

Advanced de-interlacing techniques

E.B. Bellers and G. de Haan

Philips Research Laboratories

Television Systems Group

Prof. Holstlaan 4

5656 AA Eindhoven

The Netherlands

tel: +31-40-2744285

fax: +31-40-2742630

Abstract: This paper presents an overview of de-interlacing techniques and describes some advanced motion compensated de-interlacing techniques in more detail. These motion compensated methods are analyzed and mutually compared. It is concluded that the success of motion compensated de-interlacing depends critically on a protection mechanism. A good example is presented.

1 Introduction

Historically, interlacing has been introduced in video signals to reduce bandwidth. A major drawback of the interlaced scanning format on current bright high resolution displays is the line flicker and serration of moving edges. In the literature, several de-interlacing algorithms have been proposed to reduce these artifacts, or to serve as a base for other scan rate conversions.

De-interlacing may seem a straightforward application of general Sample Rate Conversion (SRC) theory, by just realizing an upconversion of a factor two. However, such an upconversion is only valid if the signal satisfies the Nyquist criterion. Interlaced signals are not vertically and temporally filtered prior to subsampling in order to satisfy this Nyquist criterion. Sampling in vertical and temporal direction is realized in the camera, which means that prefiltering should be realized in the optical path, which is very complicated in practice.

In addition to this practical problem, an even more fundamental problem exists. The two-dimensional upconversion cannot be solved correctly, since we do not know the temporal frequencies at the retina of a movement-tracking observer. For a tracking observer, very high temporal frequencies on the screen can be

transformed to lower frequencies or even DC at the retina. Consequently, suppression of these frequencies with a temporal lowpass filter, results in artifacts for this observer.

Many de-interlacing algorithms have been proposed in order to find a good balance between (hardware) cost and quality of the de-interlacing process. These de-interlacing algorithms range from simple intra-field interpolation methods to Motion Compensated (MC) methods, sometimes applying a generalization of the sampling theorem (GST). This paper presents a brief overview of some simple de-interlacing algorithms and describes in more detail some interesting MC de-interlacing methods. The evaluation of these algorithms shows a preference for the algorithm based on a generalization of the sampling theorem which is extended with a protection mechanism.

Section 2 provides an overview, without pretending to be complete, of several de-interlacing algorithms. Section 3 briefly introduces the 3D Recursive Search (3D-RS) block matcher of [1], which will be used in the analysis for the MC de-interlacing methods. Section 4 shows the result of experiments with these algorithms and finally in section 5 some conclusions are drawn.

2 De-interlacing techniques

Existing algorithms can be categorized in, either spatial, or spatio-temporal de-interlacing techniques. Spatial de-interlacing algorithms are by definition intra-field algorithms. Similarly, spatio-temporal algorithms are by definition inter-field techniques. This chapter presents an overview of some de-interlacing methods for both categories.

2.1 Spatial de-interlacing techniques

Spatial de-interlacing algorithms form the simplest category, and do not demand high hardware costs. These algorithms exploit the high correlation between the samples in the current field and the ones to be interpolated.

2.1.1 Line repetition

Line repetition is the simplest method, and defined by:

$$f_{out}(\vec{x}, n) \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ f\left(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right) & \text{else} \end{cases} \quad (1)$$

with $\vec{x} = (x, y)^t$ the spatial position and t for transpose, $f(\vec{x}, n)$ the input field and $f_{out}(\vec{x}, n)$ the de-interlaced output frame. Note that $y \bmod 2 = n \bmod 2$ is true for odd lines in odd fields and even lines in even fields only, which will be called *original* lines in this paper (the remaining lines of the de-interlaced image will be called *interpolated* lines). Note that $f(\vec{x}, n)$ is unknown for $\forall \vec{x} | (((y \bmod 2 = 0) \wedge (n \bmod 2 = 1)) \vee ((y \bmod 2 = 1) \wedge (n \bmod 2 = 0)))$.

2.1.2 Linear filtering

Linear vertical filtering can be a simple averaging filter defined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \frac{f\left(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right) + f\left(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right)}{2} & \text{else} \end{cases} \quad (2)$$

or a larger vertical filter with impulse response defined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \sum_k f\left(\vec{x} + \begin{pmatrix} 0 \\ 2k \end{pmatrix}, n\right) h(k) & \text{else} \end{cases} \quad (3)$$

$(k \in \{\dots, -1, 0, 1, 2, \dots\})$

2.2 Spatio-temporal de-interlacing techniques

Spatio-temporal de-interlacing algorithms require at least one field memory. This generally increases the cost of these methods, and the increased freedom in algorithms increases the quality of the de-interlacing.

2.2.1 Inter-field line averaging

Instead of vertical line averaging as realized in the spatial linear averaging de-interlacer of subsection 2.1.2, it is also possible to perform averaging in the temporal domain as shown in figure 1.

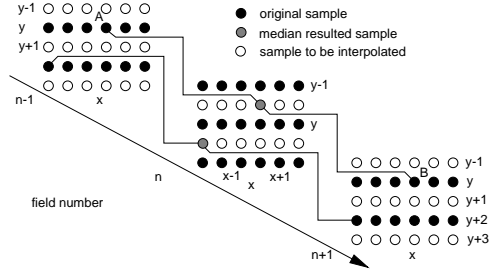


Figure 1: Inter-field line averaging

The output sample is determined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \frac{f(\vec{x}, n-1) + f(\vec{x}, n+1)}{2} & \text{else} \end{cases} \quad (4)$$

2.2.2 Vertical-temporal median filtering

Median filters, which belong to the class of order statistical filters, are very popular in image processing. Experimentally, it has also proven to be a very attractive method for de-interlacing [2]. The form most encountered in this application is the three-tap vertical-temporal median filter illustrated in figure 2.

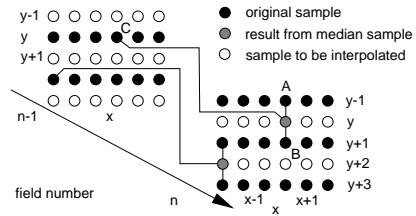


Figure 2: Vertical-temporal median filtering

The interpolated samples are found by median filtering the spatial neighbours in the vertical direction (A and B) and the corresponding sample in the previous

field (C):

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \text{median} \left\{ \begin{array}{l} f(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n), \\ f(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n), \\ f(\vec{x}, n-1) \end{array} \right\} & \text{else} \end{cases} \quad (5)$$

The underlying assumption is that in case of stationarity, $f(\vec{x}, n-1)$ is expected to have a value in between the vertical neighbours in the current field resulting in temporal interpolation, whereas in case of motion, intra-field interpolation results, since in that case, the correlation between the samples in the current field is expected to be the highest.

2.2.3 Weighted and edge dependent median filtering

Median filtering is an effective method for de-interlacing. The filter of subsection 2.2.2 is based on the assumption of the highest correlation in the vertical direction. However, this is not always optimal. With a high correlation in the diagonal direction, a diagonal interpolation is better. To cope with all possibilities, a larger neighbourhood is required.

In [3] and [4], a seven point spatio-temporal median filter was suggested applying an additional motion detector to control the importance or ‘weight’ of the individual pixels at the input of the median filter. Justusson [5] called this method weighted median filtering. Figure 3 shows the positions of the samples used in the median filter.

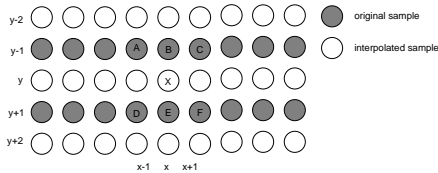


Figure 3: Spatial position of the samples participating in the weighted median filtering

The output is defined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \text{median} \left\{ \begin{array}{l} A, B, C, D, E, F, \\ \alpha f(\vec{x}, n-1), \\ \beta(B+E) \end{array} \right\} & \text{else} \end{cases} \quad (6)$$

with

$$\begin{aligned} A &= f\left(\vec{x} - \begin{pmatrix} 1 \\ 1 \end{pmatrix}, n\right) & B &= f\left(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right) \\ C &= f\left(\vec{x} - \begin{pmatrix} -1 \\ 1 \end{pmatrix}, n\right) & D &= f\left(\vec{x} + \begin{pmatrix} -1 \\ 1 \end{pmatrix}, n\right) \\ E &= f\left(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right) & F &= f\left(\vec{x} + \begin{pmatrix} 1 \\ 1 \end{pmatrix}, n\right) \end{aligned} \quad (7)$$

and α and β the integer weights where αA indicates the number of A 's that occur in expression (6). (For example $3A$ means A, A, A).

A motion detector is used to determine the weights [4]. A large value of α increases the probability of field insertion, whereas a large β increases the probability of line averaging at the output.

Doyle et al. [6] use the same neighbourhood of samples, but incorporate edge information. If intra-field interpolation turns out to be necessary because of motion in the image sequence, then preferably, the interpolation should preserve the frequency content of the image as good as possible. After determining the most stationary direction, the signal is interpolated in that direction.

The interpolated sample X , as shown in figure 3, is determined by a luminance gradient indication calculated from its direct neighbourhood:

$$f_{out}(\vec{x}, n) = \begin{cases} X_A & \begin{cases} (|A-F| < |C-D|) \wedge \\ (|A-F| < |B-E|) \end{cases} \\ X_C & \begin{cases} (|C-D| < |A-F|) \wedge \\ (|C-D| < |B-E|) \end{cases} \\ \text{Median}\{B, E, f(\vec{x}, n-1)\} & \text{else} \end{cases} \quad (8)$$

where X_A and X_C are defined by:

$$X_A = \frac{A+F}{2} \quad X_C = \frac{C+D}{2} \quad (9)$$

It is uncertain whether a zero difference between pairs of neighbouring samples, indicates the spatial direction in which the signal is stationary. For example, noise, or more fundamentally alias, can negatively influence the decision. An edge detector can be applied to switch or fade between at least two processing modes, each of them optimal for interpolation of a certain orientation of the spatial edge.

Some variations to this algorithm have also been proposed.

2.2.4 Edge detection based de-interlacing

Hentschel [7, 8] proposed to detect edges within a field, indicating vertical high frequencies, rather than motion.

The edge detector ED signal is defined by:

$$ED(\vec{x}, n) = g \left\{ f \left(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n \right) - f \left(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n \right) \right\}_{y \bmod 2 \neq n \bmod 2} \quad (10)$$

with $g()$ a (non-linear) function which determines when an edge is detected. The output of $g()$ is either 0 or 1.

If a transient is detected, the output is determined by field insertion, otherwise intra-field line averaging is used as defined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ f(\vec{x}, n-1) & \left\{ \begin{array}{l} (y \bmod 2 \neq n \bmod 2) \wedge \\ (ED(\vec{x}, n) = 1) \end{array} \right. \\ \frac{f \left(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n \right) + f \left(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n \right)}{2} & \text{else} \end{cases} \quad (11)$$

2.2.5 Vertical-temporal filtering

Linear vertical-temporal filtering can be applied for de-interlacing purposes as defined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \sum_{k,j} f \left(\vec{x} - \begin{pmatrix} 0 \\ 2k \end{pmatrix}, n-j \right) h(k, j) & \text{else (k integer)} \end{cases} \quad (12)$$

with $k = -L \dots L$ and vertical filter length of $2L+1$ taps, $j = -J \dots 0$ and the temporal filter length or depth. Since field memories are still relatively expensive in consumer electronics applications and temporal interpolation may result in severe artifacts, J is restricted to a small number.

2.2.6 MC de-interlacing

MC de-interlacing algorithms attempt to interpolate in the direction with the highest correlation, i.e. interpolation along the motion trajectory¹. However, not all temporal information changes can be adequately described with object velocities. Fades, concealed or obscured background, are some of the difficulties. Nevertheless,

¹The motion trajectory is defined by the line that connects identical picture parts from two successive pictures.

motion compensated de-interlacing methods have the strongest physical background, as due to their inertia, it always takes time for objects to completely disappear, or change geometry, resulting a strong correlation of successive images. This in contrast to spatial interpolation for which there is a statistical but no physical background.

MC median filtering

Vertical-temporal median filtering as discussed in subsection 2.2.2, generally performs line repetition in case of motion. If motion vectors are known, the correlation in the temporal domain can be increased by shifting the current spatial sample position, but in the previous field, over the motion vector. This is illustrated in figure reffig:4.

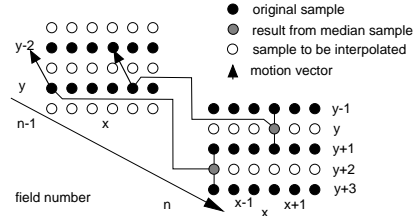


Figure 4: Motion compensated median filtering

With $\vec{d}(\vec{x}, n)$, the (sub-pixel accurate) motion vector at spatial position \vec{x} in field n , the output is defined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \text{median} \left\{ \begin{array}{l} f(\vec{x} - \vec{d}(\vec{x}, n), n-1) \\ f \left(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n \right) \\ f \left(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n \right) \end{array} \right\} & \text{else} \end{cases} \quad (13)$$

Temporal backward projection

Woods et al. [9] describe a de-interlacing algorithm that attempts to find the interpolated sample in the previous fields (backward projection), which is in this paper addressed as the Temporal Backward Projection (TBP) deinterlacing algorithm. The algorithm is illustrated in figure 5.

If the motion vector is known, the missing sample can be searched for in the previous field. If it is not within the vicinity of an existing pixel, the motion vector is extended into the pre-previous field. If still the vector is not addressing in the vicinity of an existing sample,

Woods et al. [9] propose to calculate the average between the nearest existing samples in the previous field. This algorithm implicitly assumes uniform motion over a two field period.

MC time-recursive de-interlacing

The MC Time-Recursive (TR) de-interlacing as proposed by Wang et al. [10] applies the motion compensated previously de-interlaced field, according to:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ f_{out}(\vec{x} - \vec{d}(\vec{x}, n), n - 1) & \text{else} \end{cases} \quad (14)$$

Once a perfect de-interlaced image is available, and the motion vectors are accurate, standard sample rate conversion theory can be used to calculate the lines to be interpolated in the current field. However, in practice, de-interlacing and motion vectors are not perfect. Besides, samples interpolated in the current frame are, generally, (partly) based on samples interpolated in the de-interlacing process of the previous field. As an implication, errors originating in an output frame, can propagate into later output frames. This is inherent to the recursive approach, and the most important drawback of this method.

To prevent serious errors from propagating, several solutions have been described in [10]. Particularly, the median filter is recommended to solve the problem. As a consequence, the TR de-interlacing is very similar to the motion compensated median filter approach, however, differs from this method in that the previous image consists of a previously de-interlaced field instead of

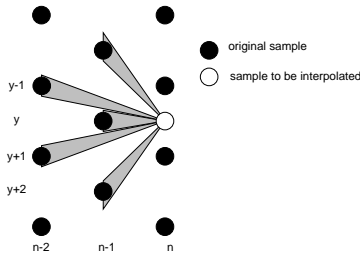


Figure 5: Temporal backward projection

the previous field. The output is defined by:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \text{median} \begin{cases} f_{out}(\vec{x} - \vec{d}(\vec{x}, n), n - 1), \\ f\left(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right), \\ f\left(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right), \end{cases} & \text{else} \end{cases} \quad (15)$$

Although this is a very effective method, the median filter can introduce alias in the de-interlaced image.

Generalized Sampling Theorem based de-interlacing

From the sampling theorem, it is known that a bandlimited signal with maximum frequency $\frac{f_c}{2}$ can exactly be reconstructed if this signal is sampled with a frequency of at least f_s . In 1956, Yen [11] showed a generalization of this theorem. Yen proved that any signal that is bandlimited by $0.5f_s$ can exactly be reconstructed by N independent sets of samples, sampled with frequency f_s/N . This theorem can effectively be used to perform de-interlacing as first presented by Delogne [12] and Vandendorpe [13]. We will call it GST *de-interlacer* method (Generalized Sampling Theorem).

Figure 6 shows the calculation of the samples to be interpolated. Samples from the previous field are shifted over the motion vector towards the current field in order to create two independent sets of samples valid at the same temporal instance. An appropriate filter that matches the desired interpolator can calculate the output sample.

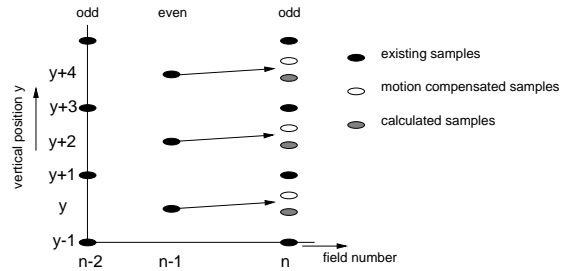


Figure 6: De-interlacing using a generalization of the sampling theorem

The calculation of the filter coefficients is explained in the papers of Delogne [12] and Vandendorpe [13]. Kalker [14] shows a generalization of this concept

which does not require the translation via the Fourier domain.

According to Kalker [14], the even output samples at temporal instance n , are calculated by (for vertical motion only²):

$$f_e(\vec{x}, n) = \sum_k f\left(\vec{x} - \begin{pmatrix} 0 \\ 2k+1 \end{pmatrix}, n\right)h_1(k) + \sum_m f\left(\vec{x} - \begin{pmatrix} 0 \\ 2m \end{pmatrix}, n-1\right)h_2(m) \quad (16)$$

where $f_e(\vec{x}, n)$ is the sample from the even field, with field number n at position (x, y) , $f(\vec{x}, n-1)$ the sample from field $n-1$ at vertical position (x, y) , $h(k)$ the desired filter impulse response which models the shift due to motion and the interpolator (which is splitted into $h_1(k)$ and $h_2(k)$), and $(f)_o$ the odd field of f . Note that, referring to figure 6, $f(\vec{x}, n-2)$ is unknown for $y \bmod 2 = 0$ and similarly $f(\vec{x}, n-1)$ is unknown for $y \bmod 2 = 1$.

Referring to the vertical component only, expression 16 can be rewritten into:

$$f_e(y, n) = \sum_k f(y - (2k+1), n)h_1(k) + \sum_m f(y - 2m, n-1)h_2(m) \quad (17)$$

In the z-domain, equation 17 is rewritten into:

$$F_e(z, n) = (F(z, n) \cdot H_1(z) + F(z, n-1) \cdot H_2(z))_e \quad (18)$$

with

$$\begin{aligned} H_1(z) &= \sum_k h_1(k)z^{-(2k+1)} \\ H_2(z) &= \sum_m h_2(m)z^{-2m} \end{aligned} \quad (19)$$

If we assume, we have the complete frame $F_p(z, n-1)$ available, (from which field $n-1$ is extracted), then:

$$F_e(z, n-1) = (F_p(z, n-1))_e \quad (20)$$

Field n can be reconstructed shifting the samples from frame $n-1$ over the motion vector, applying the desired interpolator, and extracting the desired field samples. So,

$$F_e(z, n) = (F_p(z, n-1) \cdot H(z))_e \quad (21)$$

where $H(z)$ describes the motion over one field period and the desired interpolation in the z-domain.

²Horizontal motion is irrelevant for this explanation, since it can be solved with simple sample rate conversion theory. Therefore, it is set to zero for clarity

Using the following set of characteristics:

$$\begin{aligned} x_o(z, n) &= \sum_{k \text{ is odd}} x(k, n) \cdot z^{-k} \\ x_e(z, n) &= \sum_{k \text{ is even}} x(k, n) \cdot z^{-k} \end{aligned} \quad (22)$$

$$\begin{aligned} (X(z, n) \cdot Y(z, n))_e &= X_o(z, n) \cdot Y_o(z, n) + X_e(z, n) \cdot Y_e(z, n) \\ (X(z, n) \cdot Y(z, n))_o &= X_o(z, n) \cdot Y_e(z, n) + X_e(z, n) \cdot Y_o(z, n) \end{aligned}$$

equation 21 results in:

$$F_e(z, n) = F_o(z, n-1) \cdot H_o(z) + F_e(z, n-1) \cdot H_e(z) \quad (23)$$

Similarly, the odd field at temporal instance n can be calculated according to:

$$\begin{aligned} F_o(z, n) &= (F_p(z, n-1) \cdot H_o(z))_o \\ &= F_o(z, n-1) \cdot H_e(z) + F_e(z, n-1) \cdot H_o(z) \end{aligned} \quad (24)$$

Solving $F_e(z, n)$ from equations 23 and 24 results in:

$$F_e(z, n) = a(z) \cdot F_e(z, n-1) + b(z) \cdot F_o(z, n) \quad (25)$$

with

$$\begin{aligned} a(z) &= H_e(z) - \frac{H_o^2(z)}{H_e(z)} \\ b(z) &= \frac{H_o(z)}{H_e(z)} \end{aligned} \quad (26)$$

If we assume a bilinear interpolator and a shift over $1 - \alpha$ modelling the motion:

$$H(z) = \alpha + (1 - \alpha)z^{-1} \quad (27)$$

the filter coefficients are defined by:

$$\begin{aligned} a(z) &= \alpha - \frac{(1 - \alpha)^2}{\alpha} z^{-2} \\ b(z) &= \frac{1 - \alpha}{\alpha} z^{-1} \end{aligned} \quad (28)$$

As an example, consider the situation of a motion of 0.5 pixels per field and a bilinear interpolator, then $\alpha = \frac{1}{2}$, and the output can be calculated according to:

$$f_{out}(\vec{x}, n) = f\left(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n\right) +$$

$$\frac{1}{2}f(\vec{x} - \vec{d}(\vec{x}, n), n - 1) + \quad (29)$$

$$\frac{1}{2}f\left(\vec{x} - \vec{d}(\vec{x}, n) + \begin{pmatrix} 0 \\ 2 \end{pmatrix}, n - 1\right)$$

$$y \bmod 2 \neq n \bmod 2$$

This is illustrated in figure 7.

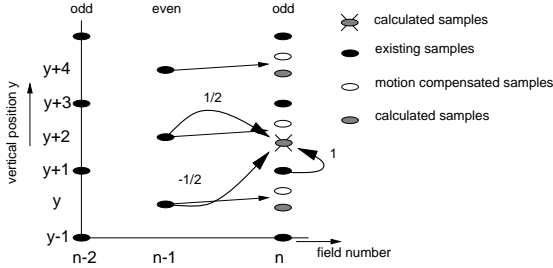


Figure 7: Calculation of the missing samples using generalized sampling

The output samples are determined by the original samples of the current and the previous field only. Therefore, errors, due to incorrect de-interlacing, will not propagate, which is a major advantage compared to the TR-algorithm.

The de-interlaced output is defined by³:

$$f_{out}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \sum_k f(\vec{x} - d(\vec{x}, n) - \begin{pmatrix} 0 \\ 2k+1 \end{pmatrix}, n)h_1(k) + \\ \sum_m f(\vec{x} - d(\vec{x}, n) - \begin{pmatrix} 0 \\ 2m \end{pmatrix}, n-1)h_2(m) & \text{else} \end{cases} \quad (30)$$

Selective-median GST de-interlacing

The GST method is effective, but error-sensitive for near-critical velocities as discussed in [15]. For near-critical velocities, shifted samples are mapped closely to original sample positions. The difference between the corresponding sample values greatly influences the value of the output sample. As a consequence, this difference is amplified, which boosts the noise level. Therefore, inaccuracies can occur, resulting in undesired artifacts for which a remedy turned out to be necessary.

³The input of the motion estimator, however, will be median filtered in order to avoid severe errors from propagating via the estimator, but not the output of the interlacer

Since a median filter has shown before to be effective in elimination of such errors, it would provide a sufficient solution. However, median filtering introduces alias in the output signal. Furthermore, only interpolation for nearcritical velocities demands a protection mechanism. In order to accomplish these characteristics, we introduced a selective median filter in [15].

The output $f_{outGSTseM}(\vec{x}, n)$ is accordingly defined by:

$$f_{outGSTseM}(\vec{x}, n) = \begin{cases} f(\vec{x}, n) & y \bmod 2 = n \bmod 2 \\ \text{median} \begin{cases} f_{outGST}(\vec{x}, n), \\ f(\vec{x} - \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n), \\ f(\vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, n) \end{cases} & \begin{matrix} \text{(near-critical} \\ \text{velocities)} \wedge \\ \text{(} y \bmod 2 \neq n \bmod 2 \text{)} \end{matrix} \\ f_{outGST}(\vec{x}, n) & \text{else} \end{cases} \quad (31)$$

with

$$f_{outGST}(\vec{x}, n) = \sum_k f(\vec{x} - d(\vec{x}, n) - \begin{pmatrix} 0 \\ 2k+1 \end{pmatrix}, n)h_1(k) + \sum_m f(\vec{x} - d(\vec{x}, n) - \begin{pmatrix} 0 \\ 2m \end{pmatrix}, n-1)h_2(m) \quad (32)$$

as in expression 30.

This approach will be referred to as the GST with selective median.

A block-diagram of a circuit to implement this algorithm is shown in figure 8.

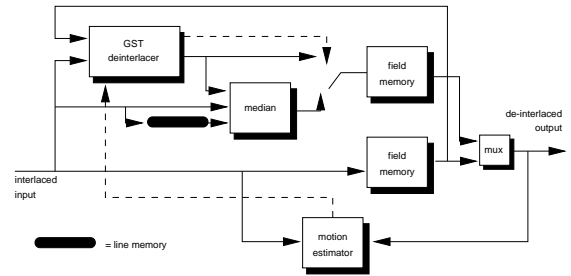


Figure 8: Block-diagram of a possible implementation

3 The 3D Recursive-search block matcher

The presented MC de-interlacing algorithms have been compared with each other, using the 3D RS block matcher of [1] as a basis for the common motion estimator.

This block matcher uses a small number of candidate vectors per block of pixels and realizes a quarter pixel accuracy. Furthermore, due to the inherent smoothness constraint, it yields very coherent vector fields that closely correspond to the true-motion of objects. This makes this method also suitable for scan rate conversion. This section briefly summarizes its characteristics.

In block-matching motion estimation algorithms, a displacement vector (or motion vector) $\vec{d}(\vec{b}_c, n)$ is assigned to the center $\vec{b}_c = (x_c, y_c)^t$, with t for transpose, of a block of pixels $B(\vec{b}_c)$ in the current n field by searching a similar block within a search area $SA(\vec{b}_c)$, also centered at \vec{b}_c , but in the previous field $n - 1$. This similar block has a center which is shifted with respect to \vec{b}_c over the displacement vector $\vec{d}(\vec{b}_c, n)$. To find $\vec{d}(\vec{b}_c, n)$, a number of candidate vectors \vec{C} are evaluated applying an error measure $e(\vec{C}, \vec{b}_c, n)$ to quantify block similarity. Figure 9 illustrates the procedure.

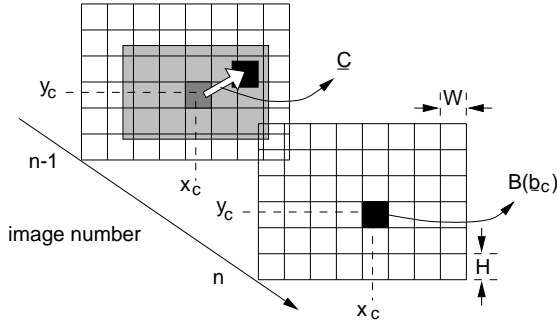


Figure 9: Illustration of block-matching

The block of pixels (positions) is defined by:

$$B(\vec{b}_c) = \left\{ (x, y) \mid \left(\left(x_c - \frac{W}{2} \leq x \leq x_c + \frac{W}{2} \right) \wedge \left(y_c - \frac{H}{2} \leq y \leq y_c + \frac{H}{2} \right) \right) \right\} \quad (33)$$

with W and H the block width and block height⁴, respectively, and the spatial position in the image.

The candidate vectors are selected from the candidate

⁴In our experiments, W was set to 8 pixels and H to 8 frames. (Block sub-sampling is neglected for clarity. A detailed description can be found in [1] and [16]).

set $CS(\vec{b}_c, n)$, which is defined by:

$$CS(\vec{b}_c, n) = \left\{ \begin{array}{l} (\vec{d}(\vec{b}_c - \begin{pmatrix} W \\ H \end{pmatrix}, n) + \vec{U}_1(\vec{b}_c, n)), \\ (\vec{d}(\vec{b}_c - \begin{pmatrix} -W \\ H \end{pmatrix}, n) + \vec{U}_2(\vec{b}_c, n)), \\ (\vec{d}(\vec{b}_c - \begin{pmatrix} 0 \\ 2H \end{pmatrix}, n - 1)), \end{array} \right\} \quad (34)$$

where the update vectors $\vec{U}_1(\vec{b}_c, n)$ and $\vec{U}_2(\vec{b}_c, n)$ are selected from an update set US , defined as:

$$US(\vec{b}_c, n) = US_i(\vec{b}_c, n) \cup US_f(\vec{b}_c, n) \quad (35)$$

with the integer updates $US_i(\vec{b}_c, n)$ defined by:

$$US_i(\vec{b}_c, n) = \left\{ \begin{array}{l} \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \\ \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix}, \\ \begin{pmatrix} 0 \\ 2 \end{pmatrix}, \begin{pmatrix} 0 \\ -2 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \\ \begin{pmatrix} 0 \\ 3 \end{pmatrix}, \begin{pmatrix} 0 \\ -3 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix} \end{array} \right\} \quad (36)$$

and the fractional updates $US_f(\vec{b}_c, n)$, necessary to realize sub-pixel accuracy, are defined by:

$$US_f(\vec{b}_c, n) = \left\{ \begin{array}{l} \begin{pmatrix} 0 \\ 0.25 \end{pmatrix}, \begin{pmatrix} 0 \\ -0.25 \end{pmatrix}, \\ \begin{pmatrix} 0.25 \\ 0 \end{pmatrix}, \begin{pmatrix} -0.25 \\ 0 \end{pmatrix}, \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \\ \begin{pmatrix} 0.5 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0 \end{pmatrix}, \\ \begin{pmatrix} 0.5 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0 \end{pmatrix}, \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix} \end{array} \right\} \quad (37)$$

Either $\vec{U}_1(\vec{b}_c, n)$ or $\vec{U}_2(\vec{b}_c, n)$ equals the zero update, $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

From these equations it can be concluded that the candidate set consists of spatial and spatio-temporal 'prediction vectors' from a 3D neighbourhood and an updated prediction vector. This implicitly assumes spatial and/or temporal consistency. The updating process involves updates added to either of the spatial predictions. Figure 10 shows where the spatial and spatio-temporal prediction vectors are located relative to the current block.

The displacement vector $\vec{d}(\vec{b}_c, n)$ resulting from the block-matching process, is a candidate vector \vec{C}

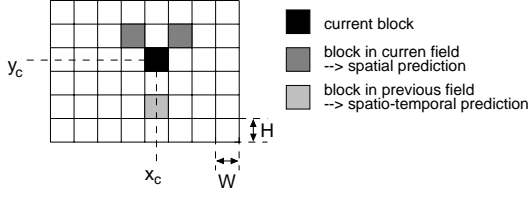


Figure 10: Positions, relative to the current block, from which the prediction vectors are taken in the 3D RS block-matcher.

which yields the minimum value of the error function $e(\vec{C}, \vec{b}_c, n)$:

$$\vec{d}(\vec{b}_c, n) = \{\vec{C} \in CS | e(\vec{C}, \vec{b}_c, n) \leq e(\vec{V}, \vec{b}_c, n) \quad \forall (\vec{V} \in CS(\vec{b}_c, n))\} \quad (38)$$

The error function is a cost function of the luminance values, $f(\vec{x}, n)$ with spatial position $\vec{x} = (x, y)^t$, of the pixels in the current block and those of the shifted block from the previous field, summed over the block $B(\vec{b}_c)$. A common choice, which we also use, is the Sum of the Absolute Differences (SAD). The error function is defined by:

$$\begin{aligned} e(\vec{C}, \vec{b}_c, n) &= SAD(\vec{C}, \vec{b}_c, n) \\ &= \sum_{\vec{x} \in B(\vec{b}_c)} |f(\vec{x}, n) - f_{out}(\vec{x} - \vec{C}, n - 1)| \end{aligned} \quad (39)$$

with $f_{out}(\vec{x}, n - 1)$ the de-interlaced output frame $n - 1$.

4 Evaluation

Spatial de-interlacing algorithms do not require a field memory, and are therefore relative cheap to implement. Spatio-temporal de-interlacing algorithms require at least a field memory, which increases the hardware costs, but the performance is also better then for the spatial algorithms. The best quality can be achieved by the MC spatio-temporal de-interlacing methods. However, the implementation cost increases again.

In this section, the presented MC techniques will be mutually compared. As a simple spatio-temporal de-interlacing algorithm, the non-MC median filtering technique is used as a reference.

4.1 Tools

Several tools can be used to evaluate the de-interlaced results ranging from objective measurements to subjective

evaluation. We preferred to use the objective measurement based on the mean-square-error, since it is used in the papers of the described de-interlacing methods. However, it is not always a reliable indicator. New tools which better reflect the relation between the measurement and the perception are still desired.

Several sequences with different characteristics were processed in order to evaluate the discussed methods. As an objective measure, the interlaced Mean-Square-Error, MSE_i , is calculated according to:

$$MSE_i(n) = \frac{1}{N_{MW}} \sum_{\vec{x} \in MW} (f(\vec{x}, n) - f_{out}(\vec{x} - \vec{d}(\vec{x}, n), n - 1))^2 \quad (40)$$

with MW indicating the Measurement Window, N_{MW} the number of samples within the measurement window, and $f(\vec{x}, n)$ the original samples in field n . All field lines within MW contribute to the interlaced MSE.

As an additional criterion, the Motion Trajectory Inconsistency, MTI [17, 15], will be calculated:

$$MTI(n) = \frac{1}{N_{MW}} \sum_{\vec{x} \in MW} (f_{out}(\vec{x} - \vec{d}(\vec{x}, n), n - 1) - f_{out}(\vec{x}, n))^2 \quad (41)$$

All frame lines within MW contribute to the MTI .

A low MTI score indicates a high correlation between the previous de-interlaced image and the currently calculated de-interlaced image, or more specific, it is a measure for the temporal consistency along the estimated motion trajectory.

As indicated in [17], a problem with this measure is that a good score is a necessary but insufficient constraint. Switching the output to zero, forces the MTI to zero, while the picture is seriously degraded. However, a lower MTI coupled to an almost stable (interlaced) MSE is a strong indication for quality improvement.

These measurements together form a useful tool to evaluate the alternative algorithms.

4.2 Results

The sequences used in the evaluation are *Renata* (ample vertical detail, horizontal velocities), *Mobile* (both slow horizontal and vertical velocities, including critical ones), *Shopping* (ample vertical and horizontal detail, with zoom and critical velocities), *RenataSpeed* (same as *Renata*, but accelerated 3 times), *Tokyo* (slow vertical and horizontal motion) and *Bicycle* (rotation, covering and uncovering). A snapshot of these sequences are shown in figure 11.

The results are categorized into two groups; one with (near) uniform motion (*Mobile*, *Shopping* and *Tokyo*),

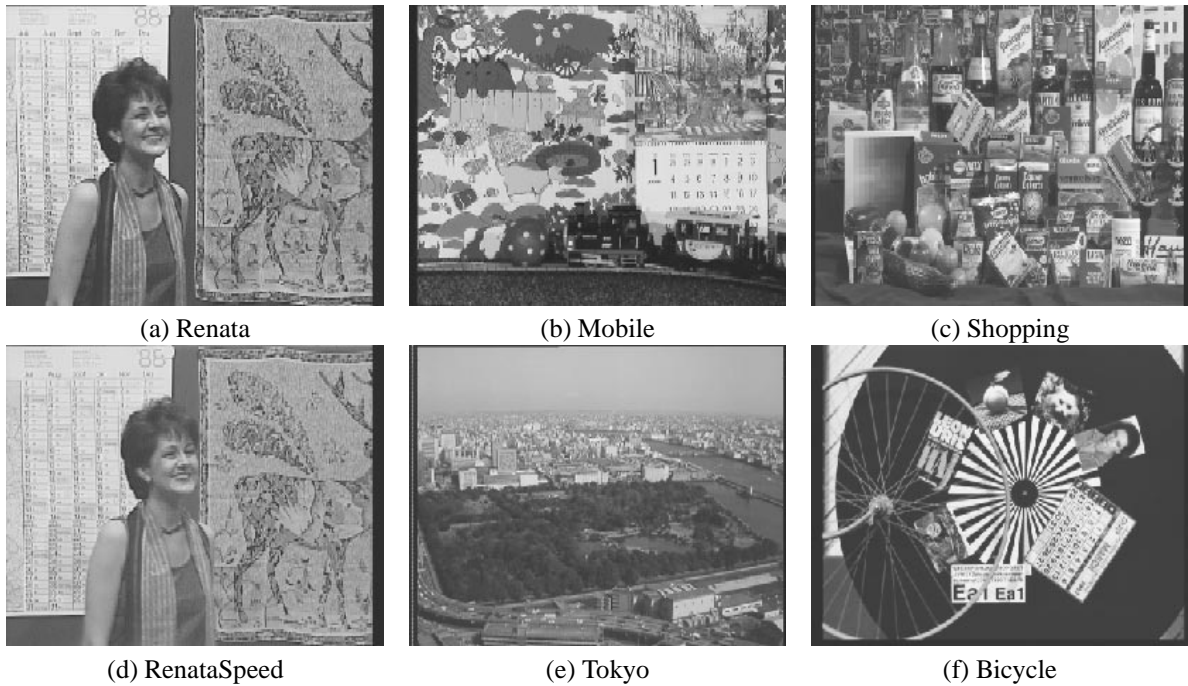


Figure 11: Images from the used sequences. Images b,c,e have nearly uniform motion, whereas a,d,f are typical non-uniform

and non-uniform motion (Renata, RenataSpeed, and Bicycle).

4.2.1 Selective median filter

The selective median filter is activated in case of near-critical velocities. The deviation around the critical velocity defining the region of median filter activity is called the aperture of the selective median filter. The distance between two samples is normalized to one.

In a first attempt to find the optimum aperture of the selective median, experiments were performed with the selective median switched on in case of near critical velocities which demand an interpolation of a quarter pixel position difference to an original sample, half pixel position difference, and three-quarter position difference. Furthermore, experiments were realized with no selective median (no median), and maximum aperture (always median filtering).

The average figures for all sequences have been plot in figure 12.

From figure 12 it is possible to conclude that the optimum aperture is between a quarter and half the distance from an original sample position. In the remaining part

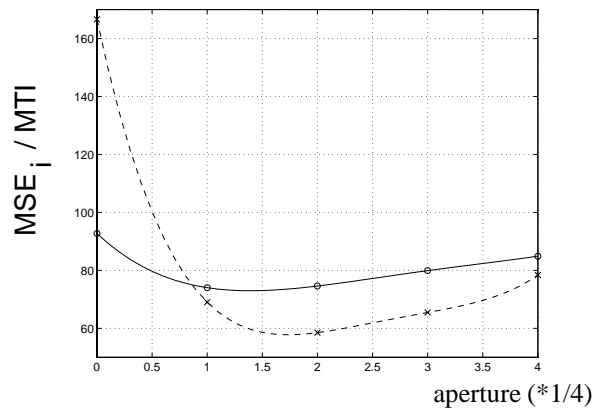


Figure 12: Aperture size for selective median versus MSE_i (solid line) and MTI (dashed line)

of this paper, an aperture of a quarter pixel distance to an original sample position is used for the selective median in the GST de-interlacer.

4.2.2 Error measurements

The results of the different de-interlacing algorithms in terms of MSE_i and MTI have been plot in figure 13

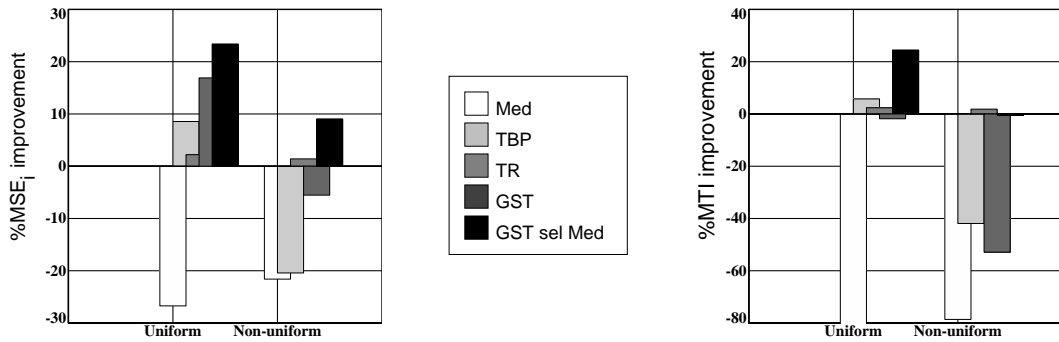


Figure 13: Percentage of MSE_i and MTI improvement compared to the MC median algorithm

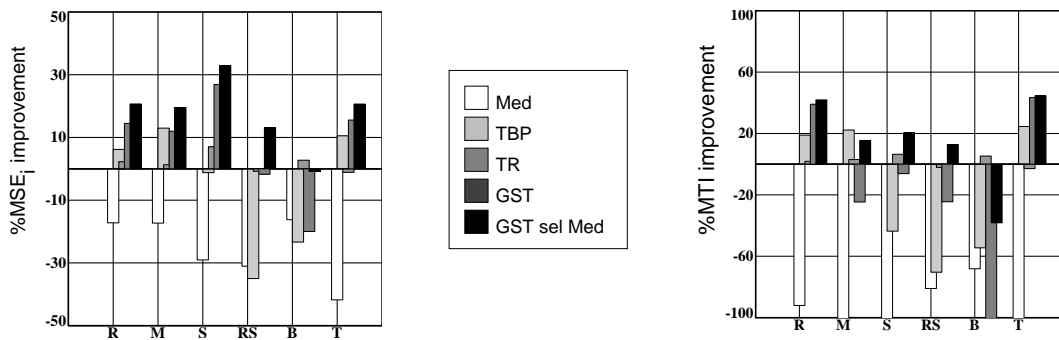


Figure 14: Percentage MSE_i and MTI improvement compared to the MC median method (R=Renata, M=Mobile, S=Shopping, RS=RenataSpeed, B=Bicycle and T=Tokyo).

and 14.

Some observations from these results:

- MC de-interlacing techniques are potentially superior to non MC techniques. In the figures 13 and 14, this is reflected in the comparison of the median (non-MC) de-interlacer with the MC techniques.
- The selective-median GST generally outperforms the other de-interlacing techniques. Apart from the sequence **Bicycle**, the MSE_i and MTI scores are the best of all methods. This selective-median GST method is, due to elimination of a permanent median filter, able to reconstruct vertical high frequencies in many cases, and also benefits from the median filter in case of high risks.
- Especially for non-uniform motion sequences, a protection mechanism is useful. As shown in figure 13, only the protected algorithms have a good score.
- The **Bicycle** sequence contains complex motion.

In this situation, the TR method is superior to the other methods. The selective-median GST does not always profit from the median. In the TR de-interlacer, the median filter is always active. Besides, **Bicycle** does not contain vertical high frequencies. Consequently, it can not benefit from the fact that the selective-median GST is capable of reconstructing such frequencies. The TBP method is invalid in this case, since the motion is not uniform over 2 field period for many part of the image.

- The TBP method has a good score for sequences with small velocities.
- The TR method performs generally better than the MC median approach, however, this difference is relative small in the score of these objective criteria.
- The improvement of the GST approach due to the protection mechanism is significant.

5 Conclusions

Several de-interlacing techniques have been evaluated. It has been shown that the MC de-interlacing methods are generally superior over the non-MC methods for moving sequences. The GST algorithm with the selective median outperforms the other de-interlacing methods. Only one sequence was found in which an alternative method performs better. For complex motion sequences (with no vertical high frequencies) the MC median and TR method are found to be superior over the other methods.

It can also be concluded that the objective improvement of the TR method compared to the MC median filtering is relatively small for the evaluated sequences. The major difference, in favour of the TR method, is expected for sequences with vertical velocities, since the de-interlaced field in the TR algorithm allows better interpolation filters to be used compared to field interpolation in the MC median filter algorithm. This is partly validated in the evaluation.

Another important conclusion of the evaluation is based on the fact that from the MC spatio-temporal algorithms, the high correlation between pixels in the current field and the interpolated ones is only exploited in the GST method. The benefit is obvious. However, it is concluded that the success of MC de-interlacing algorithms critically depends on a protection mechanism.

References

- [1] G. de Haan and P.W.A.C. Biezen, "Sub-pixel motion estimation with 3-D recursive search block matching," in *Signal Processing: Image Communication* 6. 1994, pp. 229–239, Elsevier.
- [2] M.J.J.C. Annegarn, T. Doyle, P.H. Frencken, and D.A. van Hees, "Video signal processing circuit for processing an interlaced video signal," European Patent Application no. EP-A 0 192 292.
- [3] J. Juhola, A. Nieminen, J. Salo, and Y. Nuevo, "Scan conversions using weighted median filtering," in *Proc. IEEE Int. Conf. on Circuits and Systems*, Portland OR, May 1989, pp. 433–436.
- [4] P. Haavisto, J. Juhola, and Y. Neuvo, "Scan rate up-conversion using adaptive weighted median filtering," in *Proc. 3rd Int. Workshop on HDTV and beyond*, Torino, 1989.
- [5] B.I. Justusson, "Median filtering: Statistical properties," in *Topics in Applied Physics*, T.S. Huang, Ed., vol. 43 of *Two-dimensional Digital Signal Processing II*, pp. 161–196. Springer-Verlag, Berlin, 1981.
- [6] T. Doyle and M. Looymans, "Progressive Scan Conversion using Edge Information," in *Proc. 3rd Int. Workshop on HDTV and beyond*, Torino, 1989.
- [7] C. Hentschel, "Comparison between median filtering and vertical edge controlled interpolation for flicker reduction," in *IEEE Tr. on Consumer Electronics*, August 1989, vol. VE-35,3, pp. 279–289.
- [8] C. Hentschel, *Fernsehen mit erhöhten Bildqualität, Flimmerreduktion durch erhöhter Vertikalfrequenz im Empfänger*, Ph.D. thesis, Technische Universität Braunschweig, Institut für Nachrichtentechnik, February 1990, Dissertation.
- [9] J.W. Woods and Soo-Chul Han, "Hierarchical Motion Compensated De-interlacing," *SPIE Visual Communication and Image Processing VI*, November 1991, Boston.
- [10] F.M. Wang, D. Anastassiou, and A.N. Netravali, "Time-Recursive Deinterlacing for IDTV and Pyramid Coding," in *Signal processing: Image Communications* 2. 1990, pp. 365–374, Elsevier.
- [11] J.L. Yen, "On Nonuniform Sampling of Bandwidth-Limited Signals," *IRE Tr. on Circuit Theory*, vol. CT-3, pp. 251–257, December 1956.
- [12] P. Delogne, L. Cuvelier, B. Maison, B. Van Cailie, and L. Vandendorpe, "Improved Interpolation, Motion Estimation and Compensation for Interlaced Pictures," *IEEE Tr. on Image Processing*, vol. 3, no. 5, pp. 482–491, September 1994.
- [13] L. Vandendorpe, L. Cuvelier, B. Maison, P. Quelez, and P. Delogne, "Motion-compensated conversion from interlaced to progressive formats," in *Signal Processing: Image Communication* 6. 1994, pp. 193–211, Elsevier.
- [14] A.A.C. Kalker, "Motion Estimation and Compensation for Interlaced Video," to be published in *IEEE Tr. on Signal Processing*.

- [15] E.B. Bellers and G. de Haan, “Advanced motion estimation and motion compensated de-interlacing,” in *Proc. of the Int. Workshop on HDTV*, Los Angeles, USA, October 1996.
- [16] G. de Haan, P.W.A.C. Biezen, H. Huijgen, and O.A. Ojo, “True-Motion Estimation with 3-D Recursive Search Block Matching,” *IEEE Tr. on circuits and systems for video technology*, vol. 3, no. 5, pp. 368–379, October 1993.
- [17] G. de Haan and P.W.A.C. Biezen, “Time-recursive de-interlacing for high-quality television receivers,” in *Proc. of the Int. Workshop on HDTV and the Evolution of Television*, Taipei, Taiwan, November 1995, pp. 8B25–8B33.