ISO-IEC/JTC1/SC29/WG11 MPEG92/285
CODING OF MOVING PICTURES AND ASSOCIATED AUDIO

CCITT SGXV
Working Party XV/1
Experts Group for ATM Video Coding

Document AVC-301
July, 1992

SOURCE: G. Bjøntegaard, Norwegian Telecom Research.
On behalf of the Eureka VADIS project.
TITLE : Flexible encoder defined predictions.
PURPOSE: Proposal

1. Abstract.

The purpose of the present document is to describe a method whereby the encoder is able to specify and down load predictors. The emphasis of the document is therefore to describe how this definition may be made rather than defining a specific set of predictors. Some of the essential parts of the method are:

- The predictors are not fixed in the standard but may be optimised for each application.
- The method gives room for almost all the known predictors like frame based, field based and FAMC.
- In addition it gives room for defining new predictions also after the standard is fixed.
- The predictors may be down loaded on the sequence level (or picture level).
- Only one vector is used for each macro block for P-frames/fields.
- The implementation complexity is comparable with "traditional" 1/2 pixel motion prediction.

2. Introduction.

When coding image sequences the quality of the predictor is of great importance for the performance of the total coding scheme. For this reason many prediction modes have now been brought forward in the MPEG work. At the last MPEG meeting in Haifa many prediction modes were defined in the test model for consideration concerning coding performance.

On the other hand there is considerable concern from the implementation side that too many prediction modes, too many motion vectors, etc. are introduced and thereby complicating the hardware.

In this document a unified solution for prediction is defined that may cover most of the prediction modes defined in TM1. In addition it gives room for other predictions to be developed in the future.

3. Basic characteristics of the "Down loadable Motion Prediction".

3.1. P-frames/B-frames.

The method is primarily intended to be used for P-frames/fields. One reason for this is that the predictions of P-frames/fields have so far been less efficient than for B-frames. This is therefore an attempt to improve P-frame prediction and at the same time keep the possibility for simple implementation.

If the method should be used for B-frames it would have to be developed specially for that purpose and tested accordingly. At the present this work has not been done and the method is proposed for P-frames/fields only for the time being.

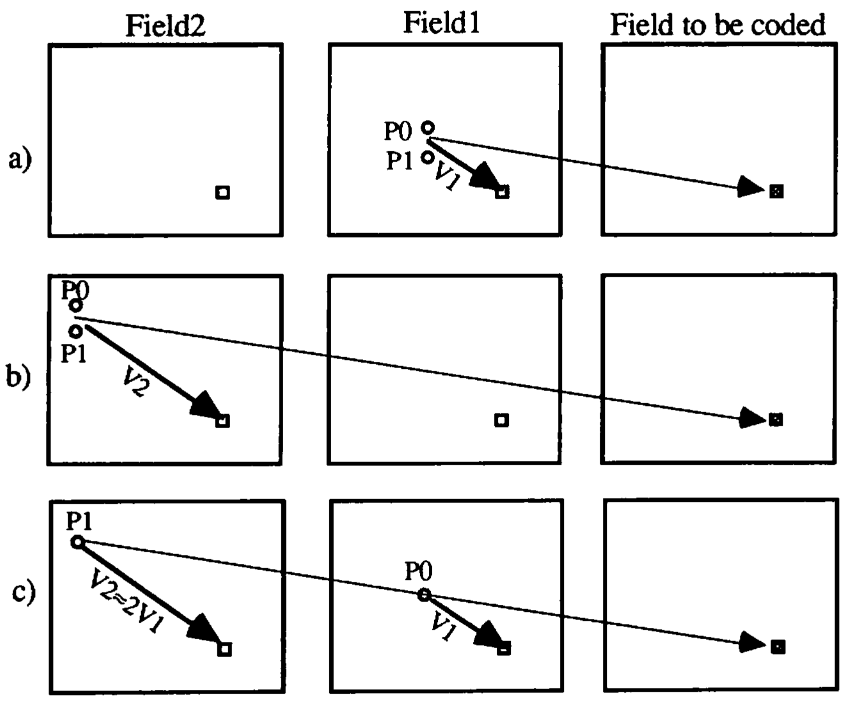
3.2. One vector to define the prediction for each macro block.

In TM1 the following prediction modes are defined for a frame macro block:
- Frame prediction.
- FAMC.
- Field prediction.
- Dual field prediction.

The two first modes use one vector, the third two vectors and the fourth 4 vectors for prediction. With the present method only one prediction mode and one motion vector are used for both frame and field macro blocks.

3.3. Flexibility of the prediction.

The predictions in MPEG have been based on 1/2 pixel resolution motion compensation. This has turned out to be efficient, but for interlaced picture material it is sometimes preferable to use field based prediction, frame based or a combination (FAMC).



■ Pixel to be predicted
□ Location of pixel to be predicted in previous fields
o Point to be used for prediction
$\longrightarrow$ Indication of vector size
$\longrightarrow$ "Trajectory" of prediction

Figure 1. Indication on how the prediction may be made from two points in the last two decoded fields. The predictors in a) and b) are "field based" whereas the predictor in c) is "FAMC like". All the prediction in a), b) and c) are suitable for the same physical motion - when the motion relative to field 2 is twice as large as the motion relative to field 1 ($V2 \approx 2V1$).

In the present method the definition of each vector may be defined by the encoder and is down loaded to the decoder at the sequence or picture level. Figure 1 indicates how the method may be used to define the prediction modes used in TM1. The following points should be noted concerning flexibility of the prediction method:

- The prediction is made from a maximum of four pixels in the previous two fields.
- The four pixels are arranged in two pairs (P0 and P1). P0 and P1 may be considered as two points with 1/2 pixel horizontal accuracy and frame line vertical accuracy (see figure 1).
- The weights of the four points are limited to: (0,1/8,1/4,3/8,1/2,3/4, 1). The sum of the weights is equal to 1.
- There is flexibility in the method to allow both field prediction and "FAMC like" prediction that I will call FAMC′.
- If P0 and P1 in figure 1c have the same horizontal position and a difference of one line in vertical position the prediction is "frame based".
- The defined predictors does not have to be equally spaced concerning the physical motion they represent. It is usually not necessary to have the same accuracy for large motion as for small since the picture is blurred due to motion anyway.
- For the same reason it is not the numbering range that sets limitation to the physical motion to be covered. This is limited by the maximum vector range that has to be defined in the standard for hardware reasons
- It is possible to define a set of predictors that is a mixture of field prediction and FAMC′ (see section 6). It has turned out to be beneficial to use FAMC′ for small vectors and field prediction for large vectors.

3.4. Separability in horizontal and vertical directions.

Separable coding of the horizontal and vertical components of the motion vectors is made. Likewise, the definitions of the predictors are made "almost" separable (see section 4). One implication of this is a large saving in bits needed to define a complete set of predictions.

3.5. "Generic prediction".

The standard resulting from the MPEG work shall be "generic" in the sense that it shall be applicable for many different uses. For this reason it is important that the prediction - which is recognised to be very important for coding efficiency - can be adapted to different situations and different picture material. This is possible with the present method as predictors may be optimised and down loaded for each situation.

4. Definition of the predictors.

The addressing of lines is assumed to be frame based. In the definition of predictors we are only interested in address differences. A line difference of an even number therefore points to the field of the same parity. Likewise a line difference of an odd number points to the field of the opposite parity.

Referring to figure 1 the prediction is assumed to be made from two locations with half pixel resolution horizontally and with frame line resolution vertically. The two "points" are called P0 and P1.

To use the present method of predictor definitions it is necessary to have well defined syntax for this definition. The following points should be considered:

- The definition should be flexible enough to allow prediction improvements both now and in the future.
- The definition must be simple from a hardware point of view.
- The down loading must not spend too many bits.

Separate predictions for even and odd fields.

For frame coding of interlaced pictures the prediction distance in time is different for the even and odd fields. Therefore we need different predictors for the two fields (denoted "even" and "odd"). In this section I will only deal with prediction of the even field for simplicity. The total definitions are given in Annexes 1 and 2.

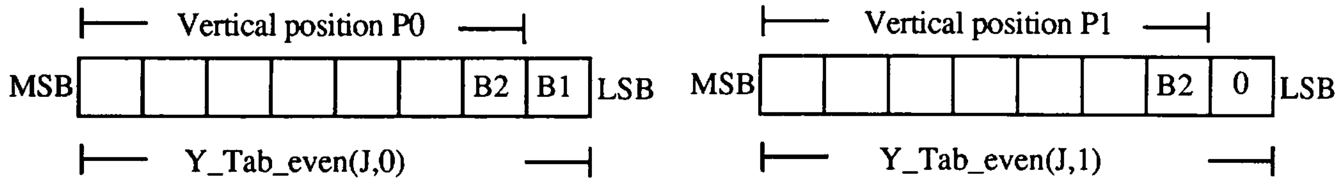
Definitions in the vertical direction.

Call the index for the vertical position J. For each J we need the vertical deflections for points P0 and P1 from figure 1. The values are contained in the matrix:

Y_Tab_even(J,k), k=0,1 to represent P0 and P1.

The figure below shows how an element of the matrix contains both the vertical positions. For k=0 the LSB contains one bit (B1) determining the relative weight to P0 and P1. For B1=0 equal weight is put on P0 and P1. For B1=1 the relative weights are 3 and 1. (If we instead had wanted relative weights 1 and 3 points P0 and P1 could have been interchanged.)

The B2 bit in the figure tells if the vertical deflection is even or odd. B2 therefore tells if the point comes from the field of same or opposite parity. This determines which if the horizontal components are used for prediction.



Definitions in the horizontal direction.

Call the index for the horizontal position I. For each I we need to define one deflection to be used if prediction is taken from the field of same parity and one deflection to be used if prediction is taken from the field of opposite parity. The values are contained in the matrix:

X_Tab_even(I,k), k=0,1 to represent fields of same and opposite parity.

More details on how the matrices are used are found in Annexes 1 and 2.

Annex 1. Syntax for down loading of prediction matrices.

The full definition of the predictions are contained in the following matrices:

X_Tab_even(I,k)	Array of horizontal deflections with 1/2 pixel resolution for even field. I number of predictor. k=0 for prediction from field of same parity. k=1 for prediction from field of opposite parity.
X_Tab_odd(I,k)	Same for odd field.
Y_Tab_even(J,k)	Array of vertical deflections with frame line resolution for even field. J: number of predictor. k=0 for the first vertical position to be used for prediction. k=1 for the second vertical position to be used for prediction.
Y_Tab_odd(J,k)	Same for odd field.

Absolute values.
The indices I and J run from 0 to the maximum value (see below). For negative indices the horizontal and vertical deflections are assumed to be the anti symmetric values.

Even/odd:
It is assumed that we have separate definitions to predict even and odd fields. If pure field coding is used, the "even" definition is used. The "odd" matrices are therefore not needed. Whether we have frame or pure field coding is given by the parameter Frame_Field_Pred.

Prediction range.
The number of predictors horizontally and vertically is defined by Pred_range. The number of predictors is: 8•2**Pred_range. (Pred_range has the same meaning as forward_f_code defined in the picture layer).

Syntax.

Pred_range	3	uimsbf
Frame_Field_Pred	1	uimsbf
X_Tab_even	2*8*8*(2**Pred_Range)	uimsbf
Y_Tab_even	2*8*8*(2**Pred_Range)	uimsbf
if(Frame_Field_Pred)		
X_Tab_odd	2*8*8*(2**Pred_Range)	uimsbf
if(Frame_Field_Pred)		
Y_Tab_odd	2*8*8*(2**Pred_Range)	uimsbf

Annex 2 Decoding procedure.

The parameters below need to be known to make predictions.

X_even0	Horizontal half pixel position of point 0 to be used for prediction of the even field.
X_even1	Same for point 1.
X_odd0	Horizontal half pixel position of point 0 to be used for prediction of the odd field.
X_odd1	Same for point 1.
Y_even0	Vertical frame line position of point 0 to be used for prediction of the even field.
Y_even1	Same for point 1.
Y_odd0	Vertical frame line position of point 0 to be used for prediction of the odd field.
Y_odd1	Same for point 1.
W_even0	Weight to point 0 of prediction of the even field.
W_even1	Weight to point 1 of prediction of the even field.
W_odd0	Weight to point 0 of prediction of the odd field
W_odd1	Weight to point 1 of prediction of the odd field

When decoding the "motion vectors" the decoder produces two indices i and j. We need a transformation from the indices i,j to predictor deflections X and Y. Let **i** and **j** be the absolute values of i and j. From these two parameters we need to produce the corresponding horizontal and vertical deflections to be used for prediction. The relations needed are given below:

W_even0 = (Get_LSB(Y_Tab_even(**j**,0))+2)/4
W_even1 = 1 - W_even0
W_odd0 = (Get_LSB(Y_Tab_odd(**j**,0))+2)/4
W_odd1 = 1 - W_odd0

Y_even0 = Sign(j)•Y_Tab_even(**j**,0)>>1
Y_even1 = Sign(j)•Y_Tab_even(**j**,1)>>1
Y_odd0 = Sign(j)•Y_Tab_odd(**j**,0)>>1
Y_odd1 = Sign(j)•Y_Tab_odd(**j**,1)>>1

X_even0 = Sign(i)•X_Tab_even(**i**,Get_LSB(Y_even0))
X_even1 = Sign(i)•X_Tab_even(**i**,Get_LSB(Y_even1))
X_odd0 = Sign(i)•X_Tab_odd(**i**,Get_LSB(Y_odd0))
X_odd1 = Sign(i)•X_Tab_odd(**i**,Get_LSB(Y_odd1))

Get_LSB(B) returns the Least Significant Bit(LSB) of the bitstring B.

Annex 3. Prediction of chrominance.

The same motion vector is used for both luminance and chrominance. If the sampling formats are different for luminance and chrominance the motion vector components have to be rescaled.

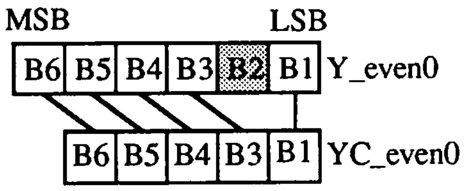
4:2:2
The vertical resolution is the same for luminance and chrominance. The horizontal resolution is only half for chrominance. The horizontal vector components therefore have to be divided by 2 as is also defined in TM1.

4:2:0
In this case the vertical resolution is also halved for chrominance. The vertical vector components therefore have to be divided by 2. But since we have an interlaced signal we must be careful. For a given index j, the offsets used for luminance are Y_even0, Y_even1, etc. The corresponding offsets for chrominance - YC_even0, etc. - must refer to the same field as the luminance offsets.

The corresponding values may be given as a table:

Y_even0	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
YC_even0	-3	-4	-3	-2	-1	-2	-1	0	1	0	1	2	3	2	3

The figure below indicates how the transformation is done on a bit basis.



The weghts - W_even0, etc. from annex 2 - are the same for chrominanceas for luminance.

Annex 4 Motion search for encoding.

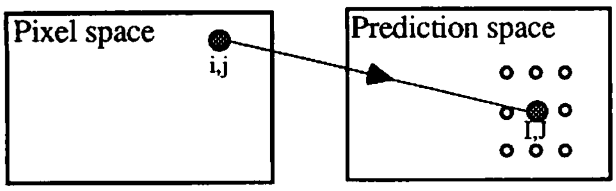
The motion search is assumed to be made in two steps as in TM1.

Step1:
Integer motion search is performed as in TM1. In this search only fields of the same parity is compared. As a result of this search we find the integer motion components i and j.

Next i and j must be transformed to the indices in "prediction space" - I and J - that matches best to the physical motion represented by i and j.

Step2:
This step should be compared to the "1/2 pixel search" in TM1. The 9 positions corresponding to I-1,I,I+1 and J-1,J,J+1 are calculated.

The process of the two steps and transformation between the "pixel space" and "prediction space" is indicated in the figure below.



● Corresponding points in pixel and prediction space.

○ Points for step 2 in prediction space.

Annex 5 Hardware considerations.

Advantage.
Less modes are needed.

Additional decoder hardware requirements.

1: Memory to store the predictions.
Memory is needed to store the matrices defined in the syntax in Annex 1. The size of the memory is

512*2**Pred_range. (bits)

"Pred_range" could be limited to maximum 4 giving maximum memory size of 8 kb.

2: Conversion between prediction indices and deflection values.
Conversion between prediction indices and addresses to be used for making the predictions has to be made. The operations are table look-ups and simple operations described in Annex 3.

Annex 6. Experiments to be performed.

The method of encoder defined predictions covers most of the prediction modes used so far plus additional definitions that may be defined in the future. Considering the already defined modes like field and frame based prediction no extra testing is needed.

Concerning the "FAMC like" definition there is one experiment that should be performed for possible simplification of the weights used for prediction.

Experiment:
Take the definition of FAMC as given in TM1 as a bases. The vertical weights used in that definition implies divisions (that are not simple shifts).

- Do rounding of the vertical weights to the nearest value that is described by: INTEGER/4.
  This is in accordance with this document.
- Do rounding of the vertical weights to the nearest value that is described by: INTEGER/2.
  This would be a further simplification that also would remove multiplications when calculating the predictions.