A Multiresolution Approach to Motion Estimation and Interpolation with Application to Coding of Digital HDTV

Kamil M.Uz and Martin Vetterli Department of Electrical Engineering
and Center for Telecommunications Research
Columbia University, New York, NY 10027

Didier LeGall
Bell Communications Research
Morristown, NJ 07960

Abstract

The notion of multiresolution signal analysis is used to develop motion estimation algorithms on a three dimensional pyramid. The motion vectors are then used to interpolate a sequence at a higher resolution, and the difference is coded as an error signal to guarantee very high quality coding. This process is repeated to generate a hierarchy of sequences. Such a scheme is thus well suited for compatible coding of HDTV. We show performance of the motion estimation algorithm and of the coding at 100 Mbit/s.

1. Introduction

Hierarchical methods in motion estimation have been quite successful [1]. Multiresolution or pyramidal approaches have been used both for vision and image coding [2]. In this paper, we show how a multiresolution view of video representation and coding leads to these techniques, and gives a natural framework for them.

First, we outline the spatio-temporal pyramidal coding based on motion interpolation. Note that the redundancy in the number of samples as compared to a subband scheme is negligible in a three dimensional pyramidal scheme, and the robustness obtained justifies the choice of pyramidal coding. Furthermore, compatible sub-channels can be designed to be cleaner than in a subband coding scheme, an issue of importance for HDTV where possibly a sub-channel has to be compatible with existing standards.

Then, the three dimensional hierarchical motion estimation, which is an extension of the method in [1], is described. The temporal subsampling is justified, and possible problems and their solutions are explained. From a complexity point of view, this is of course a method of choice, and this will be detailed.

Experimental results on a progressive test sequence demonstrate both the accuracy of the motion estimation as well as the quality of the interpolated sequences, and some preliminary coding results are given to indicate the potential of this method for high quality coding of HDTV.

2. Overview of the Spatio-Temporal Pyramid

In this section, we introduce our multiresolution scheme that employs a three dimensional pyramid structure [3] (see Figure 1). The structure consists in representations of an image sequence at multiple scales, both in time and in space. It is encoded hierarchically in a top–down manner. The decoder uses a motion based interpolator [4] in the temporal direction, and builds the same pyramid structure top–down, starting from the coarsest level, and using the motion information and the interpolation error supplied at each level by the encoder.

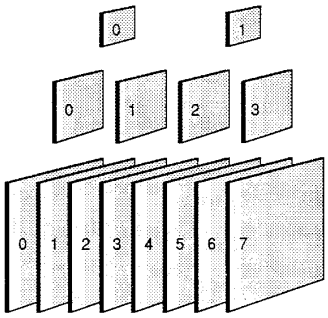

Figure 1: The spatio-temporal pyramid structure

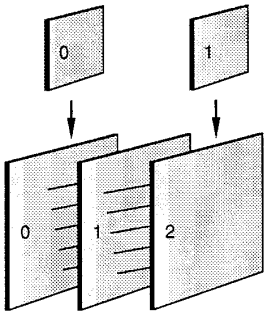

Figure 2: First spatial resolution is increased, and then missing frames are temporally interpolated yielding the next level in the pyramid.

The encoder, starting from the original sequence $I_{0,0}$, forms a set of K successively lower resolution representations, $I_{1,1}$ to $I_{K,K}$. The subscripts denote the subsampling factors in spatial and temporal dimensions. So note that each subsequence $I_{j,j}$ has both temporal and spatial resolutions reduced by 2^j. To obtain the coarser subsequence $I_{j,j}$ from $I_{j-1,j-1}$, first the sequence is decimated along time, obtaining $Ij-1,j$. Next, this intermediate sequence is spatially lowpass filtered to prevent aliasing, and spatially decimated giving $I_{j,j}$.

The decoder starts from the coarsest sequence $I_{K,K}$, and at each step obtains a finer version. Suppose the pyramid has been reconstructed down to level j. Then the decoder interpolates $I_{j,j}$ to a finer spatial grid, forming $\tilde{I}_{j-1,j}$. The encoder sends the difference $E_{j-1} = I_{j-1,j} - \tilde{I}_{j-1,j}$, reconstructing $I_{j-1,j}$. Motion based temporal interpolation follows, constructing a signal $\tilde{I}_{j-1,j-1}$ at the new grid for level $j-1$. The encoder now sends the difference $E'_{j-1} = I_{j-1,j-1} - \tilde{I}_{j-1,j-1}$ and the decoder adds this signal to complete the reconstruction of the pyramid down

CH2868-8/90/0000-1298$1.00 © 1990 IEEE

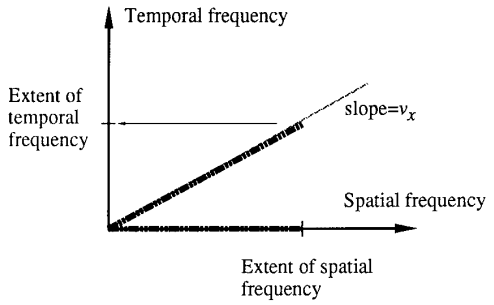

Figure 3: A signal has been uniformly displaced by v_x along time. Note that the Fourier transform is tilted in (ω_x, ω_t) plane, and the temporal frequency content depends both on the velocity and on the original spatial frequencies present.

to level j (see Figure 2). Note that order of reconstruction is important: we expect to have less error in temporal interpolation, at least in case of uniform motion.

Before going into further detail, we would like to address one particular question that may be raised: the lack of filtering prior to temporal decimation. There are two arguments to justify this choice. First, we note that aliasing is not a major concern in the pyramid, even spatially, and the prefiltering has more to do with efficiency than with anti-aliasing. The goal is finding a "suitable" representation of the image at the coarser grid that (i) is suitable for coding; (ii) gives a good prediction when interpolated back to the fine grid; (iii) is pleasant to look at (in case used in conjunction with progressive transmission). We should emphasize that the spatial prefiltering meets all these criterion, while temporal lowpass filtering is not likely to help in any.

The second observation is that spatial processing has already reduced the temporal frequency content. We could define temporal aliasing and make a rigorous statement, but the following argument should be satisfactory: Let $I(\omega_x, \omega_y)$ be the Fourier transform of a still image. Suppose a sequence is formed by displacing I with a uniform velocity $\mathbf{v} = (v_x, v_y)$. Then the Fourier transform of the sequence is

$$I'(\omega_x, \omega_y, \omega_t) = I(\omega_x, \omega_y)\delta(\omega_t + v_x\omega_x + v_y\omega_y) \quad (1)$$

which tells that $I'(\omega_x, \omega_y, \omega_t)$ assumes the value of $I(\omega_x, \omega_y)$ on the plane

$$\omega_t = -v_x\omega_x - v_y\omega_y \quad (2)$$

To further illustrate the point, consider Figure 3, where a one dimensional signal is uniformly displaced in time. This displacement is equivalent to a rotation in the $I(\omega_x, \omega_t)$ Fourier plane. Notice the equivalence of suitably adjusted spatial and temporal filters. Indeed, one can limit the temporal frequencies solely by spatial filtering, and vice versa.

It should now be clear that motion mixes spatial and temporal frequencies. Admittedly, the previous example was oversimplified, with the whole picture being uniformly displaced. However, the interplay of time and space in a real scene seems to be very delicate, and the design of motion adaptive spatio-temporal filters is an interesting problem.

In the next sections, we first describe the motion estimation algorithm which plays a key role in the temporal interpolation. Then we outline how to construct a nested hierarchy, which may

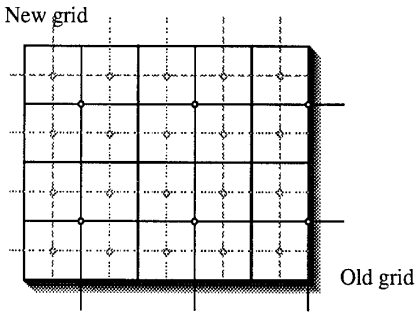

Figure 4: The blocks and corresponding grids at two successive scales in the spatial hierarchy

be of interest for digital coding of HDTV. The scheme allows progressive transmission, prioritized transmission for packet video, and may be used to support multiple standards.

3. Hierarchical Motion Estimation

We start with the video signal $I(\mathbf{r}, n)$ where $\mathbf{r}$ denotes the spatial coordinate (x, y), and n denotes the time. The goal is to find a mapping $\mathbf{d}(\mathbf{r}, n)$ that would help reconstruct $I(\mathbf{r}, n)$ from $I(\mathbf{r}, n-1)$ and $I(\mathbf{r}, n+1)$. We assume a restrictive motion model, where the image is assumed to be composed of rigid objects in translational motion on a plane.

$$I(\mathbf{r}, n) = I(\mathbf{r} - \mathbf{d}(\mathbf{r}, n), n-1). \quad (3)$$

We also expect homogeneity in time, i.e.

$$I(\mathbf{r}, n) = I(\mathbf{r} + \mathbf{d}(\mathbf{r}, n), n+1). \quad (4)$$

Furthermore, we are using a block based scheme, expecting these assumptions are approximately valid for all points within a block b using the same displacement vector $\mathbf{d}_b$. These assumptions are easily justified when the blocks are much smaller than the objects, and temporal sampling is sufficiently dense.

In what follows, we change the notation slightly, omitting the spatial coordinate $\mathbf{r}$ when the meaning is clear, and replacing $I(\mathbf{r}, n)$ by $I_0(n)$, and $\mathbf{d}(\mathbf{r}, n)$ by $\mathbf{d}_0(n)$.

To compute $\mathbf{d}_0(n)$, we require a hierarchy of lower spatial resolution representations of $I_0(n)$ denoted $I_k(n)$, $0 < k \leq K$. $I_k(n)$ is computed from $I_{k-1}(n)$ by first spatially low-pass filtering with half-band filters, reducing the spatial resolution. This filtered image is then decimated, giving $I_k(n)$. Note that this reduces by 4 the number of pels in a frame at each level.

3.1 Basic search procedure

The search starts at the top level of the spatial hierarchy, $I_K(n)$. The image $I_k(n)$ is partitioned into non-overlapping blocks of size MxM. For every block b in the image $I_k(n)$, a displacement $\mathbf{d}_b$ is searched to minimize the matching criterion

$$\sum_{\mathbf{r} \in b} |I_K(\mathbf{r}, n) - \tilde{I}_K(\mathbf{r}, n)| \quad (5)$$

where $\tilde{I}_K(\mathbf{r}, n)$ is the motion based estimate of $I_k(n)$ computed as

$$\tilde{I}_k(\mathbf{r}, n) = 1/2(I_k(\mathbf{r} - \mathbf{d}_b, n-1) + I_k(\mathbf{r} + \mathbf{d}_b, n+1)) \quad (6)$$

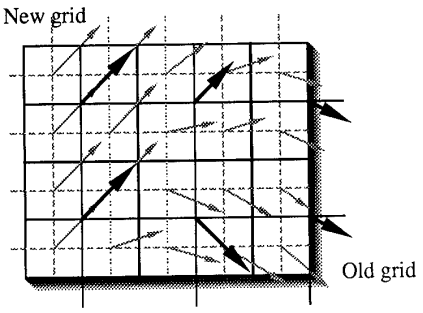

Figure 5: The motion field is resampled in the new finer grid, giving an initial estimate for the search.

Notice that this estimate implies that if a block has moved the distance $\mathbf{d}$ between the previous and the current frame, it is expected to move $\mathbf{d}$ between the current and the following frame. This constitutes the symmetric block based search.

Suppose the displacement in the original sequence $I_0(n)$ were limited to $\pm d_{max}$ pels in each dimension. Then the displacement in the kth level is limited to $\pm d_{max}/2^k$, since the frames at this level have been k times spatially decimated by 2. In general, K can be chosen such that the displacement at the top level is known to be less than $\pm D$. Given this choice, we can limit the search for each $\mathbf{d}_K(r,n)$ to $\{(x,y) : -D \leq x,y \leq +D\}$. This results in $(2D+1)^2$ tests to compute $\mathbf{d}_b$ for each block. For a typical sequence, D can be 2 or 3, and K can be 3, allowing a maximum displacement of (at least) 25 or 49, respectively. As will be explained in the next paragraphs, even larger displacements can actually be handled.

3.2 Stepwise refinement

Going one step down the hierarchy, we want to compute $\mathbf{d}_k(n)$, given that $\mathbf{d}_{k-1}(n)$ is already known. We may think of $\mathbf{d}_k$ as a sampled version of an underlying continuous displacement field. Then $\{\mathbf{d}_k : 0 \leq k \leq K\}$ is a set of multi-resolution samples taken from the underlying displacement field. Therefore, $\mathbf{d}_k$ can be written as

$$\mathbf{d}_k(\mathbf{r},n) = 2\tilde{\mathbf{d}}_{k+1}(\mathbf{r},n) + \Delta\mathbf{d}_k(\mathbf{r},n) \tag{7}$$

where $\tilde{\mathbf{d}}_{k+1}$ is the displacement field $\mathbf{d}_{k+1}$ interpolated at the new sampling grid (see Fig), and $\Delta\mathbf{d}_k$ is the correction term for the displacement due to increased resolution at level k. The rescaling by 2 arises due to the finer sampling grid for I_k: the distance between two pels in the upper level has now doubled. Thus, we have reduced the problem of computing $\mathbf{d}_k$ into two subtasks which we now describe.

Computing $\tilde{\mathbf{d}}_{k+1}$ on the new sampling grid requires a resampling, or interpolation [1]. The discontinuities in the displacement field are often due to the occluding boundaries of moving objects. Therefore, the interpolation may be use a (tentative) segmentation of the frame based on the displacement field, or even on the successive frame difference. The segmentation would allow classifying the blocks as belonging to distinct objects, and preventing the "corona effect" around the moving objects. However, we use a simple bilinear interpolation for efficiency, moving the burden to computing $\Delta\mathbf{d}_k$.

After the interpolation, every block b has an initial displacement estimate $\hat{\mathbf{d}}_b$, which is equal to $2\tilde{\mathbf{d}}_{k+1}(\mathbf{r}_b,n)$, where $\mathbf{r}_b$ is the coordinate of the center of block b. Now the search for $\Delta\mathbf{d}_k$ is done inside a window centered around $\hat{\mathbf{d}}_b$, i.e

$$\mathbf{d}_b = \hat{\mathbf{d}}_b + \Delta\mathbf{d}, \qquad \Delta\mathbf{d} \in \{(x,y) : -D \leq x,y \leq +D\}. \tag{8}$$

Note that the number of displacements searched for a block is constant, while the number of searches increases exponentially down the hierarchy. The procedure is repeated until one obtains the motion field $\mathbf{d}_0(n)$, corresponding to $I_0(n)$. The maximum displacement that can be handled is

$$\sum_{i=0}^{K} 2^i d_{max} = (2^{K+1}-1)d_{max}. \tag{9}$$

This constrained window symmetric search algorithm yields a smooth motion field with high temporal and spatial correlation, at the same time allowing large displacements.

4. Motion Based Interpolation

Given frames $I(n-1)$ and $I(n+1)$, and the displacement $\mathbf{d}(n)$, we form $\tilde{I}(n)$, the motion based estimate of $I(n)$ by displaced averaging:

$$\tilde{I}(\mathbf{r},n) = 1/2(I(\mathbf{r}-\mathbf{d},n-1) + I(\mathbf{r}+\mathbf{d},n+1)). \tag{10}$$

Here, we use $\mathbf{d} = \mathbf{d}_b$, the displacement of the block containing $\mathbf{r}$. There are other alternatives, including a pel-level resampling of the displacement field $\mathbf{d}(\mathbf{r})$. However, this would significantly increase the complexity of the decoder, which is kept to a minimum by the blockwise averaging scheme. Furthermore, simulations indicate that the time-varying texture distortions caused by pel-level interpolation are visually unpleasant. It is important to preserve textures, but it is crucial to avoid time-varying distortions.

A special case occurs around the frame boundaries. A block close to the boundary may be interpolated outside the previous or the following frame. In that case, only the other frame is used for interpolation (and estimation). This is of particular relevance for images exhibiting a global motion (zoom and/or pan). For instance, the boundary is replaced from the following frame in the case of zoom with almost no unpleasant artifacts.

Even the best motion estimation will not, in general, yield perfect interpolation. Therefore, the interpolation error has to be made available to the receiver. For now, we note that we have more freedom in the error coding, unlike predictive schemes that are commonly used. We will elaborate on this problem in the next section, where we give some preliminary coding results.

4.1 Extension for the temporal pyramid

We shall use motion based interpolation as the temporal interpolator in a three dimensional pyramid scheme. However, the method, as presented, has some limitations especially when temporal sampling rate is reduced. Consider temporally decimating $I_1(n)$, call it $I_1^1(n)$ (recall that $I_1(n)$ has a reduced spatial resolution). If the original sequence $I_0(n)$ has a frame-to-frame maximum displacement of d_{max}, this property is preserved in $I_1^1(n)$. However, as we go up the spatial hierarchy, the displacements start to deviate from our original assumptions: We can no longer assume that the velocities are constant, nor that time

[1] This may seem to be a minor point, however the interpolation is crucial for best performance with the constrained search.

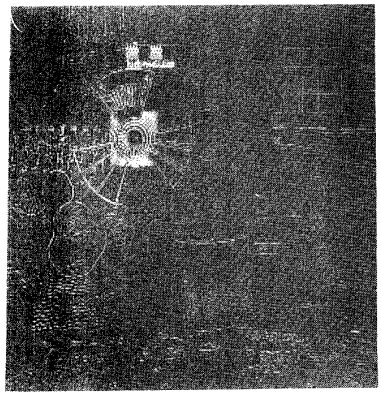

Figure 6: Typical difference image after spatial interpolation

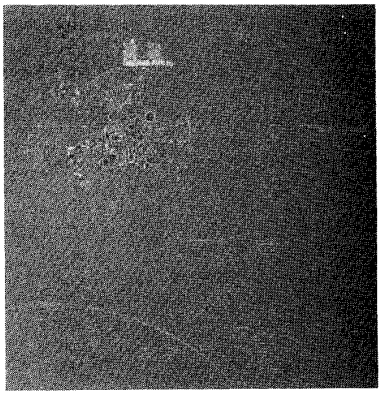

Figure 7: Typical difference image after temporal interpolation

is homogeneous, although we know that the maximum displacement is preserved. To make matters even worse, the area covered (or uncovered) by a moving object increases proportional to the temporal decimation factor. Thus, an increasingly larger ratio of the viewing area is covered/uncovered as we go up in our pyramid structure.

For a successful interpolation, we have to be able to (i) handle discontinuities in motion; (ii) interpolate uncovered areas. These are fundamental problems in any motion based scheme; a manifestation of the fact that we are dealing with solid objects in 3-D space, of which we have incomplete observations in the form of a 2-D projection on the focal plane.

To overcome these difficulties, we allow the interpolator to selectively use previous, following, or both frames on a block by block basis. The algorithm is thus modified in the final step, where $\Delta \mathbf{d}_0$ is computed to yield the final displacement $\mathbf{d}_0$. The estimator computes 2 additional estimates $\Delta \mathbf{d}_p$ and $\Delta \mathbf{d}_f$, using only the previous or the following frames, respectively. Now the displacement minimizing the interpolation error is chosen as the final displacement. In practice, one may weigh the errors, favoring the displacement obtained by the symmetric search.

5. Simulations

In this section, we give some preliminary simulation results, and note directions for further research. Our results indicate that excellent picture quality can be achieved at below 2 bits/pel. This corresponds to a compression factor of more than 10, and allows high quality digital transmission at 100 Mbits/s. for all digital HDTV.

We have used a spatio-temporal pyramid of three levels, with two sub-sequences roughly corresponding to NTSC and CIF quality. The source is a 512x512 RGB progressive test sequence developed at MIT. The sequence contains very high spatial detail and has a rotating test pattern with pan and zoom in the background.

A discrete cosine transform (DCT) based coder has been used for coding the top level and the difference images. This forms a hierarchical, embedded code of three layers with two compatible sub-channels. It may be desirable to have an interlaced subchannel, in which case an approach similar to [5] can be used.

For HDTV coding, the most critical level is the first one, containing the signal at full space and time resolution. We have found that difference signal after a spatial interpolation can be coded at 2 bits/pel, while the temporally interpolated frames require about 1 bit/pel (see Figure 2).

The next level, containing eight times less pixels, can be coded at a higher rate. The motion vectors and interpolation information contribute very low overhead, about 3-6 bits per 8x8 block. The motion vectors are differentially coded, while the interpolation information is run-length encoded.

The decoded sequences have a signal-to-noise ratio of about 35 dB, and have very good perceptual quality. In particular, errors due to temporal interpolation are almost invisible, and might be dispensed with as part of a rate control strategy.

The authors wish to thank ATRP at MIT for providing a progressive test sequence. First two authors acknowledge support by the National Science Foundation under grants CDR-84-21402 and MIP-88-08277.

References

[1] M. Bierling, "Displacement estimation by hierarchical block-matching," in *SPIE Conference on Visual Communications and Image Processing*, (Boston), pp. 942–951, November 1988.

[2] P. J. Burt, "Multiresolution techniques for image representation, analysis, and 'smart' transmission," in *SPIE Conference on Visual Communications and Image Processing*, (Philadelphia, Pennsylvania), pp. 2–15, November 1989.

[3] P.J.Burt and E.H.Adelson, "The laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. 31, pp. 532–540, April 1983.

[4] K. M. Uz and D. LeGall, "Motion compensated interpolative coding," in *Picture Coding Symposium*, (Cambridge, MA), March 1990.

[5] M. Vetterli, J. Kovačević, and D. LeGall, "Perfect reconstruction filter banks for HDTV representation and coding," in *Proc. of the Third Inter. Workshop on HDTV*, (Torino, Italy), September 1989.