AVC-168

:NTERNATIONAL STANDARDIZATION ORGANIZATION

ISO/IEC JTC1/SC2/WG11

()DING OF MOVING PICTURES AND ASSOCIATED AUDIO

Proposal 40

MPEG 91/232

Wavelet/subband based TV codec for bitrates up to 10 Mbps

This proposal is submitted by:

T. Ebrahimi (EPFL, Swiss Federal Institute of Technology at Lausanne)
phone: +4121 693 2796
fax: +4121 693 4660

and has been developed in the framework of the
European VADIS/COST collaboration

#40

Summary specification of EPFL proposal for MPEG II

Signal Processing Laboratory
Swiss Federal Institute of Technology at Lausanne
CH-1015 Lausanne, Switzerland

Picture format

A large variety of different formats are supported. In particular the 4:2:2 level of CCIR 601 and the CCITT/CIF

Deinterlacing

The deinterlacing is performed by motion compensated spatial interpolation for the luminance component and by field skipping for chrominance components.

Subband decomposition

The spatial correlation is reduced using a tree structure subband decomposition (wavelet transformation). This wavelet subband transform leads to a hierarchical data structure.

Filter bank specification

The filter bank contains a class of bi-orthogonal filters. The analysis filters are 10 taps with coefficients in powers-of-two. The synthesis filters are 6 taps with coefficients in powers of two. In addition they are optimally localized in the joint spatial/frequency domain.

The filters are rectangular separable and are realized in polyphase structure.

Motion estimation

The motion is estimated by block matching. The search is performed using a coarse to fine strategy (hierarchical). A two-step search is applied at each level, instead of the full search, to reduce the computational complexity. A median filtering is applied after each level to reduce the propagation of errors in estimation.

The accuracy of the motion vectors is limited to integer pels. The maximum detected displacement is ±21 pels.

Coding modes

Intraframe coding is applied after each group of 10 frames in order to reduce the propagation of errors and to access in a short time a randomly selected frame.

Predictive and interpolative modes are applied alternatively.

Scalar quantization

A scalar quantizer is applied to the output of the subband decomposition stage. The same step size is used for all the subbands, and the entire scene.

Adaptive entropy coding

An adaptive arithmetic coder is used to generate the bitstream. The adaptive entropy decoder has its own internal model, so that no side information is needed to be transmitted.

A similar adaptive arithmetic coder is used to code the motion vectors.

The entropy coder is frequently reset in order to reduce the propagation of channel errors.

Reinterlacing

The reinterlacing is performed by simply taking the even and odd lines of the decoded frame as odd and even fields.

Some additional features

The proposed codec is not block based.

A generic coding is inherently possible.

A fully progressive frame coding is straightforward.

A partial decoding of the bitstream for a lower resolution decoding is possible. This property is spacially very suitable for fast forward mode without a buffer overflow.

A progressive transmission is possible.

The proposed codec is particularly suitable for both constant bitrate and variable bitrate transmissions.

A frequency weighting matrix for quantization is straightforward but not implemented.

The forward and backward compatibility between two different levels of resolution is possible.

#40

EPFL proposal for MPEG II

Touradj Ebrahimi, Frédéric Dufaux,
Iole Moccagatta, Pierangela Cicconi
and Murat Kunt

Laboratoire de Traitement des Signaux
EPFL-Ecublens (DE)
CH-1015 Lausanne, Switzerland

tel: +4121 693 2796
fax: +4121 693 4660
email: ebrahimi@ltssg1.epfl.ch

1 General description

This paper describes briefly the EPFL proposal for MPEG II. Although the MPEG requirement only covers the bitrate range of 1 to 10 Mbps, the proposed codec is capable to function at higher and lower bitrates without any change in its general strategy. In particular, this codec can handle any input format and it is completely independent of the initial resolution of the input signal. By the same occasion a total forward and backward compatibility can be maintained within different formats and different resolutions. The core of the system works on progressive scan, however a deinterlacing module at the input level permits to support both interlaced and progressive formats such as ISO/CCIR 601 which is required by MPEG II and CCITT /CIF format which is not required by MPEG II but very frequently used in several applications. The interlaced to progressive conversion is performed by a spatial motion compensated interpolation. For progressively scanned formats the conversion module is simply bypassed.

The spatial correlation is reduced by a Gabor-like wavelet transformation. Among all filters of a given length, the Gabor filter has the best frequency response with respect to the Heisenberg uncertainty principle. The wavelet transformation used here performs octave bands partitioning of the spatial-frequency domain. This is motivated by typical image statistics and also by the spatial frequency sensitivity of the human visual system. In addition, this partitioning produces a multiresolution structure. Such a structure is very suitable for a generic coding scheme by discarding or taking into account a certain number of levels in the multiresolution pyramid. For example, an HDTV sequence can be decoded by a TV decoder and vice versa.
The temporal correlation is reduced by a hierarchical motion estimation. The motion field in the scene is computed between the previous and the current frame. The motion estimation procedure starts by estimating the motion field at a low resolution level and then the obtained results are injected into the next level of resolution as initial estimates and refined at this resolution level. this procedure is repeated until the motion vectors at the original resolution are obtained. This procedure is very efficient in computational cost and leads to accurate and robust estimations. Once the motion field is obtained. the current frame is then predicted and only the error of this prediction is coded. A fully predictive coding has however the disadvantage that the uncovered areas cannot be predicted, and the error image has a high energy at occluded

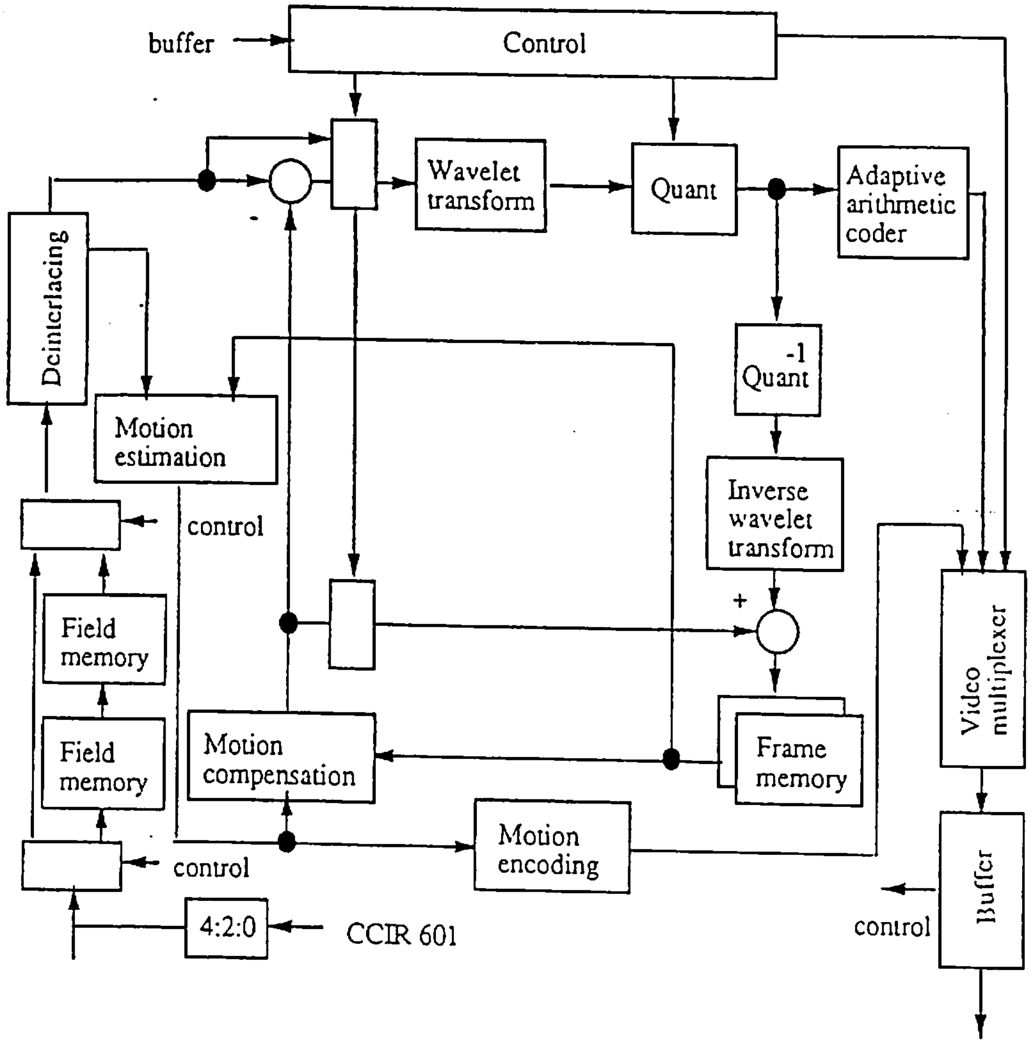


Figure 1: Block diagram of the coder.

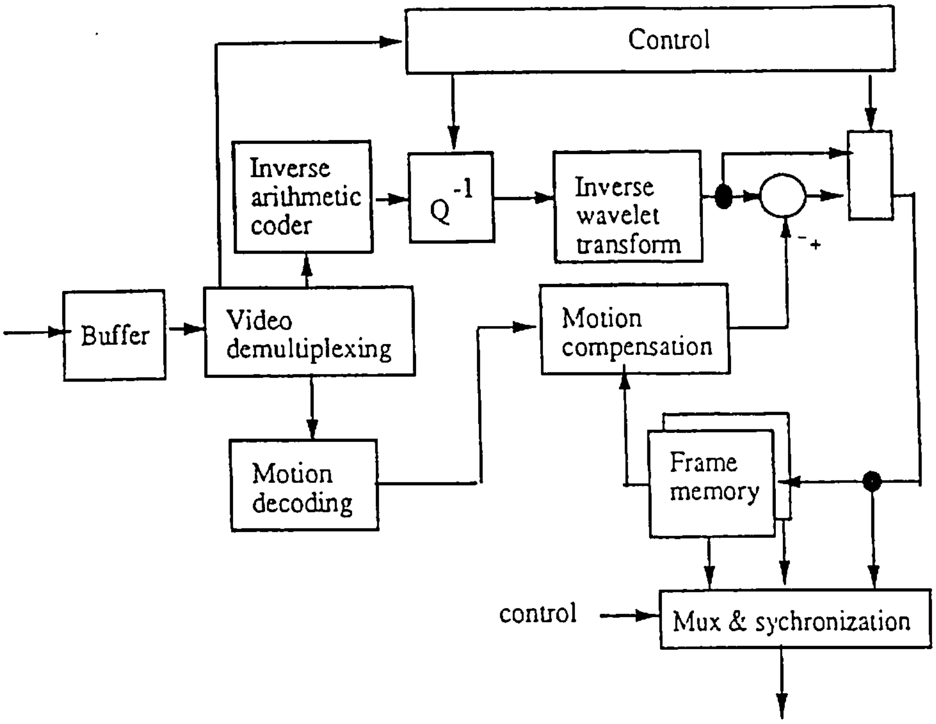


Figure 2: The decoder block diagram.

points. This disadvantage can be reduced partially by inserting interpolative modes between predictive ones. Errors due to occlusions are then reduced considerably. However this procedure will introduce an additional frame memory and an additional frame delay. In our current system the same vectors are used for the predictive and the interpolative modes. Furthermore in order to avoid the propagation of errors in estimating the vector field at each resolution level, a median filtering is applied at the end of motion estimation for each level of resolution. To prevent the propagation of errors, and to be able to access in a short time any particular frame randomly, within each group of 11 frames the first one is always intraframe coded.

The coefficients after the transformation stage are all uniformly scalar quantized. The same step size is used inside each subband, and inside the entire scene. A generalization into different quantization steps per subband is straightforward by introducing a visually significant weighting matrix, but it is not used in the current system.

The output of the quantization stage is entropy coded by an adaptive arithmetic coder. The adaptation is performed by generating and updating an internal histogram of symbols after an already defined period of coded symbols. The same arithmetic coder, with a different adaptive model, is used to code the motion vectors. To avoid the propagation of the errors the models are reset frequently. In the current system the reset is applied after each frame.

The chrominance components are first subsampled in the vertical direction by a simple field skipping, and then they are treated exactly in the same way as for the luminance component. For the motion compensated predictions and interpolations, the vectors obtained for the luminance component are used, where the amplitudes and their spatial resolutions are divided by two in each direction.

Figure 1 represents the block diagram of the proposed coder, and the decoder block diagram is presented in figure 2.

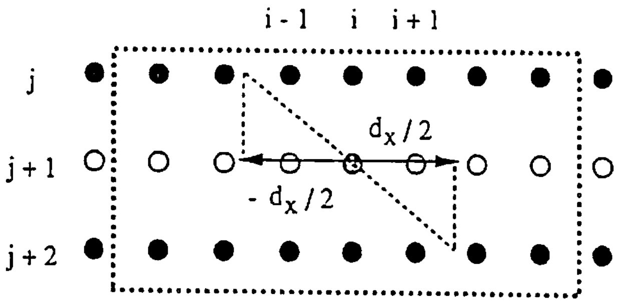


Figure 3: The deinterlacing procedure.

2 Interlaced to Progressive Conversion

If the input is in an interlaced format, a first block implements the interlaced to progressive conversion. The system can be used to code both ISO/CCIR 601 (interlaced) and CCITT/CIF (progressive) formats. The conversion is based on motion compensated interpolation. Each field in the scene is interpolated into a progressive frame, by predicting the missing lines between every two lines in the current field. Figure 3 illustrates the interpolation procedure. The black points in the figure represent the actual lines in the frame and the white pixels are to be interpolated. The pixel to be interpolated is represented in gray in Figure 3. We assume a simple translation between two portions of lines included in a window around this pixel. This displacement is estimated with a 1-D spatio-temporal constraint, and used for a more accurate interpolation. Because of the computational complexity, the same displacement is assigned to a number of neighboring pixels.
The reinterlacing is performed in the decoder by simply taking half of the lines of the decoded frames and displaying them alternatively.

3 Gabor-like wavelet with fast implementation

The proposed video codec system performs a Gabor-like wavelet transform by recursive filtering. The basic structure is a 1-D two channel frequency decomposition. The high frequency filter is obtained from the low frequency filter by a π shift. The low frequency filter in its turn is an approximation of a gaussian where the frequency response at points π and $-\pi$ have been set to zero. This operation makes the high frequency filter's response have a zero at DC. By applying the basic two channel decomposition in a recursive way, all the frequency channels, except the low pass one, will have a zero at DC (see Figure 4). Moreover, the synthesis filters are approximated by a filter having coefficients in powers-of-two. The analysis filters are obtained from the relation between the analysis and the synthesis filters for an exact reconstruction, and approximated by filters having coefficients in power-of-two. Filters with coefficients in powers-of-two have very attractive implementation properties. All values can be generated from shift and addition operations. Figure 5 gives the analysis and the synthesis filters frequency response, for real and power-of-two coefficients. The use of the Gabor-like wavelet transform is motivated on the one hand by the fact that Gabor functions, which are gaussians modulated by complex exponentials, have optimal joint localization in the spatial/spatial-frequency domain. On the other hand, according to recent experiments, the majority of the receptive field profiles of the mammalian visual system can be fit quite well by this type of functions.
A number of remarks seem necessary to be given here. The complexity of the synthesis filters is less than those of the analysis filters. This feature is very suitable for broadcasting applications, because the receivers can be very simple. Both synthesis and analysis filters are very short in the time domain and both of them contain only coefficients in powers-of-two. This is a further simplification in order to make

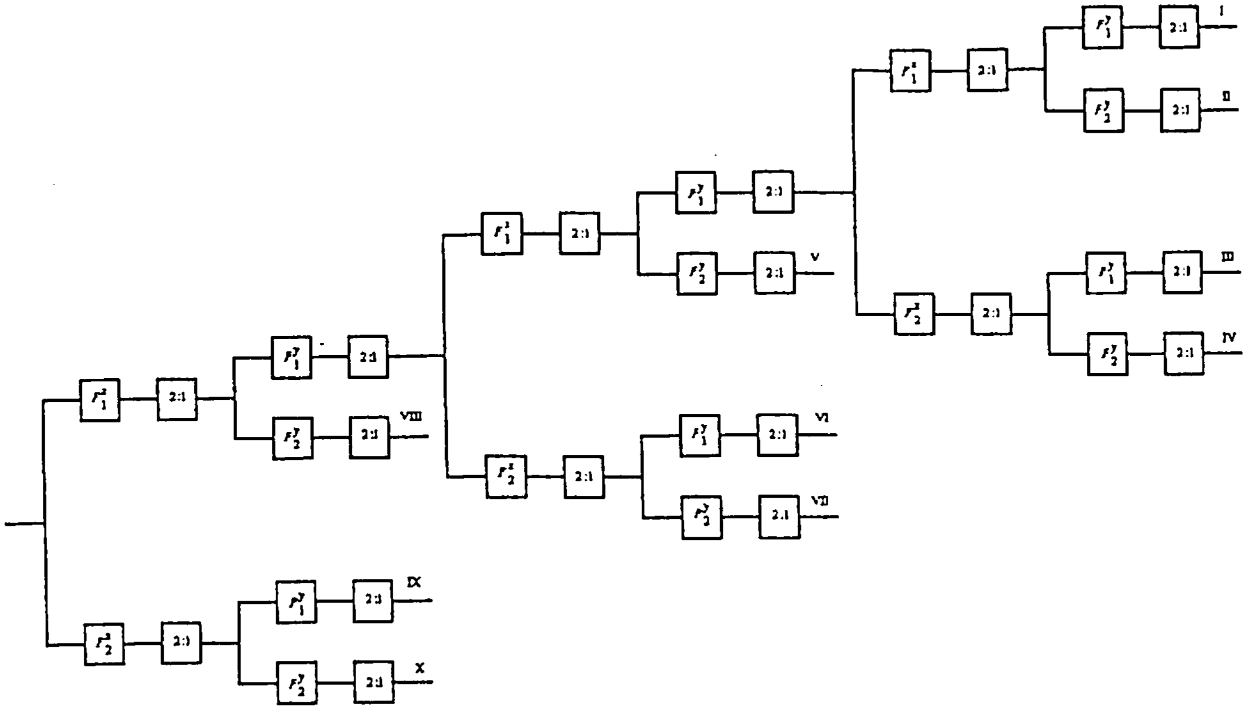


Figure 4: Tree structure recursive filtering to perform the fast Gabor-like wavelet transform.

Filter	*Coefficients Value*									
$g_0(\cdot)$	2^{-7}	2^{-3}	2^{0}	2^{0}	2^{-3}	2^{-7}				
$g_1(\cdot)$	-2^{-7}	2^{-3}	-2^{0}	2^{0}	-2^{-3}	2^{-7}				
$f_0(\cdot)$	2^{-6}	0	-2^{-3}	-2^{-7}	2^{0}	2^{0}	-2^{-7}	-2^{-3}	0	2^{-6}
$f_1(\cdot)$	-2^{-6}	0	2^{-3}	-2^{-7}	-2^{0}	2^{0}	2^{-7}	-2^{-3}	0	2^{-6}

Table 1: The value of coefficients in the synthesis and the analysis filters.

these filters suitable for video systems with medium bit rates. The relation between the low pass and high pass filters allows a polyphase implementation of these filters.

It is also important to notice that, because of the approximations, the reconstruction obtained by these filters is not perfect. However in all our experiments the signal to noise ratio of the reconstructed images has been more than $46dB$. In addition the reconstruction error in bright regions is larger than the error of reconstruction in darker regions. This property fits very well the human visual system, whose sensitivity is lower in bright areas.

The tree structure based on a two channel decomposition is a further simplification in implementation, since the same filters with the same coefficients are used at different levels.

The Table 1 gives the values of the coefficients in the analysis and the synthesis filters, as used in our implementation.

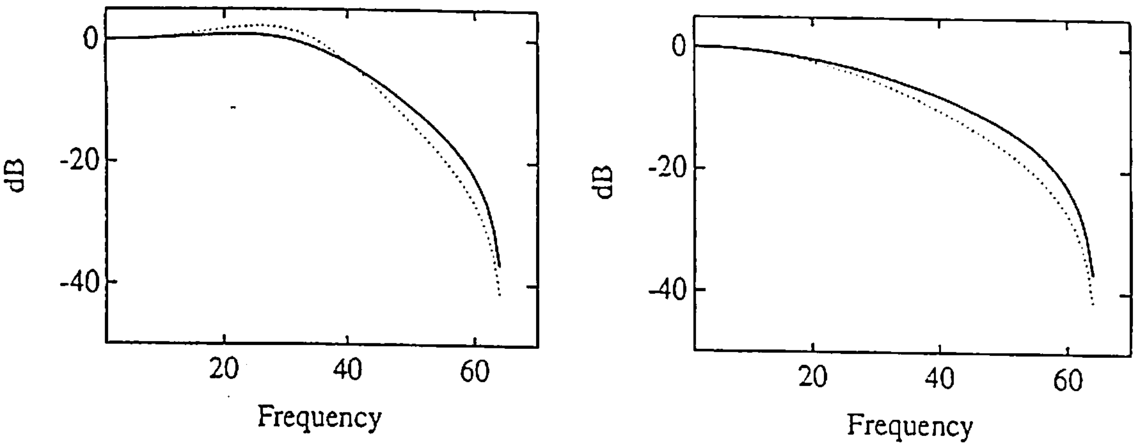


Figure 5: Filters characteristics in the frequency domain. The continuous lines are the filters with coefficients in power-of-two and the dotted lines are the ones with real values. left: analysis filters, right: synthesis filters.

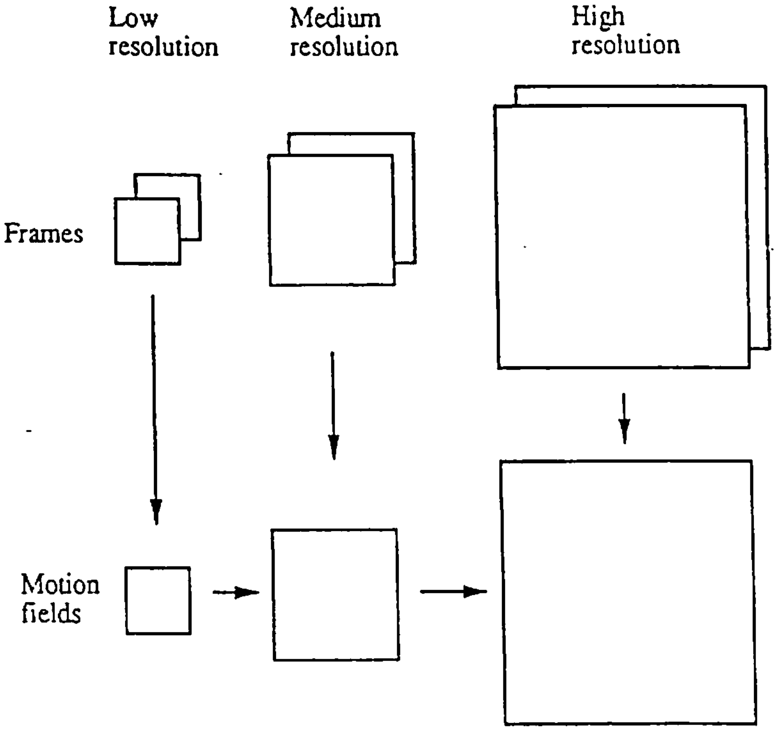


Figure 6: Hierarchical block matching technique.

4 Hierachical motion estimation

The motion estimation is performed by a hierarchical block matching, applied to the multiresolution data structure, as shown in figure 6.
The motion model assumed by block matching is an image composed of rigid objects in translational motion. A displacement vector is obtained by matching the information content of a rectangular measurement window, containing a certain number of pixels, with that of a corresponding measurement window within a search area, placed in the previous frame. Due to the high complexity of a full search algorithm, as well as to the limitation of the maximum motion vector allowed and the very noisy motion field that would result, we have adopted the following hierarchical block matching scheme. A multiresolution representation of images is generated by the Gabor-like wavelet transform. A first approximation of the motion field is obtained at the lowest resolution by a 2-step search. This motion field is recursively down projected to a higher resolution, used as initial conditions and refined, until the full resolution is reached. In order to avoid a wrong approximation propagating in the highest level, a median filter is applied to the motion field obtained at each level. The hierachical structure and the median filtering leads to a very robust and smooth motion field. Furthermore, this strategy has a very low complexity compared to the full search technique. The range of motion vectors is ±21 pels in each direction, with pixel accuracy. However, its improvement for higher accuracy is trivial.

5 Predictive and interpolative coding modes

The Gabor-like wavelet transform, as described in section 3, achieves spatial decorrelation. In order to reduce the temporal redundancies as well, an interframe coding technique is applied. As shown in Figure 7, predictive and interpolative modes are applied. In predictive mode, the current frame is predicted from

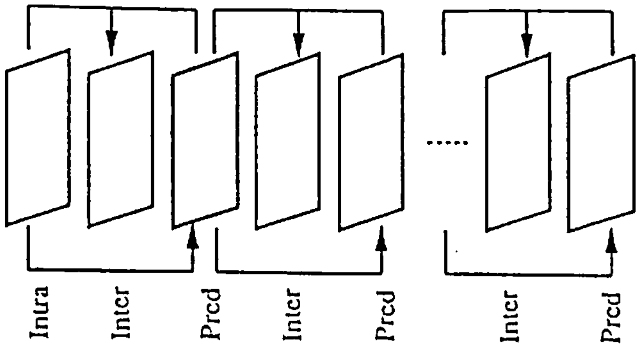


Figure 7: Structure of intraframe and interframe modes in a group of frames.

one previous frame and the motion field between these two frames, then only the error of this prediction is coded. In interpolative mode, the current frame is predicted from one previous and one next frame, using the motion field obtained in the predictive mode. In order to avoid the error propagation and also to be able to access randomly any frame in a short time, a fully intraframe mode is applied periodically. In the proposed system, one frame among each group of 11 frames is intraframe coded, and the remaining frames are interpolative and predictive coded alternatively.
In order to gain one frame memory and one frame delay, the sequence of interpolative and predictive frames is shuffled in a such a way that a predictive frame always precedes its corresponding interpolative frame. After decoding, the sequence is deshuffled to send the proper sequence to the screen. The deshuffling will add one frame delay at decoding, without any additional frame memory.

6 Quantization

The proposed system is not block based. Hence a scalar quantization with the uniform step for all the subbands is applied to the entire image. It is possible to introduce a frequency weighting of the quantization step size, using a weighting matrix, but it is not implemented in the current system. The same comment is valid for the dead-zones on quantizers.

7 Entropy coder

An adaptive arithmetic coder is applied at the output of the quantizer in order to generate the bitstream. The block diagram of this entropy coder is reported in Figure S. The adaptive model is a simple histogram generator. This histogram is used to provide the probability table for each symbol to be encoded. The histogram is updated after each incoming symbol, and rescaled when a maximum occurence is reached. The decoder is the dual of the coder part and has its own adaptive histogram. Therefore no side information needs to be transmitted for updating the model.

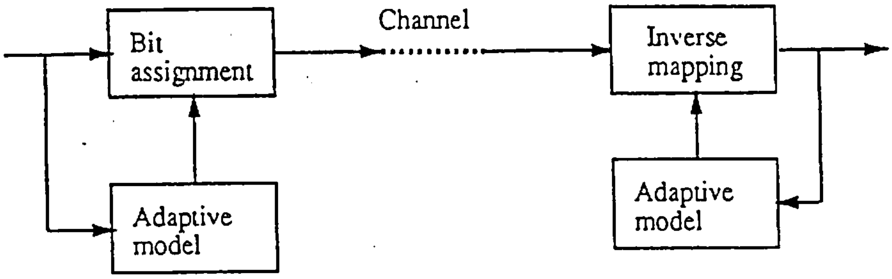


Figure 8: The adaptive arithmetic coder block diagram.

#40

Hardware evaluation of EPFL proposal for MPEG II

Jacques Kowalczuk, Regis Hervigo,
Touradj Ebrahimi, Marco Mattavelli,
Daniel Mlynek, and Murat Kunt

Laboratoire de Traitement des Signaux
Laboratoire d'Electronique Générale
EPFL-Ecublens (DE)
CH-1015 Lausanne, Switzerland

tel: +4121 693 2796
fax: +4121 693 4660
email: ebrahimi@ltssg1.epfl.ch

1 Deinterlacing

One line delay is necessary to compute the deinterlacing. One additional window delay of 8 pels allows the start of calculations. The area of 1 line delay of 720 pels is estimated to 3.5 mm^2 in a standard 2μ CMOS technology.

2 Subband decomposition

The goal is to filter in 2 dimensions (X and Y) an image in order to separate it into several subbands. This filtering is performed hierarchicaly four times. Of course, the first filtering which takes into account the whole image (576 × 720) is the most time and silicon consuming. That is why we will first focus on that part since the others are always less complex in term of transistor count.
The first stage of the 2D filtering is a group of two one dimensional filters (figure 1) [1]. A set of two filters for low pass and high pass filtering in a polyphase structure is used, as shown in figure 2 [2].
The filters are clocked at the rate of the incoming data. It is assumed that the error image is obtained when doing the difference between two images stored in 2 separated video RAMS (Size : 414 KBytes). The difference feeds the filters in flow. A dedicated control part has to be designed in order to provide the difference and to transfer the data to the filters. Every 74 ns a new pixel has to be processed in the filter bank. This means that the filters will function at a frequency of 13.5 Mhz. Several well known implementations show that this is a feasible solution. Since the size of the filters is fixed, the only variable parameter will be the number of the delay lines at each stage. This delay is equal to two lines for the first stage. At the second stage, the length of the delay lines will be the half of the previous one but the number of lines will still be the same. This is illustrated on figure 3. This particularity may lead to the implementation of the last stages on one chip.
Concerning the video RAM, It is possible to find actually on the market video RAMs with an access time equal to 80 ns. Even if the speed of the video RAM would be slower, it would be possible to transfer a certain amount of data into a multiport cache memory. Then a controler would feed the filters with the data at the necessary rate.

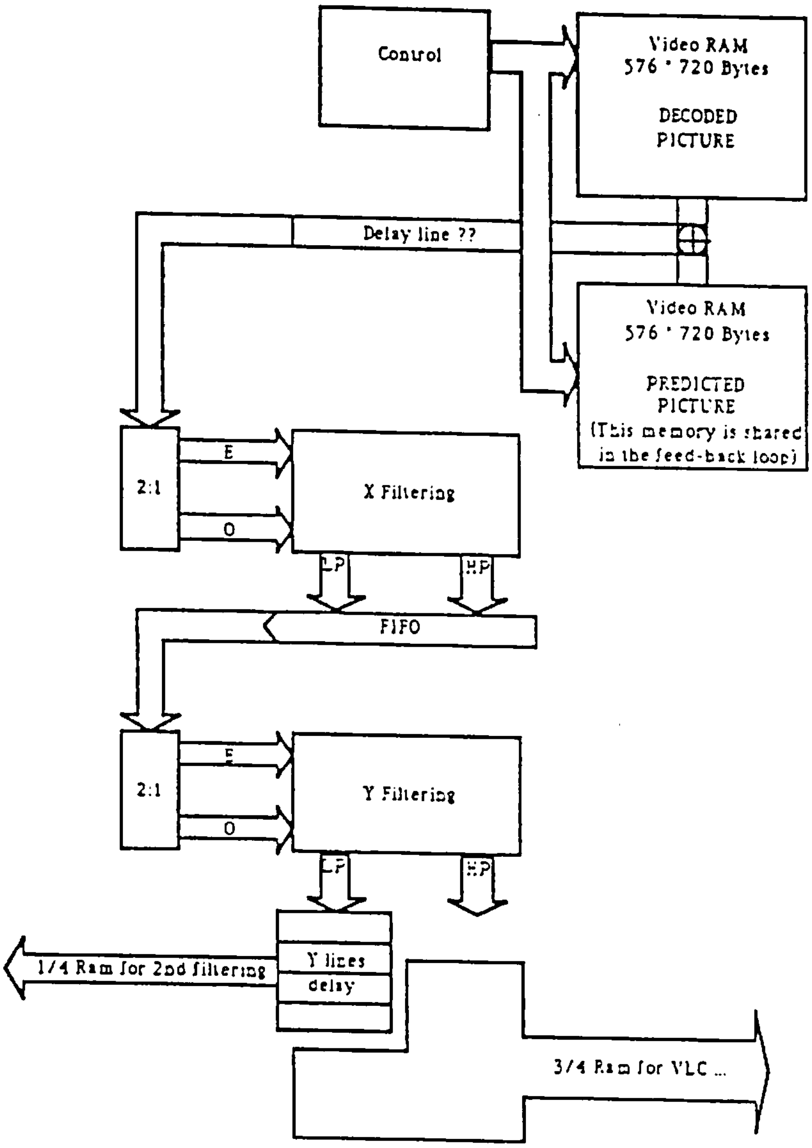


Figure 1: First stage of the hierarchical subband coding.

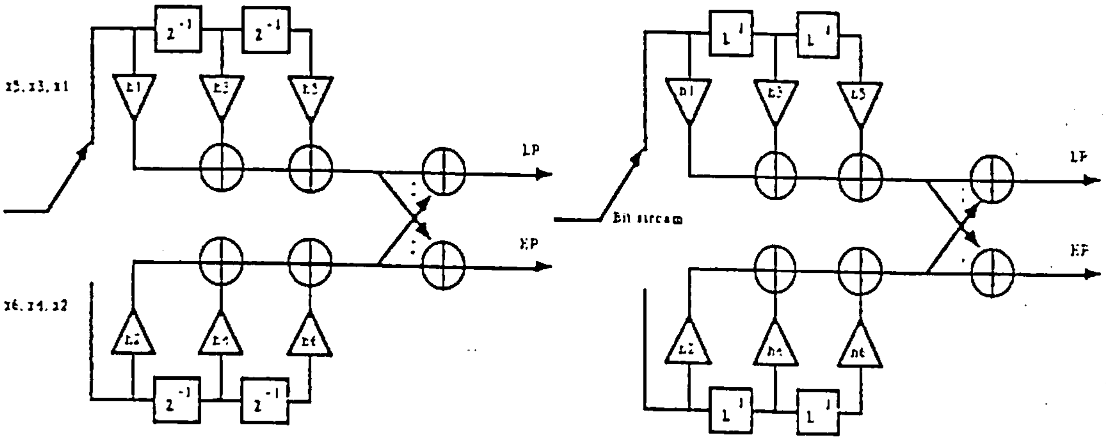
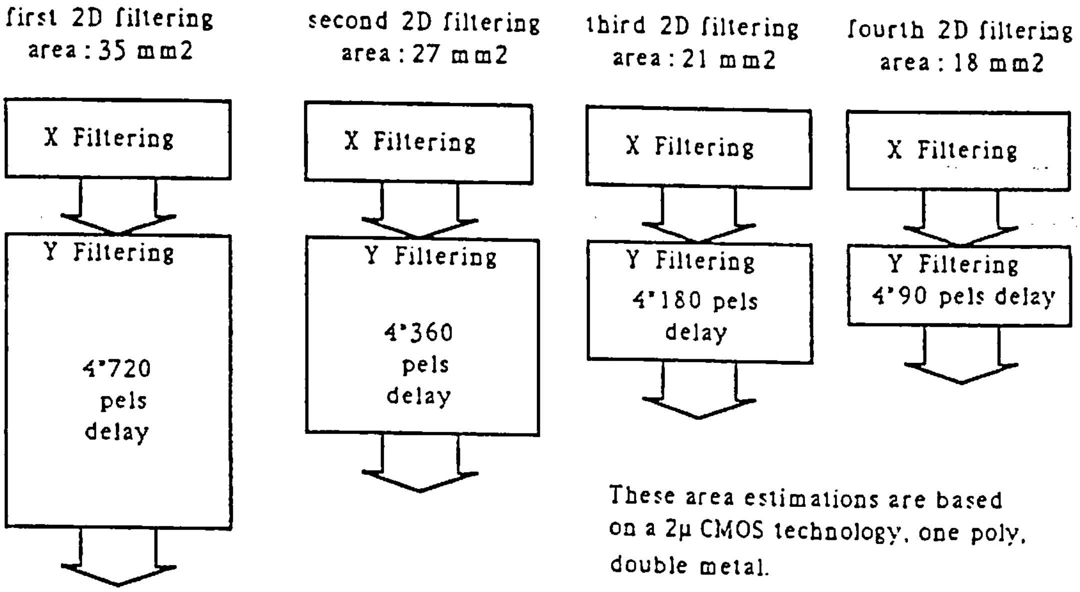


Figure 2: Polyphase filtering of low and high pass filtering.



These area estimations are based on a 2µ CMOS technology, one poly, double metal.

Figure 3: Hierarchical 2D subband coding.

2.1 Motion estimation

Our goal in this VLSI implementation is to provide 6500 motion vectors between two full resolution images at a distance of 80ms. The architecture that we propose is a set of 21 elementary processors which are connected in a tree structure (figure 4) [3]. In this way we could use a six step pipe-lined procedure. Moreover all processors connected to the same resolution level can work in parallel. The high level control of the tree structure is not totaly defined yet. We believe that a software control, realized for instance with a 68XXX microprocessor will be much easier to develop, compared to a dedicated controller. The elementary processor is divided into two parts which are similar and allow a two step pipe-lined processing of the two-step hierarchical block matching algorithm (figure 5). The operative part will be able to process in parallel the following expression:

$$\sum_{j=1}^{N}\sum_{i=1}^{N}|a_{ij}-b_{ij}| \tag{1}$$

Each processor will contain a multi-port RAM to speed up the access time to each pixel [4]. To reduce the size of the embedded RAM a 20 by 20 pels window focused on the reference pixel will be loaded. A set of nine pointers will be initialized on the first pixel of each search window. An incrementation of these registers leads to the following pixel. If we use a 6-step pipe-lined procedure, we can compute a set of 16 motion vectors in full resolution every 0.1ms. The processing of the lowest resolution level implies 405 operations. A first estimation of our system is:

Video RAM	674 Kbyte
Video RAM bandwidth	5 Mbyte/s
Embedded RAM	1.3 Kbyte
Embedded RAM bandwidth	50 Mbyte/s
Number of additions	34 adders / Pe 8 bit 200 Mhz
RAM area	50 mm^2
Adders area	14 mm^2
Routing area	15 mm^2

These area estimations are based on a 2μ CMOS process [5].

3 Quantization module

Uniform or non-uniform quantization can be implemented by a simple general purpose circuit. It consist of a multiplier, clocked with the input data rate, a small ROM to select a factor for uniform or non-uniform quantization and a logic circuit to round the result to the desired symbol size. The inverse quantization circuit differs only from the fact that no rounding of the data is performed at the output of the multiplier. The state of the art technology allows a straigthforward integration of this circuit for the working frequency of interest (13.5 Mhz). Being this module very simple and always placed after and before the direct and reverse wavelet subband transform an integration in the chips of those modules is suggested.

4 Entropy coder

The entropy coder can be divided in two main functional parts:

- a coder that performs aritmetic operations on the bitstream

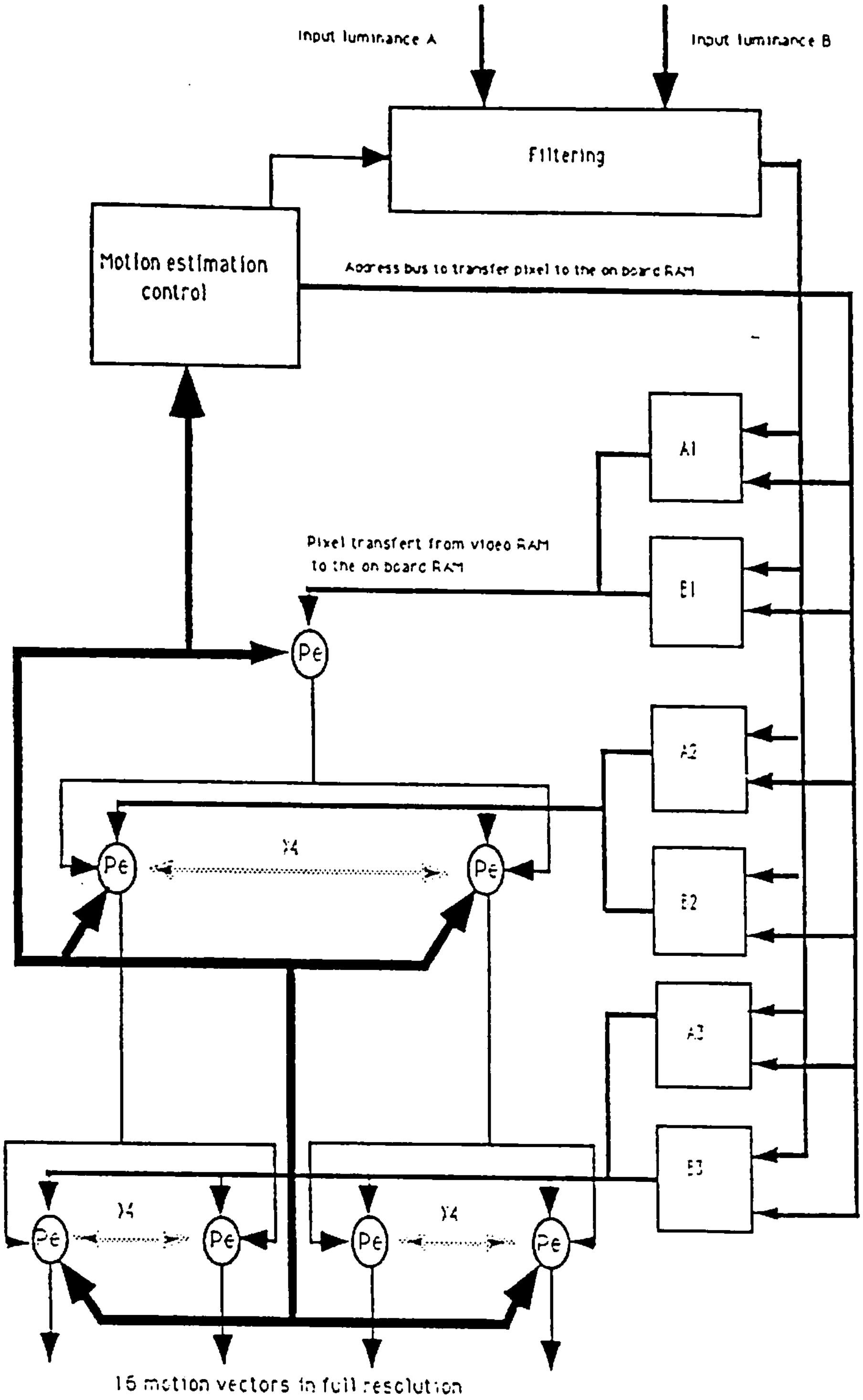

Input luminance A
Input luminance B
Filtering
Motion estimation control
Address bus to transfer pixel to the on board RAM
A1
Pixel transfert from video RAM to the on board RAM
E1
Pe
A2
X4
Pe
Pe
E2
A3
E3
Pe
X4
Pe
Pe
X4
Pe
16 motion vectors in full resolution

Figure 4: Overview of the motion estimation part.

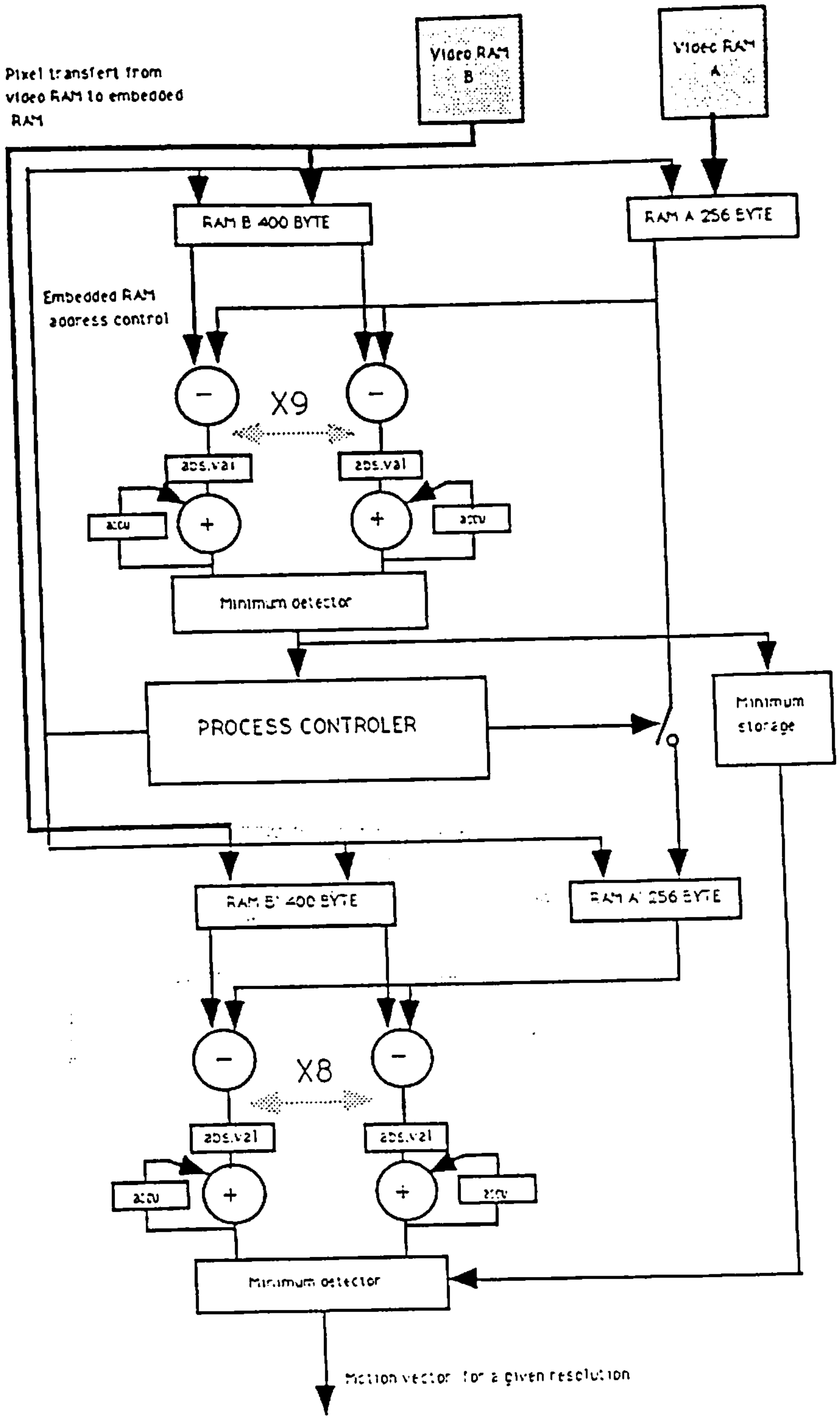


Figure 5: Elementary processor.

- an adaptive model that assigns to each codeword two numbers (the probability interval of the symbol) which feed the coder for the aritmetic operations. The model is adaptive because these two numbers depend not only on the incoming symbol but also on the running statistics of the data flow.

4.1 Aritmetic coder

In general the operation necessary to encode each input symbol are : two multiplications, two additions, two multiple shift operations, a control to avoid improper carry propagation (bit stuffing). Since some operations can be executed in parallel flows, the only speed constraint to consider has to be applied on the sequence of one addition one multiplication and a multiple shift operation for each input symbol clock interval (48 ns). Available technology for CMOS IC allows an implementation of the described sequence of operations, that satisfies the time constraint for the required aritmetic precision (16 × 16).

4.2 Adaptive model

The adaptive model module has to perform the following operations:

- collection of the statistics of the input data in real time.
- a probability interval for each input symbol assignment.
- encoding of the input data into a probability interval for the operation of the aritmetic coder

It requires :

- two RAM arrays, one for the tracking of the statistic (1024 × 8 bit), one for the assignement of the probability interval (1024 × 8 bit)
- an ALU processor that, each n clock perid , updates the probability interval table.

In conclusion the implementation of the aritmetic coder and the adaptive model in one entropy coder IC is certainly feasible with the current available CMOS technology.

5 Entropy decoder

The entropy decoder performs the same operations of the encoder but in reverse order. For the implementation of this module all the considerations done for the encoder remain valid.

6 Delay evaluation and synchronizations

The diagram of figure 6 gives the delay study as well as the timing in the encoder. In a first approximation, the delays less than a few lines are neglected. According to this diagram the encoder has a delay of 120 ms.

Similarly, figure 7 represents the diagram of synchronization of the decoder. In this diagram the delay due to the buffer which is equal to 40 ms (The buffer size in the system is equal to one frame memory) is taken into account. Hence, the overall coding/decoding delay is equal to 4 frames or 200 ms.

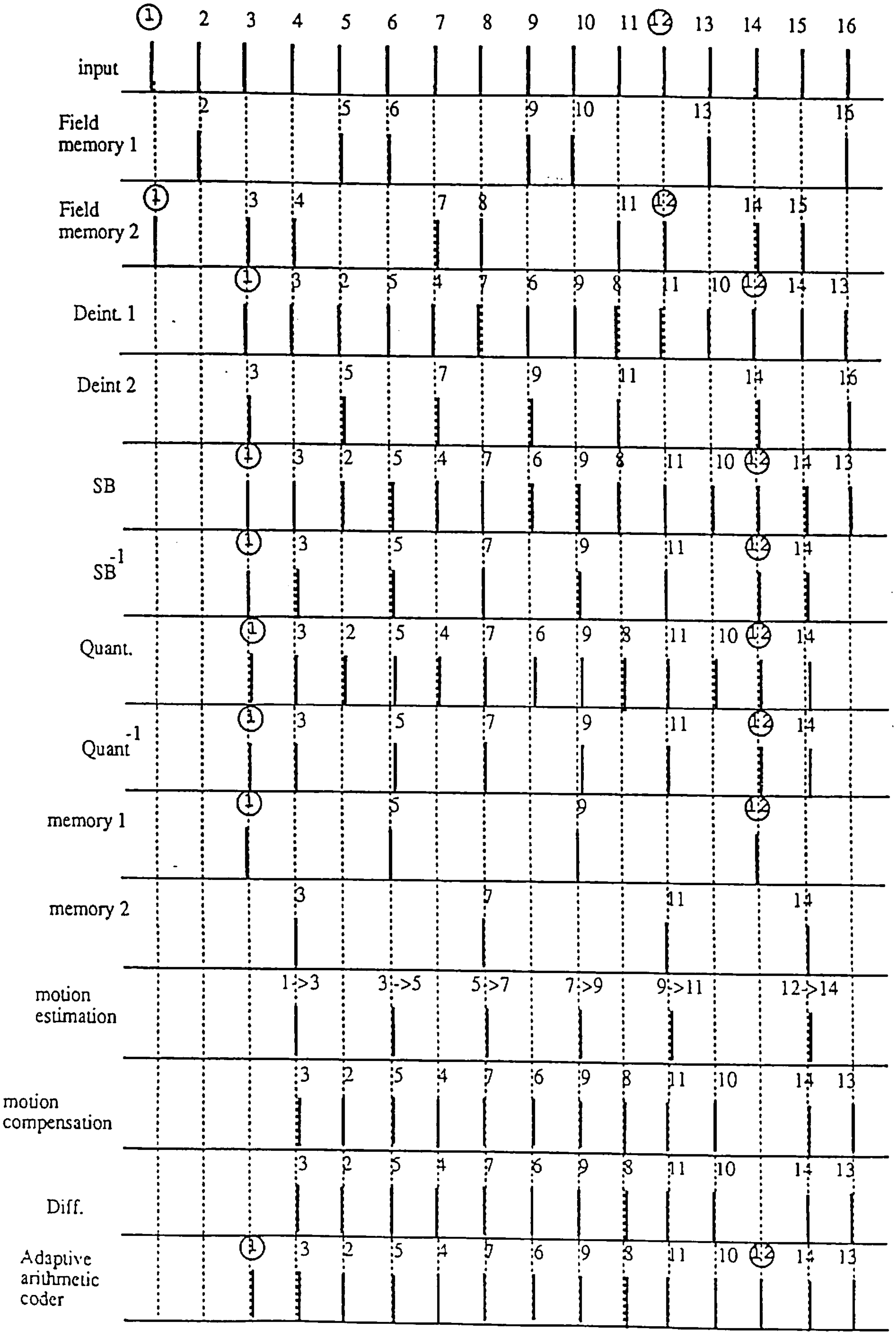


Figure 6: Synchronization diagram of the encoder

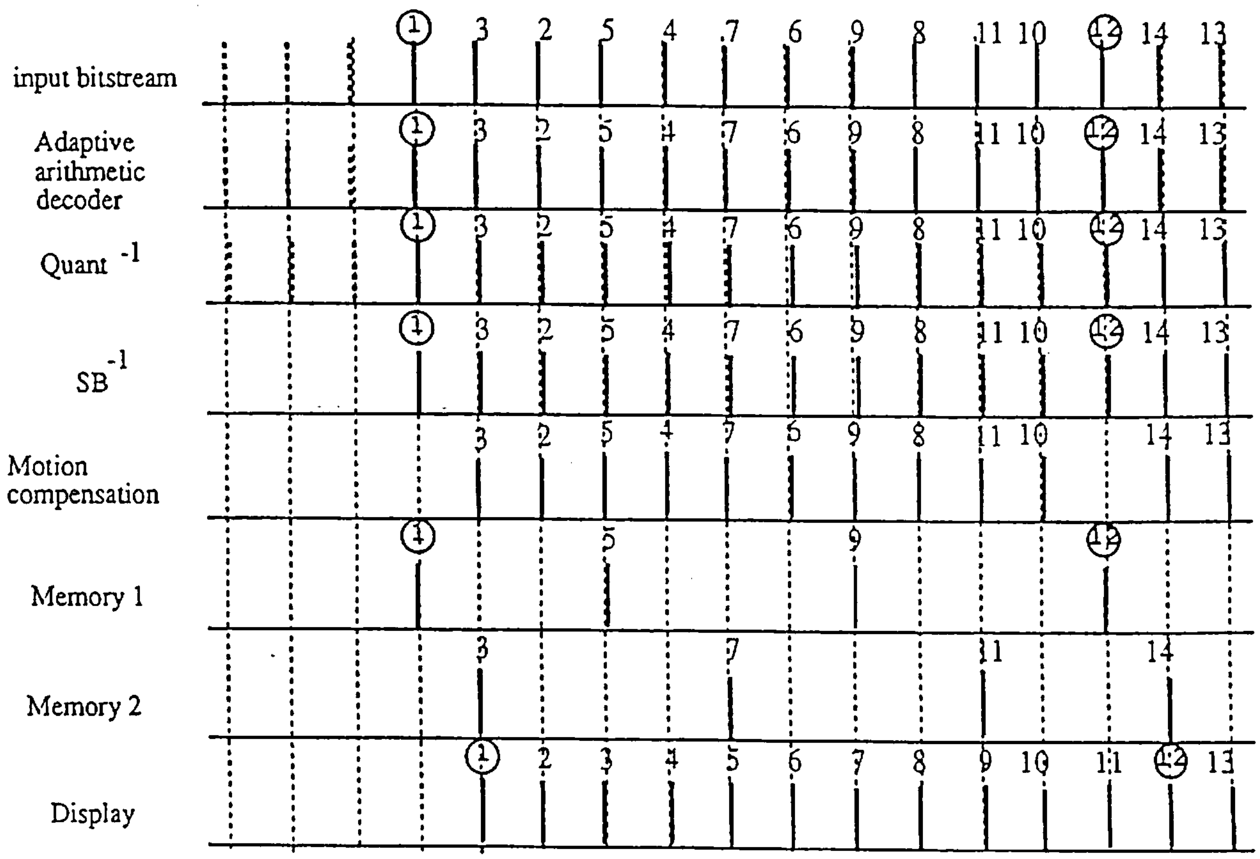


Figure 7: Synchronization diagram of the decoder

References

[1] T. Ebrahimi, T. R. Reed, and M. Kunt. Video coding using a pyramidal gabor expansion. In M. Kunt, editor, *Visual Communications and Image Processing '90*, volume 1360, pages 489–502, Lausanne, Switzerland, October 1-4 1990.

[2] M. Winzker, K. Gruger, and P. Pirsch. VLSI architecture of filter banks for an HDTV subband coder with 140 Mbit/s. In *Fourth international workshop on HDTV and beyond*, volume 1, Torino, Italy, September 1991.

[3] T. Komarek and P. Pirsch. VLSI architectures for hierarchical block matching algorithms. In *IFIP-Workshop*, volume 168-181, Grenoble, France, December 1989.

[4] H. Shinohara, N. Matsumoto, K. Fujimori, and S. Kato. A flexible multi-port RAM compiler for datapath. In *CICC'90*, volume 168-181, pages 16.5.1–16.5.4, Boston, 1990.

[5] J. Kowalczuk, S. Tudor, and D. Mlynek. A new architecture for an automatic generation of fast pipelined adders. In *European solid-state circuit conference*, volume 101-104, Milano, Italy, September 1991.

#40

ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE

PROPOSAL FOR MPEG2
TEST SEQUENCES GRAPHS

Here follow the graphs illustrating the performances of the system. The simulations were carried out on the test sequences according to the specifications given by MPEG2 Draft Proposal Description. For each test sequence we will show:

- the number of bytes for each component (Y, U, V and motion vectors)
- the total number of bytes for each frame
- the number of bytes transmitted every 0.4 seconds
- the signal-to-noise ratio for each field and each component

Flower Garden 9Mbs

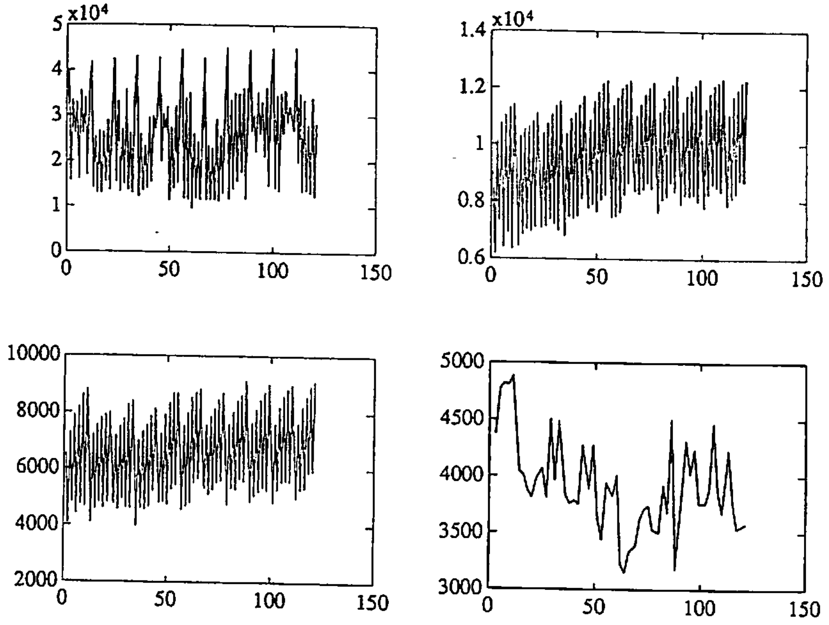


Figure 1: Upper left, upper right, bottom left: Y, U, V components respectively; bottom right: vector component

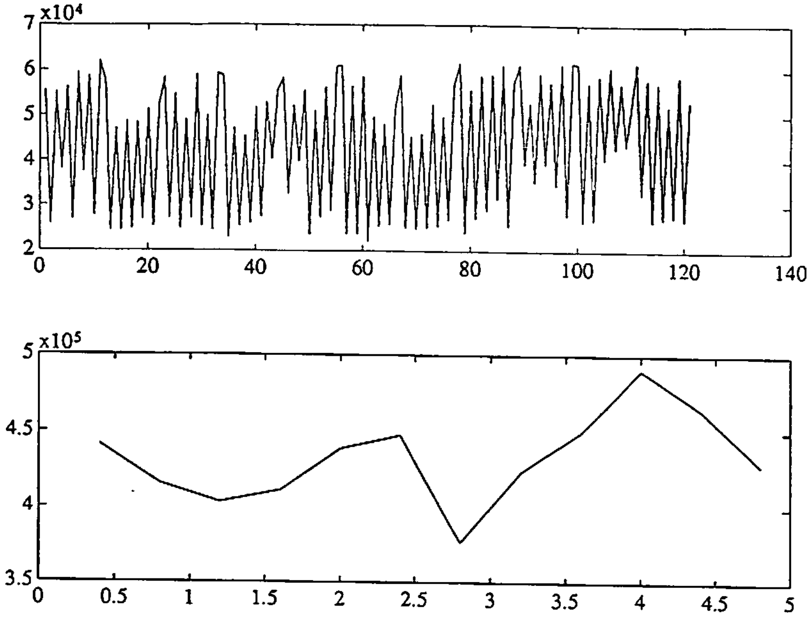


Figure 2: Upper: total number of bytes per frame; bottom: number of bytes every 0.4 seconds

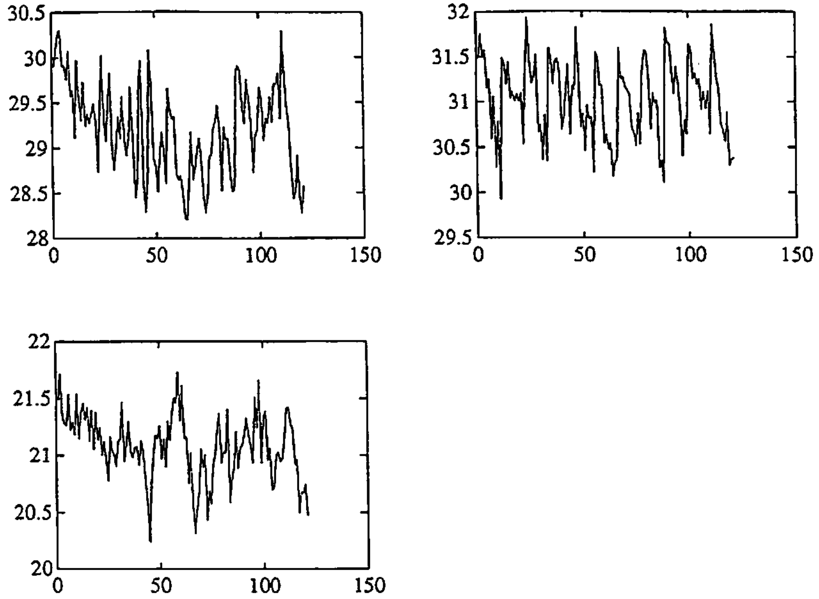


Figure 3: Upper left, upper right, bottom left: signal to noise ratio of the even fields, U, V, Y components respectively

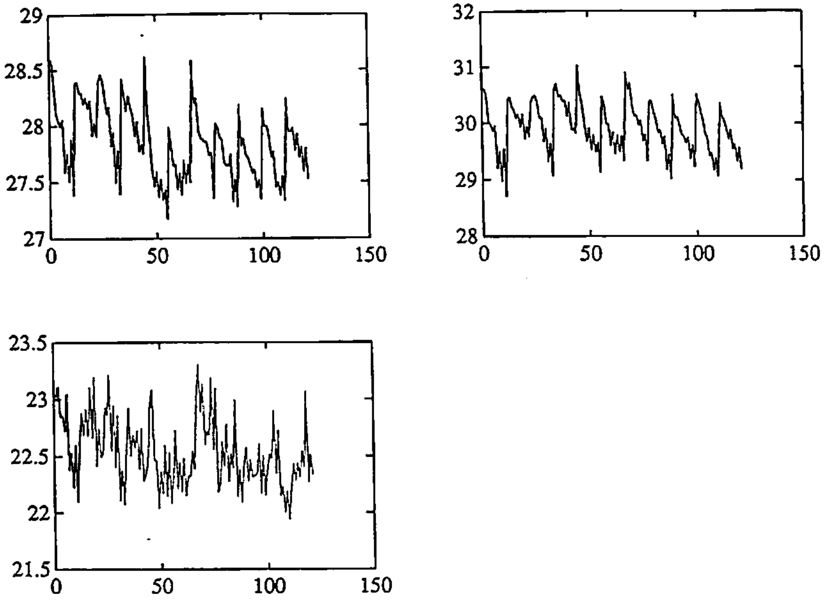


Figure 4: Upper left, upper right, bottom left: signal to noise ratio of the odd fields, U, V, Y components respectively

Mobile & Calendar 9Mbs

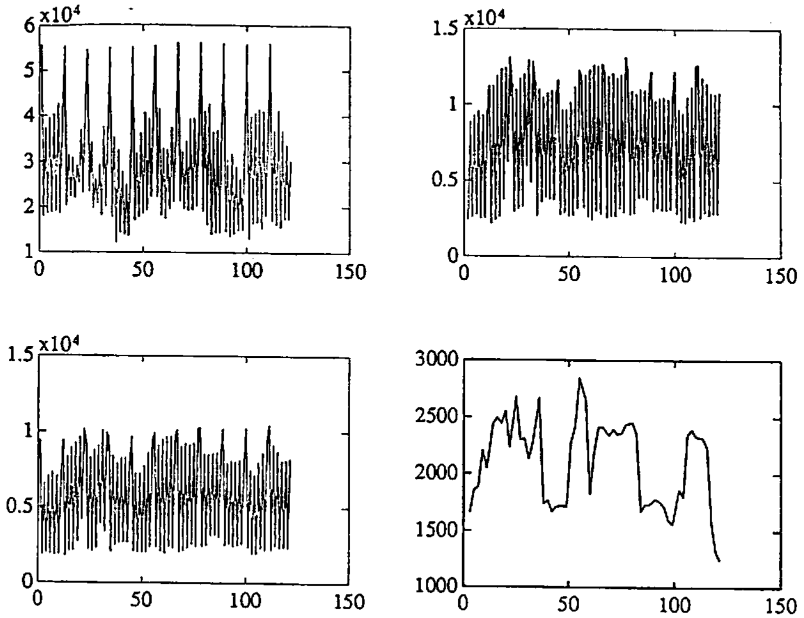


Figure 5: Upper left, upper right, bottom left: Y, U, V components respectively; bottom right: vector component

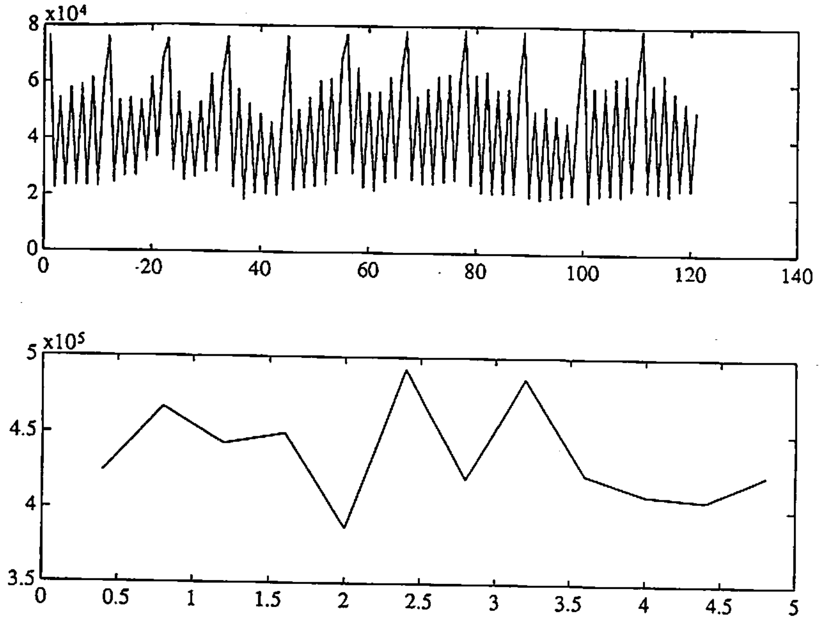


Figure 6: Upper: total number of bytes per frame; bottom: number of bytes every 0.4 seconds

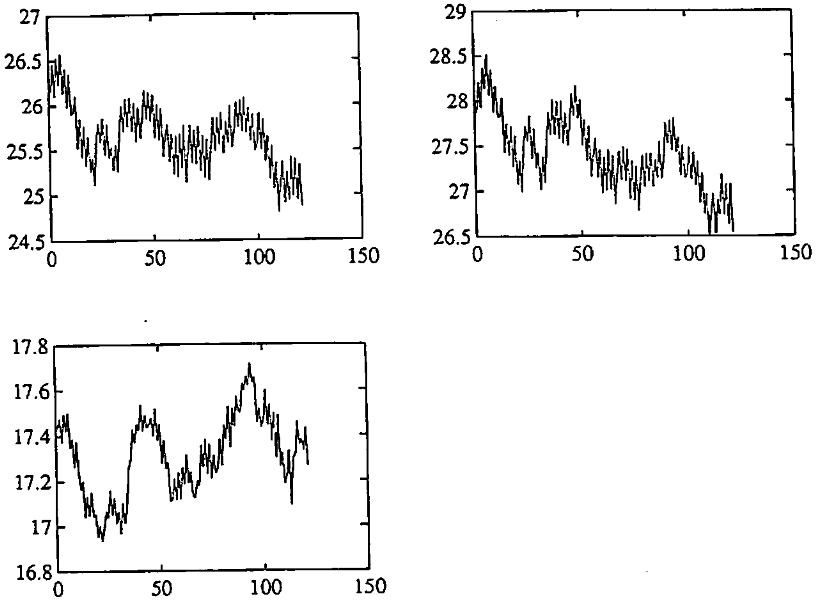


Figure 7: Upper left, upper right, bottom left: signal to noise ratio of the even fields, U, V, Y components respectively

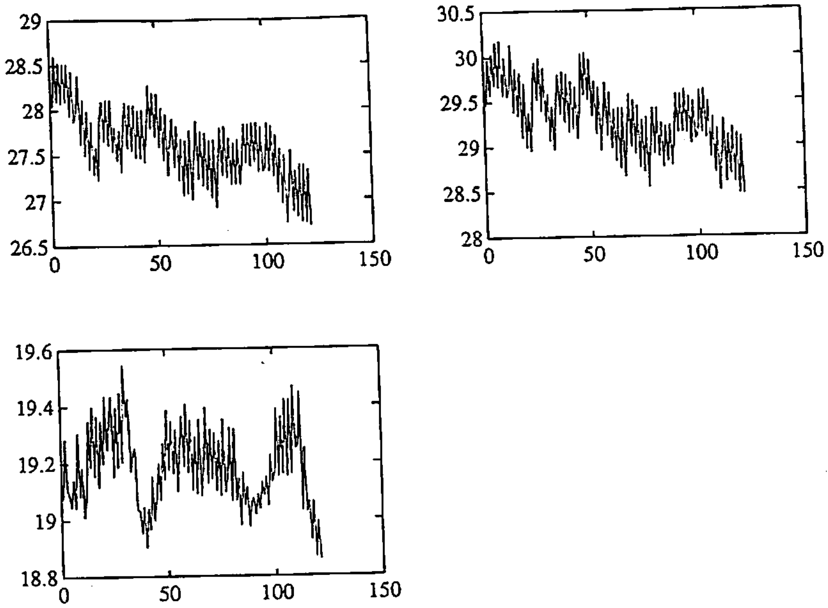


Figure 8: Upper left, upper right, bottom left: signal to noise ratio of the odd fields, U, V, Y components respectively

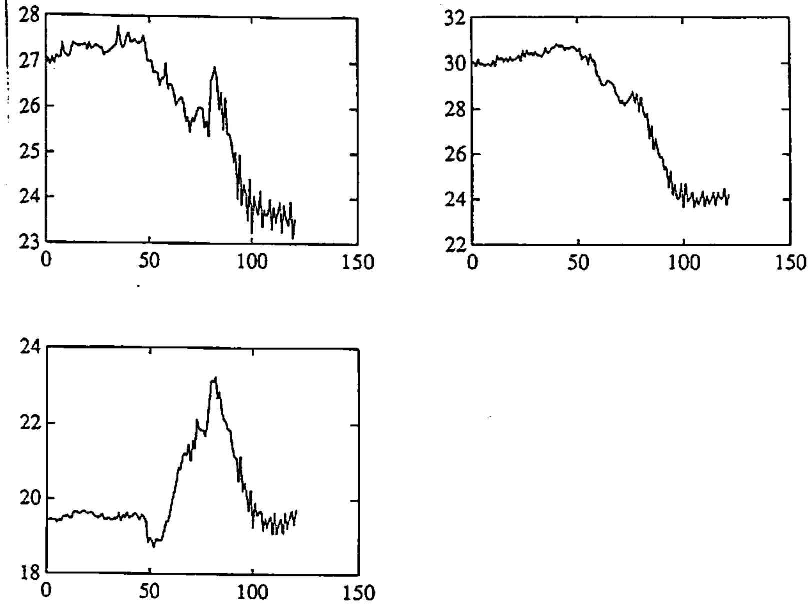


Figure 11: Upper left, upper right, bottom left: signal to noise ratio of the even fields, U, V, Y components respectively

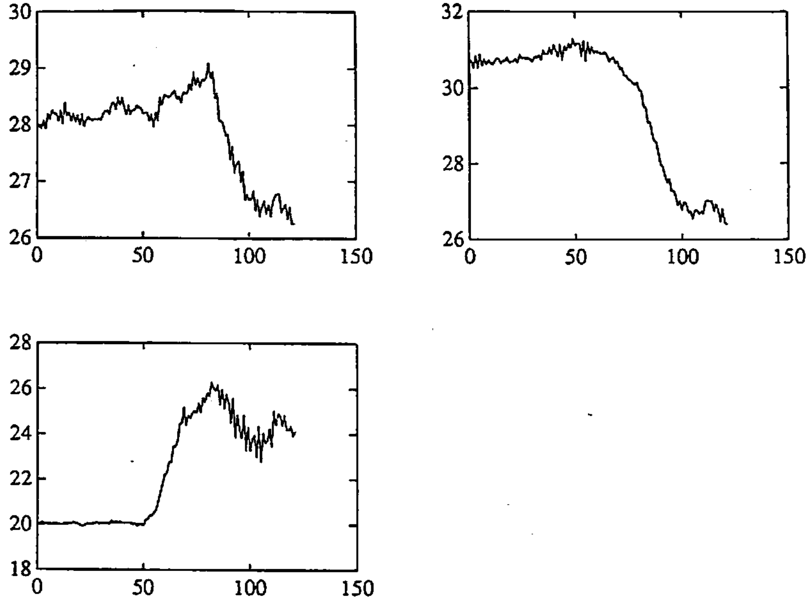


Figure 12: Upper left, upper right, bottom left: signal to noise ratio of the odd fields, U, V, Y components respectively

Popple 9Mbs

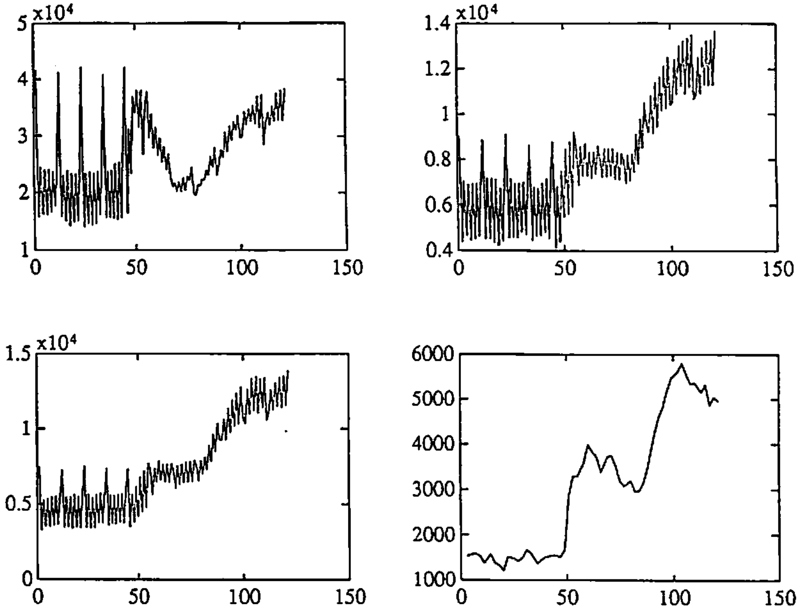


Figure 9: Upper left, upper right, bottom left: Y, U, V components respectively; bottom right: vector component

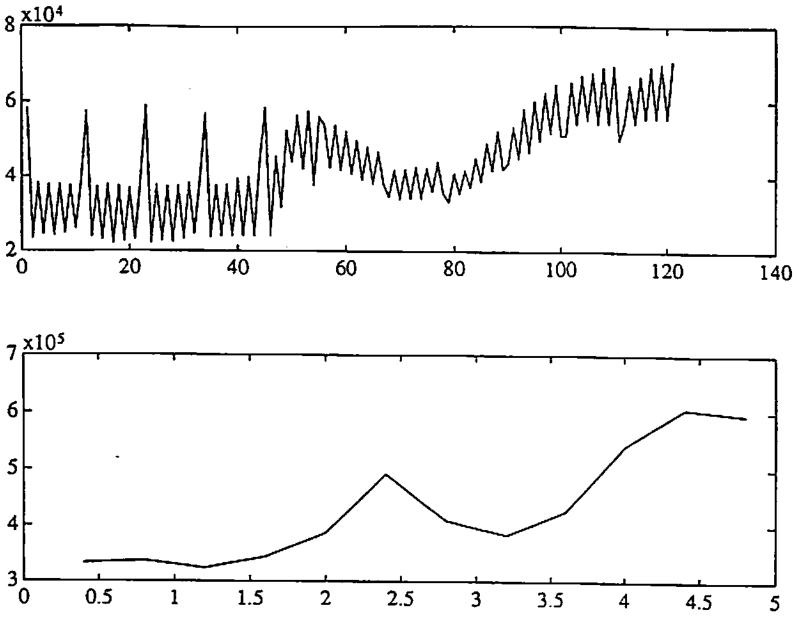


Figure 10: Upper: total number of bytes per frame; bottom: number of bytes every 0.4 seconds

Table Tennis 9Mbs

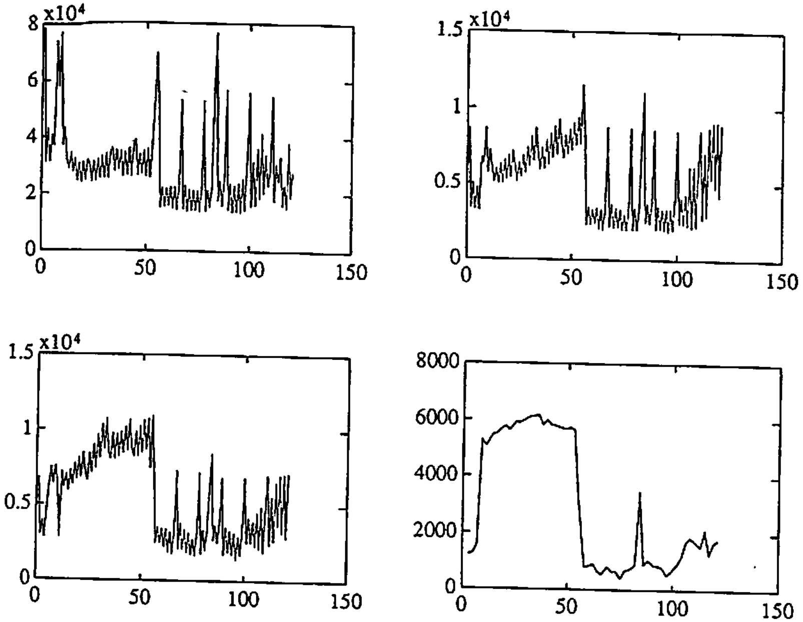


Figure 13: Upper left, upper right, bottom left: Y, U, V components respectively; bottom right: vector component

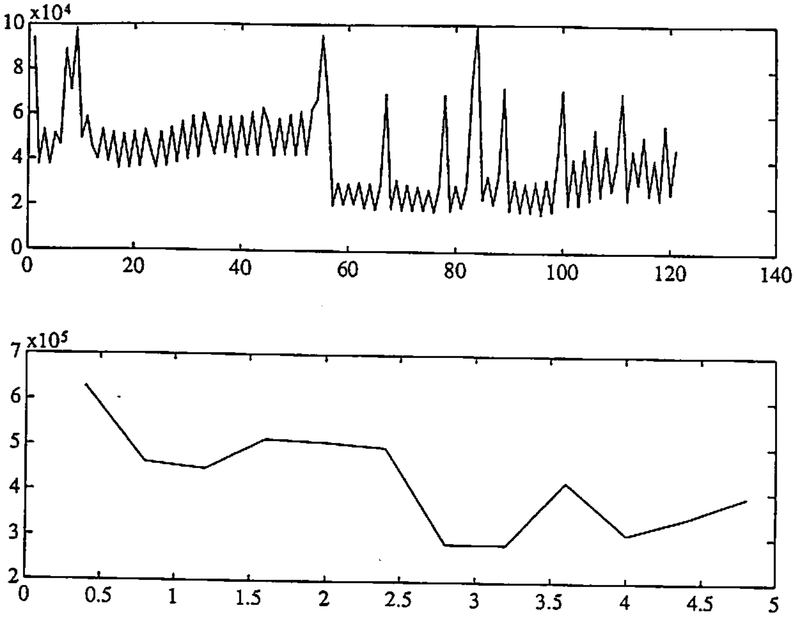


Figure 14: Upper: total number of bytes per frame; bottom: number of bytes every 0.4 seconds

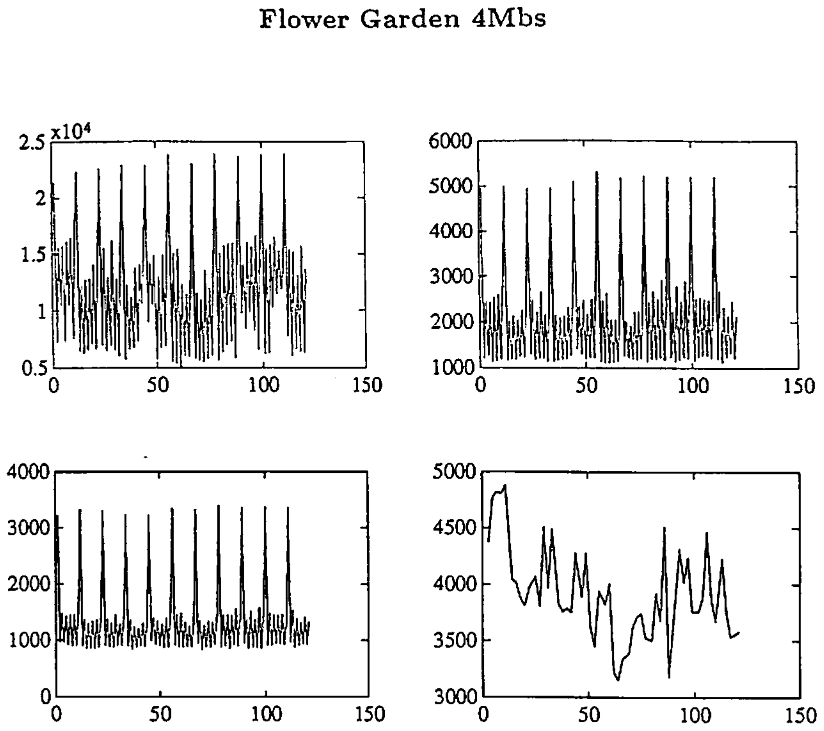


Figure 17: Upper left, upper right, bottom left: Y, U, V components respectively; bottom right: vector component

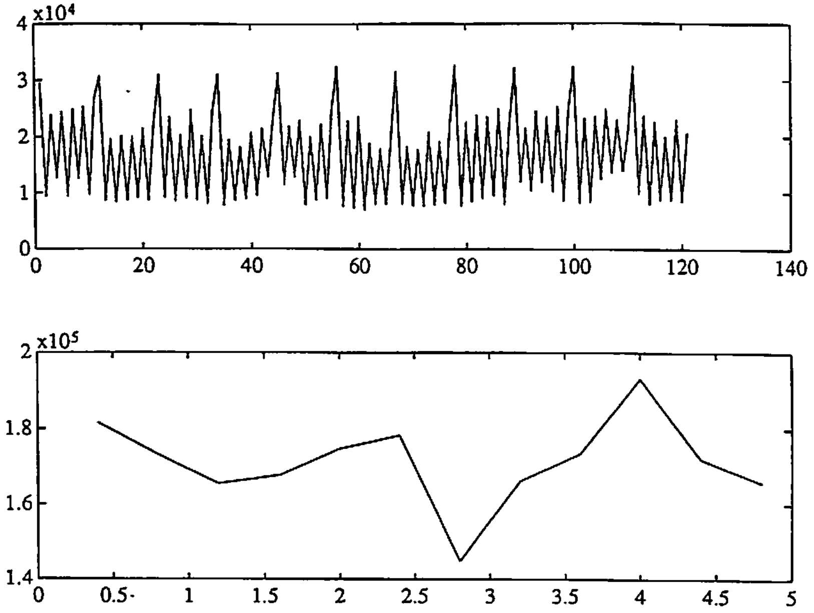


Figure 18: Upper: total number of bytes per frame; bottom: number of bytes every 0.4 seconds

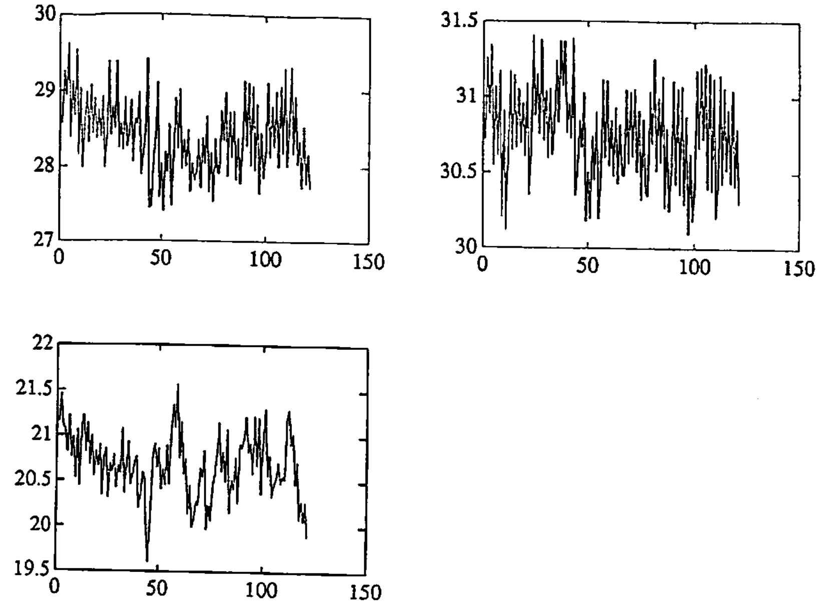


Figure 19: Upper left, upper right, bottom left: signal to noise ratio of the even fields, U, V, Y components respectively

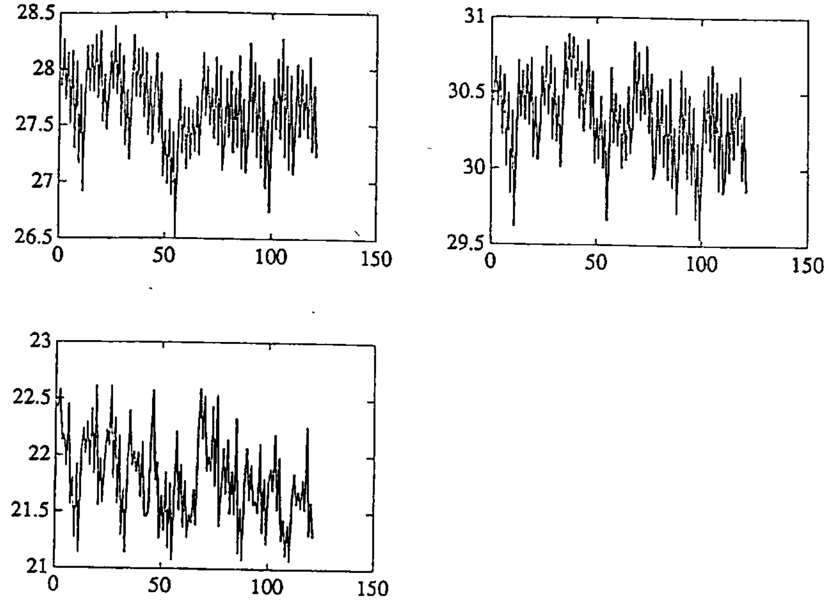


Figure 20: Upper left, upper right, bottom left: signal to noise ratio of the odd fields, U, V, Y components respectively

Mobile & Calendar 4Mbs

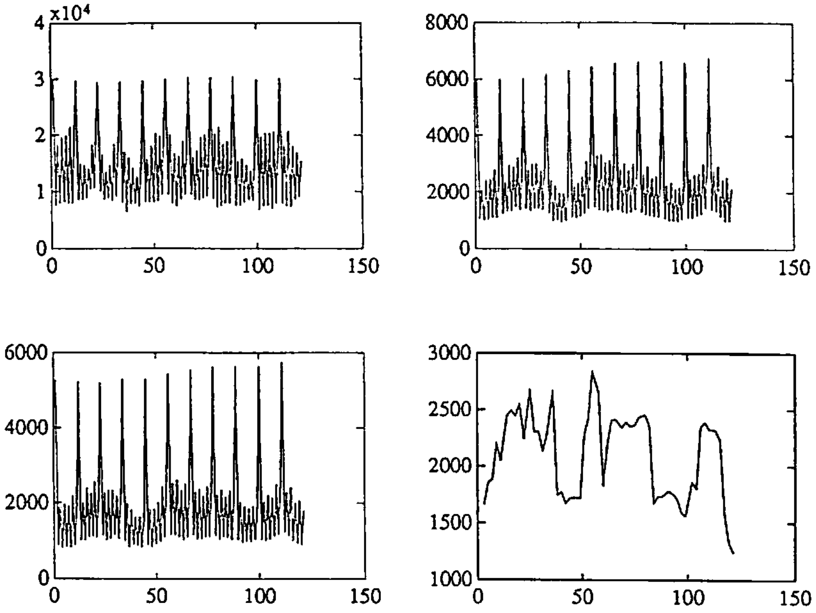


Figure 21: Upper left, upper right, bottom left: Y, U, V components respectively; bottom right: vector component

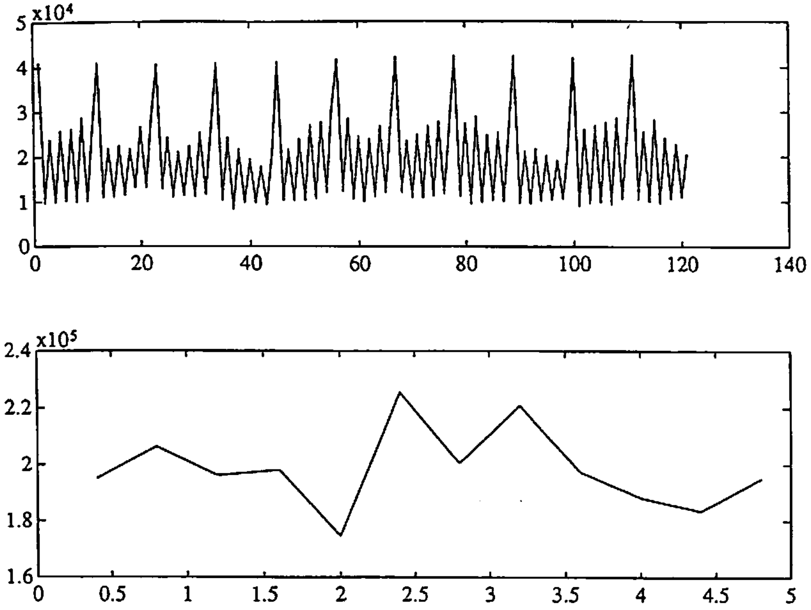


Figure 22: Upper: total number of bytes per frame; bottom: number of bytes every 0.4 seconds

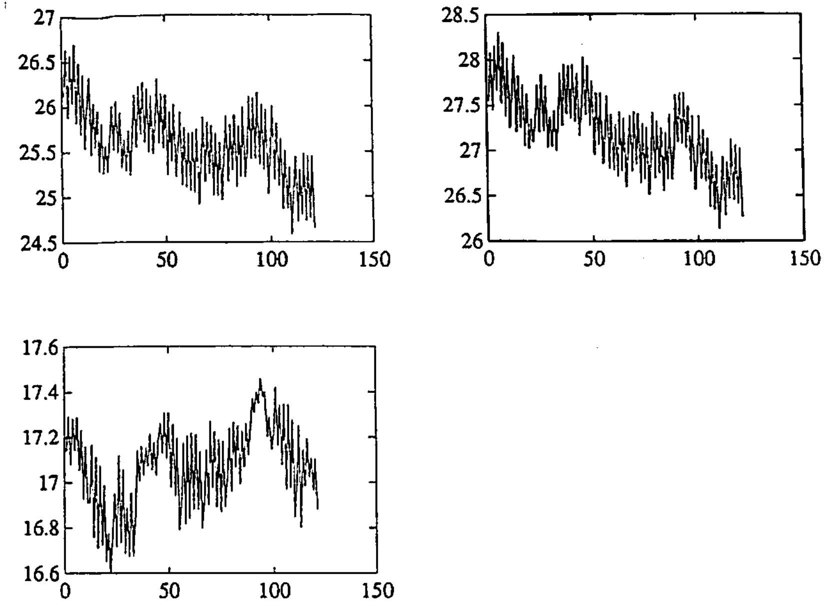


Figure 23: Upper left, upper right, bottom left: signal to noise ratio of the even fields, U, V, Y components respectively

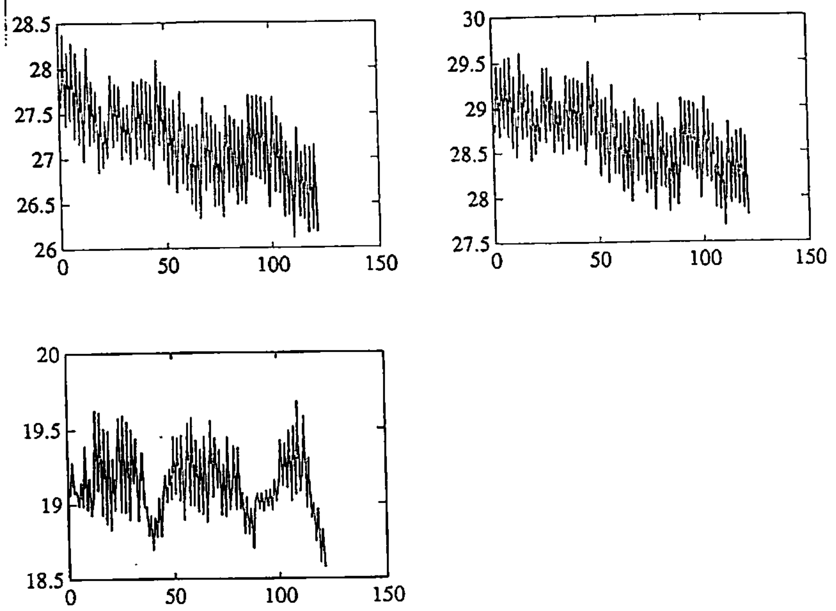


Figure 24: Upper left, upper right, bottom left: signal to noise ratio of the odd fields, U, V, Y components respectively

Table Tennis 4Mbs

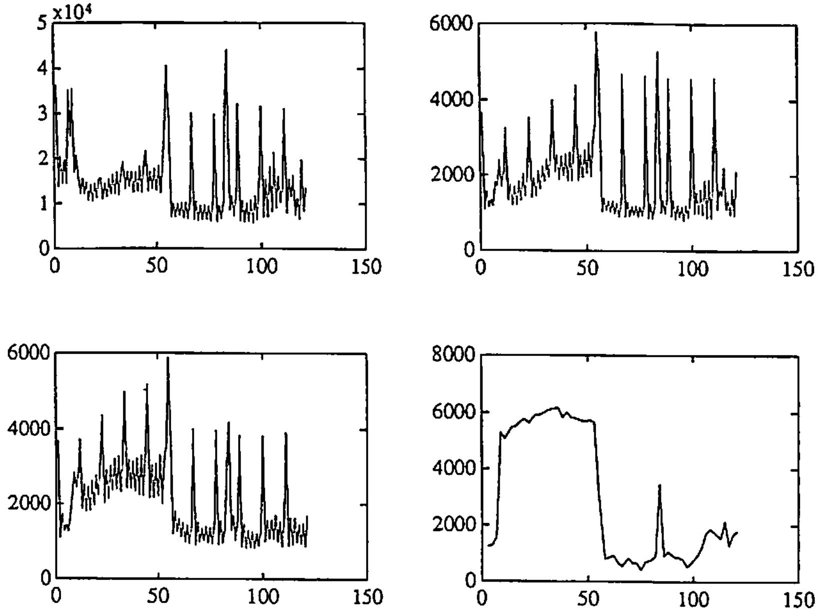


Figure 25: Upper left, upper right, bottom left: Y, U, V components respectively; bottom right: vector component

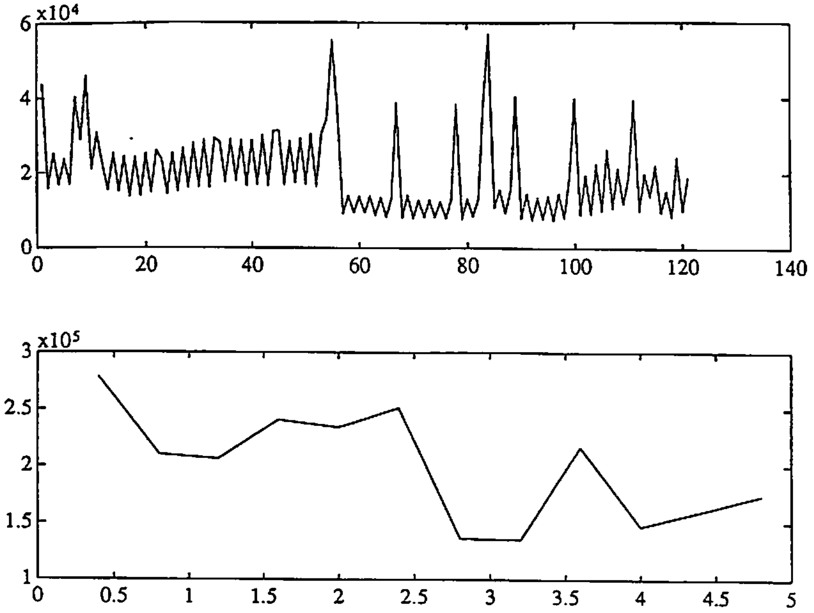


Figure 26: Upper: total number of bytes per frame; bottom: number of bytes every 0.4 seconds

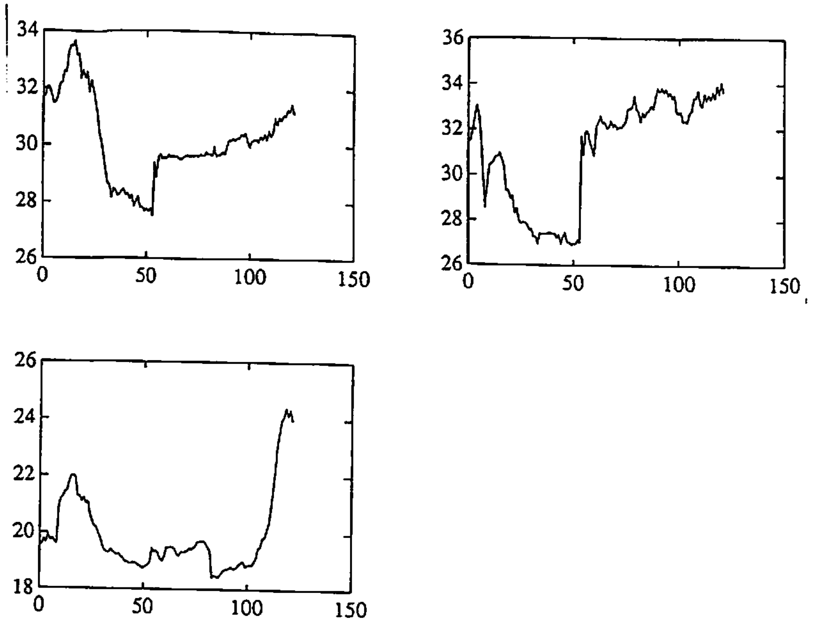


Figure 27: Upper left, upper right, bottom left: signal to noise ratio of the even fields, U, V, Y components respectively

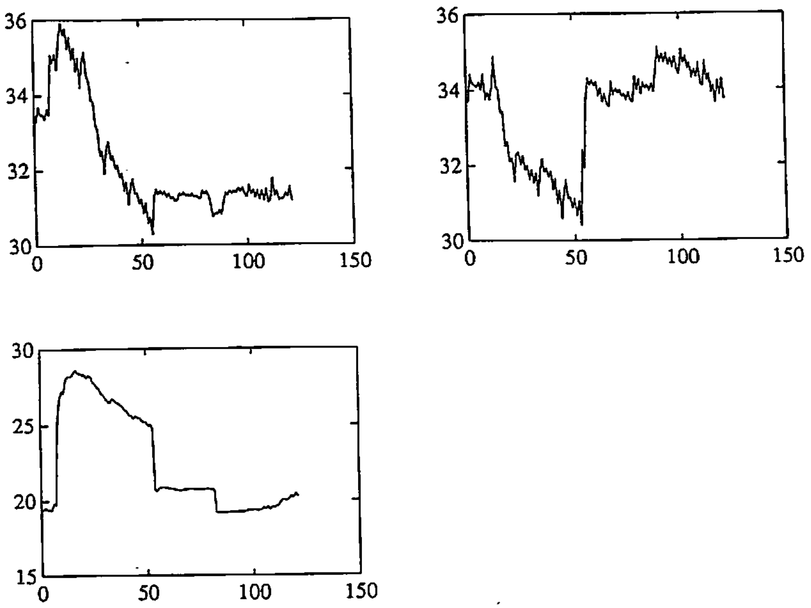


Figure 28: Upper left, upper right, bottom left: signal to noise ratio of the odd fields, U, V, Y components respectively

Multiplexing

The delimitation and the synchronization of each field within each frame and of each frame within each sequence is carried out by means of the following multiplexing technique. A flag is used as a separator so that the lenght of each frame has no constraints; the control fields are in fixed position. The transparency is obtained by means of the so called *bit stuffing technique.*
The overhead information for the coded test sequences are shown in the following table.

Test Sequence	*Overhead Percentuage*
Flower Garden 9Mbps	0.75%
Mobile & Calendar 9Mbps	0.25%
Popple 9Mbps	0.61%
Table Tennis 9Mbps	0.44%
Flower Garden 4Mbps	0.096%
Mobile & Calendar 4Mbps	0.87%
Table Tennis 4Mbps	1.24%