

Motion Adaptive Video Deinterlacing Using One Dimensional Fuzzy FIR Filters

A. Sanz¹, F. Fernández², J. Gutiérrez², G. Triviño², A. Sanchez¹, J.C. Crespo² and A. Mazadiego²

¹ ESCET-URJC, Campus de
Móstoles, 28933 Madrid, Spain
a.sanz@escet.urjc.es
an.sanchez@escet.urjc.es

² DTF-FI-UPM, Campus de
Montegancedo, 28860 Madrid, Spain
Felipe.Fernandez@es.bosch.com, jgr@dtf.fi.upm.es,
gtrivino@dtf.fi.upm.es, juanc_crespo@ieci.es
angel.mazadiego@mad.teccsidel.es

Abstract¹

This paper proposes a new fuzzy motion adaptive video deinterlacer. It is based on an orthogonal decomposition of the corresponding fuzzy motion detector into two main modules: a linear spatio-temporal low-pass filter described by two separable 1D FIR filters and two fuzzy modules described by two piecewise-linear saturation functions. The involved fuzzy system has been implemented in real time using MS DirectShow. Experimental results with several video benchmarks demonstrate the effectiveness of the considered algorithm in comparison with other alternative deinterlacing methods.

Keywords: Fuzzy filter, video processing, motion adaptive deinterlacing,

1 Introduction

At the time of the introduction of television, a video transmission system of 50 or 60 pictures per second was considered not be economically practicable. The interlaced video scan format was found to reduce the required signal bandwidth transmission. Nowadays this format is still used in the common NTSC, PAL and SECAM standards. In an interlaced scan format, video frames are split into odd and

even line fields, and they are transferred sequentially. This process gives images that are inherently incomplete in contrast to progressive scan formats.

On low cost television displays, the consequential line flicker is noticeable only at very fine detail and the interlaced scan profits from the characteristics of the human visual system, which is less sensitive to flickering details than to a large area flicker. So, interlaced video scan format was found the solution to economize video broadcast while preserving largely the vertical resolution and avoids the flicker of large areas.

Despite of the interlaced standard for common television broadcast, the evolution of high quality monitors and matrix displays went in a different direction. These types of displays avoid interlaced scanning and uses non-interlaced or progressively scanned displays. The required video bandwidth for these displays is double than for the interlaced displays, as the number of lines per picture is doubled, whereas the picture-update frequency is remained constant.

The technique used to reconstruct a final frame from each field, or group of fields, is called deinterlacing. This technique is a simple problem for stationary pictures since the alternating odd and even fields describe the captured scene. However, for non-stationary pictures, it requires to estimate from current and from

¹ This research has been supported by CICYT TIC2000-1420

neighboring pictures the missing information that most likely reconstructs the original scene.

There are many deinterlacing methods, commonly grouped in two main categories: motion compensated and non-motion compensated methods. Motion compensated (MC) algorithms provide the highest reconstruction quality. They are computationally more expensive because they require the estimation of two-dimensional motion vector and pixel shifting calculations. On the other hand, non-motion compensated (non-MC) methods are cheaper and can achieve a good compromise between performance and quality. An extensive review of de-interlacing technology is made by Gerard de Haan in [1,3], and Yao Wang in [12].

Simplest non-MC methods are based on time or space replication. This way, the missing lines of the actual field replicate the lines from the previous field (temporal replication) or replicating the lines of the actual field (spatial replication). Better results are achieved averaging known data by interpolation instead of replication.

Temporal techniques work better than spatial ones for static scenes, because when there is no motion, the missing lines are the same than the known previous ones. On the other hand, when there is motion in the scene, the previous lines contain information that is not coincident with the present data and tearing or combing artifacts appear in the moving regions (Figure 1). For these moving areas, spatial interpolation gives better results.

Temporal or inter-field techniques are also named weave methods, and spatial or intra-field techniques are also named bob methods.

Many spatio-temporal hybrid-deinterlacing techniques have been proposed to exploit the spatial and temporal correlation of video pictures and to overcome the artifacts associated with simple deinterlacers. The corresponding techniques called motion-adaptive (MA) algorithms, compute a motion-weighted combination of a temporal

interpolation function $It(.)$ and a spatial interpolation function $Is(.)$:

$$Its(i, j, t) = (1 - \alpha)It(i, j, t) + \alpha Is(i, j, t) \quad (1)$$

where $Its(i, j, t)$ is the obtained luminance on the column i , line j and time t of the corresponding field, and $\alpha \in (0,1)$ is the involved motion value. To compute this weighting parameter, most of these techniques are based on the computation of the absolute difference function $h(.)$ between the luminance of two adjacent frames to detect motion areas:

$$h(i, j, t) = |I(i, j, t+1) - I(i, j, t-1)| \quad (2)$$

Unfortunately, due to several noise sources, the luminance difference does not become zero in all picture parts without motion. This implies that the corresponding motion detector should include some kind of additional spatio-temporal filtering in order to avoid some undesirable noise effects.

This motion filter must be designed taking into account the following two main assumptions: the noise level is usually small in comparison to signal level and the moving objects are large compared with pixels size.



Figure 1. Unpleasant video combing artifacts.

The rest of the paper mainly describes the fuzzy motion detector proposed and the experimental deinterlacing results obtained using some standard benchmark videos.

2 Fuzzy Motion Detector

In MA methods, motion detection is critical and a nontrivial low-pass spatio-temporal motion filter is usual the key to obtain a

reliable deinterlacer. Motion detection failure can result in the use of inter-field data in moving parts of the video, where it can cause the appearance of combing artifacts (Figure 1). On the other hand, oversensitive motion detection can cause the motion detector to be triggered by noise, resulting in intra-field data in still parts of the picture. This can lead to noticeable loss of resolution in the video. Thus, there is a need to balance the algorithm's motion sensitivity with the ability to provide good resolution.

To accomplish this task, a fuzzy motion detector was developed by Van de Ville [9,10,11] based on a set of 5 fuzzy rules (*FMD1*). An improved version of this approach it is shown in [15]. This paper proposes an alternative fuzzy motion detector (*FMD2*) that simplifies the corresponding computation and provides a good picture quality in both moving and still image areas. The associated low computational cost gives the chance to implement it in software and to get a real-time execution with a low load on the corresponding general or specific processor.

The proposed *FMD2* is a fuzzy filter that is based on the serial composition of two linear 1D FIR low-pass filters [4] and two nonlinear saturation functions (Figure 2).

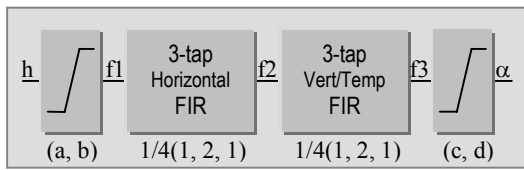


Figure 2. General scheme of *FMD2*.

The motion signal is subjected to an initial saturation function, a spatio-temporal filter and ulterior saturation-normalization function. The filter structure is decomposed into a horizontal filter and a vertico-temporal filter. These filters are used for smoothing and spreading the motion signal, such that the filtered signal is adequately expanded around moving edge boundaries. Insufficient spreading would result in noticeable tearing artifacts, caused from combining inter-field data in moving picture areas.

After completion of all linear filtering, the additional non-linear operation also normalizes the motion signal to values ranging between 0 and 1. The normalized motion signal α is then used as a weighting factor between the inter-field and intra-field components. For large motion values, interpolation is biased towards using intra-field components, which is required due to the variation between the successive video fields. In this case, the use of inter-field data in the presence of motion would result in visible motion artifacts due to the lack of correlation between the current and previous field. However, for small motion values, a high degree of temporal correlation exists between adjacent video fields, which necessitates heavily weighting the inter-field component.

The saturation parameters can be made dynamically dependent on the detected motion average, in each frame of the corresponding video sequence, by means of an additional frame fuzzy filter. For the sake of brevity, in this paper only the static case of saturation parameters is analyzed.

The recurrence equations of the proposed *FMD2* adaptive deinterlacing algorithm for the missing pixels in odd and even fields are:

```

System_of_recurrence_equations_FMD2 (I)
For i=1..N, j=1..M, t=1..K /*Video Sequence*/
If (j and t are odd) or (j and t are even) Then
    h(i,j,t)=|I(i,j,t-1)-I(i,j,t+1)| /* Motion input*/
    f1(i,j,t)=sata,b( h(i,j,t))*255 /*Input saturation*/
    /*1D 3-tap FIR low-pass filters: H and V-T*/
    f2(i,j,t)=1/4(f1(i-1,j,t)+2f1(i,j,t)+f1(i+1,j,t))
    f3(i,j,t)=1/4(f2(i,j-1,t-1)+2f2(i,j,t)+f2(i,j+1,t-1))
    alpha(i,j,t)= satc,d(f3(i,j,t)) /*Output saturation 0_1*/
    /* Output Luminance by T-S interpolation*/
    Its(i,j,t)= (1-alpha(i,j,t))*(I(i,j,t-1)+I(i,j,t+1))/2 +
                (alpha(i,j,t) )*(I(i,j-1,t)+I(i,j+1,t))/2
End If
End For
Function sat() /* sat(.): R->[0, 1] */
    satx1,x2(x)={ (x<x1)->0; (x>x1)->1; (x-x1)/(x2-x1) }
End Function

```

The chrominance components and RGB values are computed using the same weighting factor of luminance interpolation α .

The basic spatial data dependence graph of *FMD2* algorithm is shown in Figure 3.

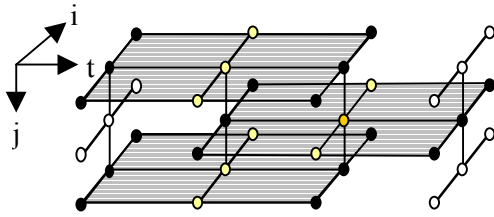


Figure 3. Spatial DDG of *FMD2* algorithm

Note that the saturation functions used capture the nonlinearities of the corresponding fuzzy filter [6]. The saturation function $s_{x_1, x_2}(x)$ has been initially specified by the set of fuzzy rules:

$\{IF (x \text{ is } LOW) \text{ Then } s=0; IF (x \text{ is } HIGH) \text{ Then } s=1\}$

where the fuzzy labels *LOW* and *HIGH* belong to the corresponding trapezoidal type-1 fuzzy partition [5] defined by the coordinates $(x_{min}, x_1, x_2, x_{max})$. This saturation function has been compiled during the fuzzy design process into the classical conditional function shown above. Parameters x_1 and x_2 simultaneously specify the threshold, gain and saturating regions of the corresponding variable.

The equivalent fuzzy filter obtained preserves the interpretability property of the original system, and is easily understandable for a fuzzy or classical system designer.

Moreover, the specification of two saturation functions at the input and output of the *FMD2* filter, by means of parameters (a, b) and (c, d), gives more flexibility to remove undesirable noise of the corresponding motion detector.

3 Fuzzy implementation in DirectShow

Microsoft DirectX [2] is a group of multimedia technologies designed for multimedia software developers. DirectShow is a DirectX component that takes charge of audio and video streams [7]. It is highly modular and based on Microsoft Component Object Model

(COM). Its objects called filters are classified into three major types: source filters, transform filters and renderer filters (Figure 4). GraphEdit is a DirectX visual tool for building and testing filter graphs (Figure 5). With GraphEdit is easy to build a custom filter graph, in a visual way, connecting the corresponding pins.



Figure 4. Basic filter graph of DirectShow

DirectShow can be used to solve problems in machine vision or for other kinds of audio or video processing, including the real-time processing of signals.

A DirectShow transform filter *EDAB-FMD2* (Figure 5) has been developed to encapsulate the proposed fuzzy algorithm, using GraphEdit for designing and testing purposes.

4 Results

In order to evaluate and compare the quality of the algorithm presented, some standard progressive video sequences have been tested. The interlaced video format has been emulated, by removing the corresponding even and odd lines and applying the algorithm in these missing lines. We have made a preliminary real time software implementation in a 1.2 GHz, 256 MB SDRAM Pentium III under Windows 2000 equipped with DirectX 9, and video sequences of size 176 x 144 pixels [8]. The throughput obtained was 30.18 FPS.

As for the results of deinterlacing algorithms, we are cautious about artifacts generation. Subjective visual quality assessment is necessary and complementary to objective video quality measurements [14]. Partial objective evaluation and partial human

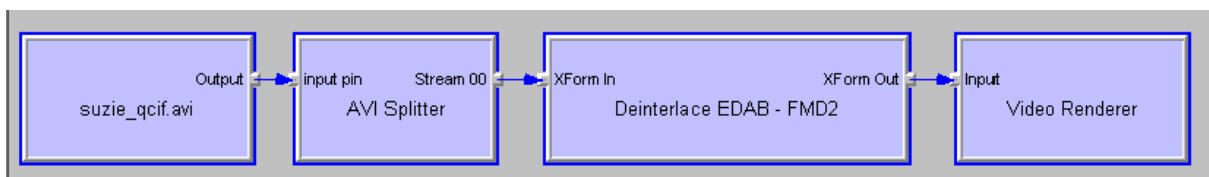


Figure 5. Filter graph *FMD2* in GraphEdit

evaluation were considered important to compare the efficiency of the corresponding deinterlacing algorithms in some specific parts of the video sequences.

Although regularly criticized, it seems that the common Mean-Square Error (MSE) and the related Peak Signal-to-Noise Ratio (PSNR) are still the most generally accepted measures as simple objective error criteria:

$$MSE = \frac{1}{N M K} \sum_{i,j,k} |I^*(i, j, k) - I(i, j, k)|^2 \quad (3)$$

$$PSNR = 10 \cdot \log \frac{255^2}{MSE} \quad (4)$$

Figure 6 shows the MSE results from the different deinterlacing algorithms for each frame of Suzie sequence from #5 to # 150. The MSE values are obtained excluding border pixels because their contribution is not determined with the uniform filter mask defined.

Table 1 shows the average MSE and PSNR

comparison of the different deinterlacing algorithms for two standard video sequences: Suzie sequence from frame #5 to #150 and Salesman one from frame #5 to #453. We have observed that in general Suzie sequence presents more motion than Salesman sequence. Consequently, for Suzie sequence we obtain better results with a simple spatial interpolation method $Is(av)$ than with a simple temporal one $It(av)$. The proposed method $FMD2: Its$ ($a=4, b=9, c=10, d=255$) combines spatial and temporal interpolation, uses horizontal and vertical-temporal weighting averaging windows, and it is more accurate and robust than simple line replicating (bob) and field insertion (weave) methods. The weighting factor for the spatio-temporal interpolation is the motion coefficient α extracted by means of the FIR filters and saturation functions defined. Algorithm $FMD1$ [10] gives in some cases similar quality results but it is less flexible since it has only one input saturation function and it is clearly more computational demanding.

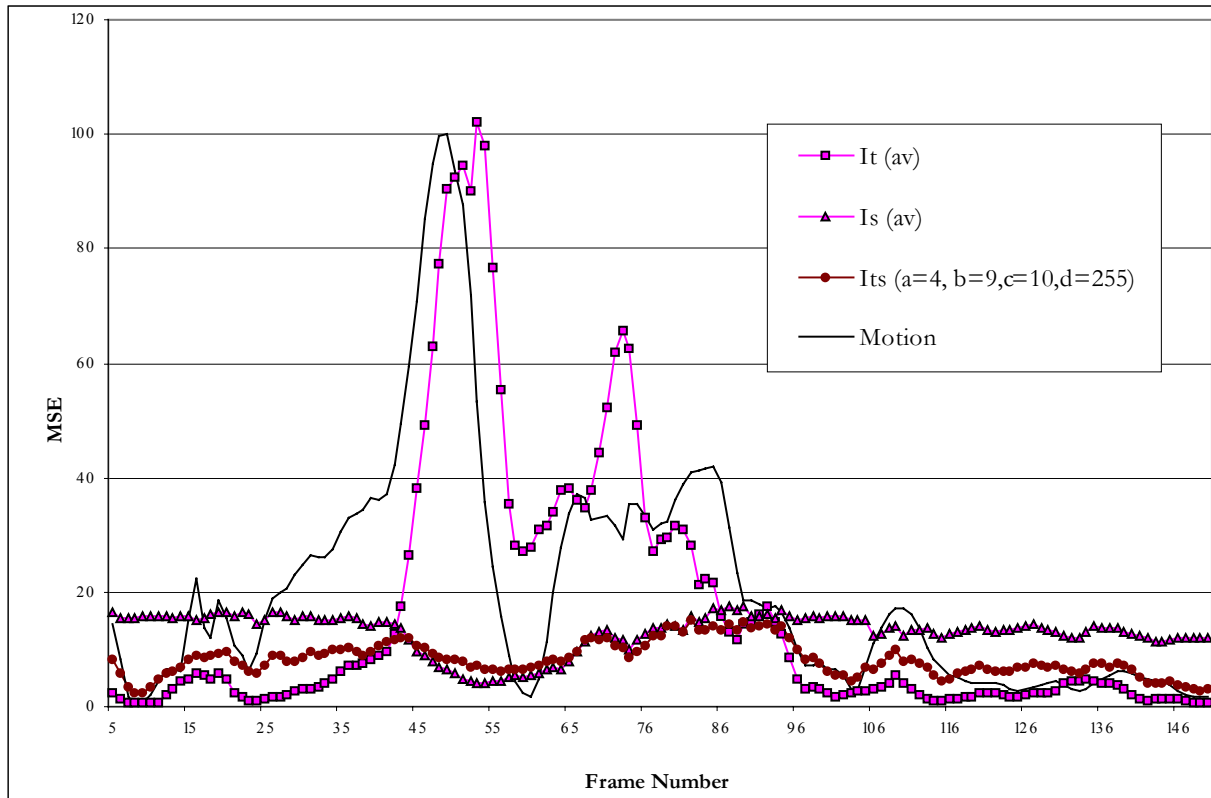


Figure 6. MSE comparison among different deinterlacing methods for Suzie sequence: time averaging (It), line averaging (Is), and the proposed fuzzy algorithm $FMD2$ (Its) with saturation parameters (a, b, c, d)=(4,9,10,255). The graph also shows the amount of motion calculated for each sequence frame.

It is relevant to observe that the degree of freedom, with four saturation parameters, is extremely high, and in general, parameters that suit very well for static scenes (a_0, b_0, c_0, d_0) tend to show some undesirable artifacts when motion is present. Conversely, parameters that suit very well for moving scenes (a_1, b_1, c_1, d_1) tend to show some undesirable artifacts in static frames.

Table 1: Average MSE and PSNR (dB) comparison chart for the Suzie <5,150> and Salesman <5, 453> sequences.

METHOD	Suzie		Salesman	
	MSE	PSNR	MSE	PSNR
<i>Line replicat.</i>	39.11	32.21	129.46	27.01
<i>Line average.</i>	13.07	36.97	58.83	30.43
<i>Field insertion</i>	40.77	32.03	6.98	39.69
<i>Time average.</i>	16.97	35.83	2.90	43.50
<i>FMD1</i>	7.89	39.16	11.93	37.36
<i>FMD2</i>	8.28	38.94	4.31	41.79

The election of the parameters is very critical for the filter performance, but on the other

hand we can achieve very good results by adjusting conveniently the saturation functions of the fuzzy filter.

Figure 7 shows the MSE results of *FMD2* algorithm for each frame of Salesman sequence with different saturation parameters. The optimal saturation parameters, for this relative low-movement sequence, are in this case: $(a, b, c, d) = (1, 200, 2, 50)$.

Figures 8 and 9 show the visual results of different deinterlacing methods applied to Suzie sequence: for frame #48 (with high motion average) and for frame #8 (with low motion average) respectively.

We also evaluate the deinterlacing algorithms with a more complex quality assessment: the Structural Similarity Index Measure (SSIM) proposed by Wang et al. [13]. This index expresses the similarity between two images by weighting more factors than MSE measures, which just evaluate intensity differences. This measure is applied to individual frames of the sequence.

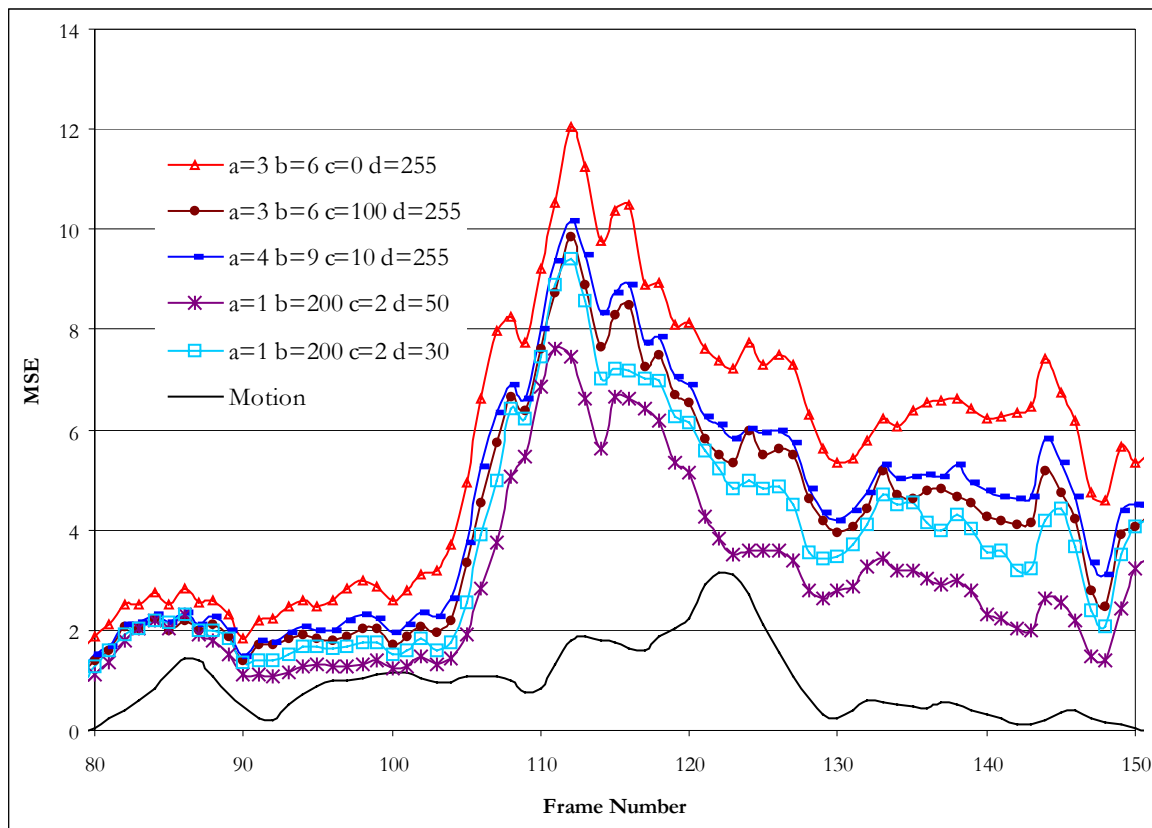


Figure 7. MSE results for the Salesman sequence between frames #80 and #150 for different saturation parameters. The graph also shows the amount of motion of each sequence frame.



Figure 8: Frame # 48 of Suzie sequence. Top to down: a) Line-repeating algorithm; b) Field insertion method; c) Saturated fuzzy motion detection; d) Proposed algorithm *FMD2* with $(a_1, b_1, c_1, d_1) = (2, 3, 0, 100)$.



Figure 9: Frame # 8 of Suzie sequence. Top to down: a) Line-repeating algorithm; b) Field insertion method; c) Saturated fuzzy motion detection; d) Proposed algorithm *FMD2* with $(a_0, b_0, c_0, d_0) = (2, 15, 50, 255)$.

In particular we tested it with a static frame and with a frame with a lot of motion of Suzie sequence (Table 2). Roughly speaking we observe that the results are approximately the same than those obtained by the MSE evaluation.

Table 2: SSIM indexes for two individual frames of the Suzie sequence with low and high motion.

METHOD	Frame#8	Frame#48
<i>Line replication</i>	0.9299	0.9489
<i>Line averaging</i>	0.9689	0.9798
<i>Field insertion</i>	0.9904	0.8285
<i>Time averaging</i>	0.9963	0.8882
<i>FMD2</i>	0.9961	0.9636

5 Conclusions

A DirectShow fuzzy motion adaptive deinterlacer has been presented in this paper that gives a good performance/cost trade-offs. It is mainly based in a fuzzy motion detector filter composed of two linear 1D FIR low-pass filters and two nonlinear saturation functions. The corresponding deinterlacing process has high efficiency concerning objective and subjective quality criteria. The chain of saturation functions and FIR filters developed allow improving the accuracy of the involved motion detector.

The obtained results for video deinterlacing, in performance and computational cost, are significantly better than the results of other currently implemented non-MC systems.

References

- [1] E. B. Bellers and G. de Haan, *De-interlacing. A Key Technology for Scan Rate Conversion*, Elsevier, 2000.
- [2] DirectX Microsoft Developer Center, <http://msdn.microsoft.com/directx>
- [3] G. de Haan, E. B. Bellers, De-interlacing. An Overview, *Proceedings of the IEEE*, Vol. 86, N^o. 9, pp 1839-1857, September 1998.
- [4] J. Jostschulte, A. Amer, M. Schu, H. Schröder, Perception Adaptive Temporal Tv-noise Reduction Using Contour Preserving Prefilter Techniques, *IEEE Transactions on Consumer Electronics*, Vol. 44, No. 3, pp 1091-1096, 1998.
- [5] G. J. Klir, B. Yuang, *Fuzzy Sets and Fuzzy Logic*, Prentice Hall, 1995.
- [6] M. Nachtgael, D. Van der Weken, D. Van de Ville, E. E. Kerre (Eds), *Fuzzy Filters for Image Processing*, Springer, 2003.
- [7] M. D. Pesce, *Programming Microsoft DirectShow for Digital Video and Television*, Microsoft Press, 2003.
- [8] QCIF 176 x 144, *Test video sequences* <http://thanglong.ece.jhu.edu/~cjtj/link.html> .
- [9] D. Van de Ville, B. Rogge, W. Philips, I. Lemahieu, Deinterlacing using fuzzy-based motion detection, In *Proceedings of the IEEE Third International Conference on Knowledge-Based Intelligence Information Engineering Systems*, Adelaide, Australia, September 1999.
- [10] D. Van de Ville, W. Philips, W. Philips, I. Lemahieu, Fuzzy-Based Motion Detection and its Applications to De-interlacing, In *Fuzzy Techniques in Image Processing*, E. E. Kerre and M. N. Nachtgael (Eds), Chapter 13, pp. 337-369, Physica-Verlag, 2000.
- [11] D. Van de Ville, R. Van de Walle, Philips, I. Lemahieu, Motion Adaptive De-interlacing using Fuzzy Logic, In *Proceedings of the conference IPMU'2002*, pp 1989-1996, Annecy, France, July 2002.
- [12] Y. Wang, J. Ostermann and Y. Zhang, *Video Processing and Communications*, Prentice Hall, 2001.
- [13] Z. Wang, L. Lu, and A. C. Bovik, Video quality assessment based on structural distortion measurement, *Signal Processing: Image Communication*, Special issue on "Objective video quality metrics", vol. 19, no. 1, Jan. 2004.
- [14] S. Wolf and M. Pinson, Video Quality Measurements Techniques, *NTIA Report 02-392*, June 2002.
- [15] J. Gutiérrez et al., Motion Adaptive Fuzzy Video De-interlacing Method Based on Convolution Techniques. *Proceedings of IPMU 2004*, July 4-9 2004.