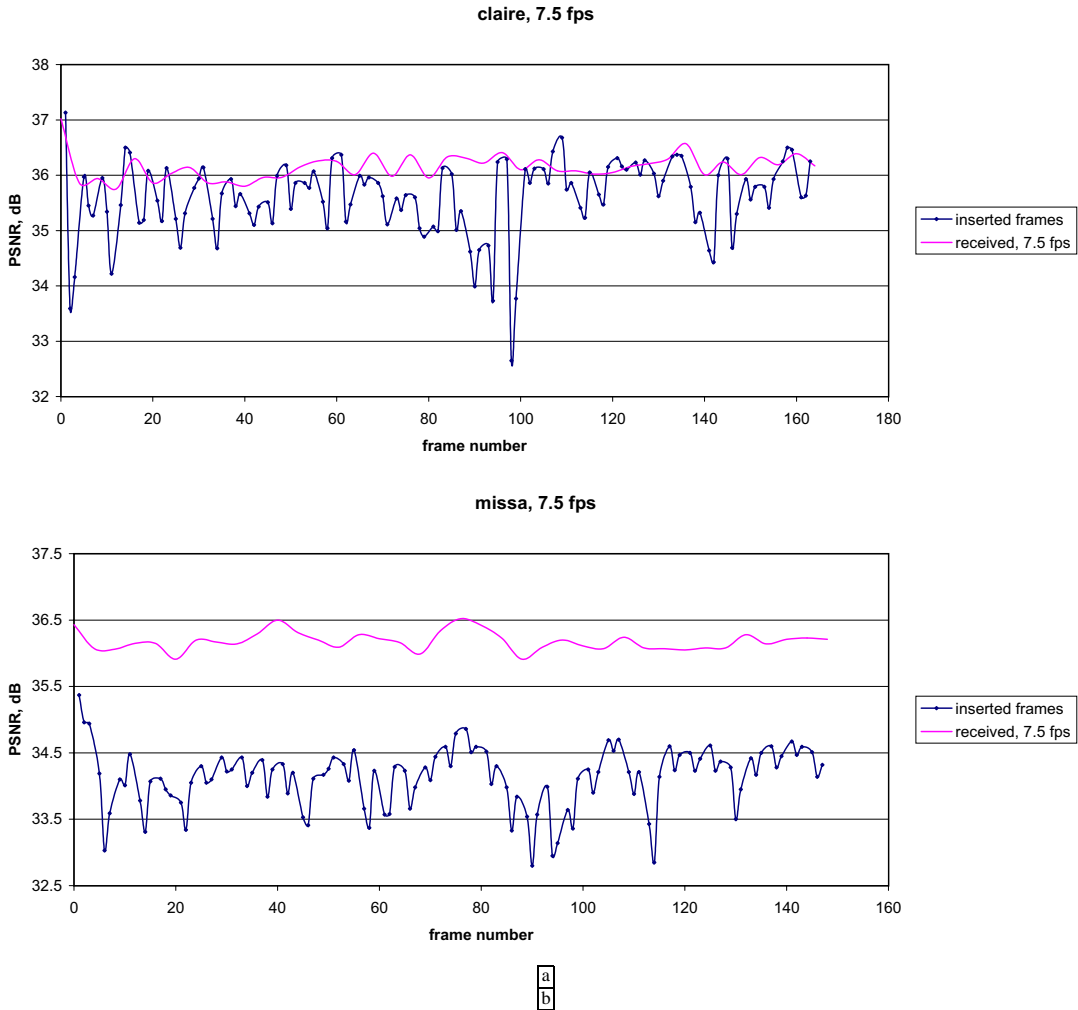


Figure 3: PSNR plots for inserted frames. (a) *claire*, 7.5 fps, QP = 14,  $\lambda_F = 0.00035$ ; (b) *missa*, 7.5 fps, QP = 14,  $\lambda_F = 0.00050$ . In both cases, three frames have been inserted between each pair of received frames to achieve 30 fps. The Huber parameter is  $T = 10.0$ .

*foreman* sequence in Fig. 2. In these two sections of the sequence, there is severe motion that is not correctly estimated by the motion estimation algorithm, and the reconstructions for the inserted frames have noticeable artifacts. This problem is not due to limitations in the FRAMEstimation formulation, but is rather due to limitations in the motion estimation algorithm—given true motion information, the algorithm would have determined a good estimate. In an implementation where quality is critical, some sort of heuristic test would need to be conducted to determine when the motion estimation has failed, and the inserted frames in these cases could just use frame repetition.

Results were only provided for sequences at relatively low bit rates, but the algorithm performs equally well for higher-quality video as well. Although not discussed previously, FRAMEstimation algorithms can also be useful for creating slow-motion sequences—for example, a 30-fps sequence could be converted to 90 fps, and then displayed at one-third speed. In general, results for slow-motion FRAMEstimation tend to be quite good because the smaller temporal sampling period results in less se-

vere motion between frames, which in turn allows for better motion estimation, enabling better FRAMEstimation results. Furthermore, the constant velocity motion model is more accurate for smaller temporal sampling periods, allowing even better quality for inserted frames.

## 6 CONCLUSION

This paper has presented a method for increasing the frame rate of video compressed using the block discrete cosine transform. A Bayesian formulation for the problem was developed, and the estimate of an inserted frame was given as the solution to a convex optimization problem. The *a priori* smoothness term in the Bayesian formulation prevents spatial compression artifacts, which may be present in the received video frames, from propagating to the inserted frames. Experimental results were presented that demonstrated the utility of the algorithm.

## REFERENCES

- [1] C. Cafforio, F. Rocca, and S. Tubaro. Motion compensated image interpolation. *IEEE Trans. on Communications*, 38(2):215–222, Feb. 1990.

- [2] R. Castagno, P. Haavisto, and G. Ramponi. A method for motion adaptive frame rate up-conversion. *IEEE Trans. on CSVT*, 6(5):436–446, Oct. 1996.
- [3] ISO/IEC 10918-1. *JPEG—Information Technology—Digital Compression and Coding of Continuous-Tone Still Images—Requirements and Guidelines*, 1993.
- [4] ISO/IEC 13818-2. *MPEG-2—Information Technology—Generic Coding of Moving Pictures and Associated Audio*, Part 2: Video, 1996.
- [5] ITU-T Recommendation H.263. *Video coding for low bit rate communication*, 1998.
- [6] S. L. S. Jacoby, J. S. Kowalik, and J. T. Pizzo. *Iterative Methods for Nonlinear Optimization Problems*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1972.
- [7] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1989.
- [8] J. Mateos, C. Ilia, B. Jiménez, R. Molina, and A. K. Katsaggelos. Reduction of blocking artifacts in block transformed compressed color images. In *International Conference on Image Processing*, volume 1, pages 401–405, Oct. 4–7 1998.
- [9] T. Meier, K. N. Ngan, and G. Crebbin. Reduction of blocking artifacts in image and video coding. *IEEE Trans. on CSVT*, 9(3):490–500, Apr. 1999.
- [10] T. P. O’Rourke and R. L. Stevenson. Improved image decompression for reduced transform coding artifacts. *IEEE Trans. on CSVT*, 5(6):490–499, Dec. 1995.
- [11] M. A. Robertson. *High-Quality Reconstruction of Digital Images and Video from Imperfect Observations*. Ph.D. dissertation, University of Notre Dame, Apr. 2001.
- [12] P. Salama, N. Shroff, and E. J. Delp. A Bayesian approach to error concealment in encoded video streams. In *International Conference on Image Processing*, volume 2, pages 49–52, 1996.
- [13] R. R. Schultz and R. L. Stevenson. Extraction of high-resolution frames from video sequences. *IEEE Trans. on Image Processing*, 5(6):996–1011, June 1996.
- [14] R. Thoma and M. Bierling. Motion compensating interpolation considering covered and uncovered background. *Signal Processing: Image Communication*, 1(2):191–212, Oct. 1989.
- [15] S. Tubaro and F. Rocca. Motion field estimators and their application to image interpolation. In M. I. Sezan and R. L. Lagendijk, editors, *Motion Analysis and Image Sequence Processing*, chapter 6, pages 153–187. Kluwer Academic Publishers, Boston, 1993.
- [16] C.-K. Wong and O. C. Au. Fast motion compensated temporal interpolation for video. In *Proc. of SPIE Visual Comm. and Image Processing*, pages 1108–1118, Taipei, Taiwan, May 24–26 1995.
- [17] Y. Yang, N. P. Galatsanos, and A. K. Katsaggelos. Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images. *IEEE Trans. on CSVT*, 3(6):421–432, Dec. 1993.
- [18] Y. Yang, N. P. Galatsanos, and A. K. Katsaggelos. Projection-based spatially adaptive reconstruction of block-transform compressed images. *IEEE Trans. on Image Processing*, 4(7):896–908, July 1995.



**Mark A. Robertson** received the B.S. in Electrical Engineering from Tri-State University, Angola, IN, in 1996, and the M.S. and Ph.D. degrees in Electrical Engineering from the University of Notre Dame, Notre Dame, IN, in 1998 and 2001. While at Notre Dame, he was supported by the Arthur J. Schmitt Fellowship, the Intel Corporation, and the Indiana Space Grant Consortium. His current research interests include image and video enhancement, high-quality electronic imaging, and image and video communication.



**Robert L. Stevenson** received the B.E.E. degree (summa cum laude) from the University of Delaware in 1986, and the Ph.D. in Electrical Engineering from Purdue University in 1990. While at Purdue he was supported by graduate fellowships from the National Science Foundation, DuPont Corporation, Phi Kappa Phi, and Purdue University. He joined the faculty of the Department of Electrical Engineering at the University of Notre Dame in 1990, where he is currently an Associate Professor. His research interests include image/video processing, image/video compression, robust image/video communication systems, multimedia systems, ill-posed problems in computational vision, and computational issues in image processing.

Dr. Stevenson is an Associate Editor of the *IEEE Trans. on Image Processing* and the *IEEE Trans. on Circuits and Systems for Video Technology*, and is a former Associate Editor of the *Journal of Electronic Imaging*.



a b  
c d

Figure 4: Example close-up inserted frames for 10-fps *foreman* compressed with H.263 quantization parameter  $QP = 8$ . (a) Received frame 114; (b) received frame 117; (c) and (d) inserted frames 115 and 116, reconstructed with  $\lambda_F = 0.00075$ ,  $T = 10$ .



a b  
c  
d e

Figure 5: Example close-up inserted frames for 7.5-fps *claire* sequence of Fig. 3. (a) Received frame 88; (b) received frame 92; (c), (d), and (e) inserted frames 89, 90, and 91.



a b  
c  
d e

Figure 6: Example close-up inserted frames for 7.5-fps *missa* sequence of Fig. 3. (a) Received frame 112; (b) received frame 116; (c), (d), and (e) inserted frames 113, 114, and 115.