OPEN ACCESS

University of BRISTOL

Hill, P. R., Chiew, T-K., & Bull, D. R. (2006). Interpolation free sub-pixel motion estimation for H.264. In 2006 IEEE International Conference on Image Processing, Atlanta, GA, United States. (pp. 1337 - 1340). Institute of Electrical and Electronics Engineers (IEEE). DOI: 10.1109/ICIP.2006.312581

Peer reviewed version

Link to published version (if available):
10.1109/ICIP.2006.312581

Link to publication record in Explore Bristol Research
PDF-document

University of Bristol - Explore Bristol Research

General rights

This document is made available in accordance with publisher policies. Please cite only the published version using the reference above. Full terms of use are available:
http://www.bristol.ac.uk/pure/about/ebr-terms.html

INTERPOLATION FREE SUB-PIXEL MOTION ESTIMATION FOR H.264

P.R. Hill, T. K. Chiew and D.R. Bull

Dept. of Electrical and Electronic Engineering, The University of Bristol, Bristol, BS5 1UB, UK.
paul.hill@bristol.ac.uk

ABSTRACT

Sub-pixel motion compensation plays an important role in compression efficiency within modern video codecs such as MPEG2, MPEG4 and H.264. Sub-pixel motion compensation is implemented within these standards using interpolated pixel values at $^1/_2$ or $^1/_4$ pixel accuracy. Such interpolation gives a good reduction in residual energy for each predicted macroblock and therefore improves compression. However, such interpolation is very computationally complex for the encoder. This is especially true for H.264 where the cost of an exhaustive set of macroblock segmentations need to be estimated in order to obtain an optimal mode for prediction. This paper presents a novel interpolation-free scheme for sub-pixel motion compensation using the result of the full pixel SAD distribution of each motion compensated block applied to an H.264 encoder. This system produces reduced complexity motion compensation with a controllable trade-off between compression performance and encoder speed. These methods facilitate the generation of a real time software H.264 encode.

Index Terms— Video coding, Interpolation, Motion compensation

1. INTRODUCTION

Sub-pixel accuracy for motion compensation is traditionally made possible using interpolated reference frames. The creation and use of such interpolated reference frames has a significant implication for the computational load of the encoder and memory bandwidth requirements. The use of multiple reference frames within the H.264 encoder is required for enhanced prediction (resulting in better compression) and error resilience. Such use of multiple reference frames will obviously have considerable memory requirements when using interpolated reference frames. This paper introduces an interpolation free method for sub-pixel block based motion estimation in order to reduce memory bandwidth requirements and improve computational efficiency in order to facilitate real time encoding and in situations with limited memory and memory bandwidth.

In section 2 of this paper, a parabolic model of the sub-pixel resolution motion estimation cost is described that uses the SAD cost at the best whole pixel resolution position and its neighbours. Section 3 describes the generation of the parameters for the model described in section 2 using the whole pixel resolution results. Once the parameters for the model are generated, the minimum value of the model is estimated using the techniques described in section 4. This method when used in a simple and direct way was found to give worse results (in a rate-distortion sense) than the fully interpolated case. In order to improve the location of the actual minimum using the developed method an interpolated fallback was developed as described in section 5. The check used for the fallback mechanism is described in 5.1 supported by the results given for the two fallback checks given in section 6. Finally, a conclusion and summary of the developed methods is given in section 7.

2. MODEL DESCRIPTION

In the H.264 reference software, quarter pixel motion estimation is enabled using interpolated reference frames. Our method obtains sub-pixel accuracy by finding the minimum of a 2D parabolic curve. This curve is fitted to the 9 whole pixel SAD neighbourhood values surrounding the whole pixel motion estimation minimum. This model is reasonable for any stationary 2D signal and extremely close to the actual interpolation surface with Gaussian shaped autocorrelation functions [1].

The parametrically controlled parabolic surface used to estimate the sub-pixel SAD values is defined by the eq. [2]:

$$SAD_i(\zeta) \approx A\zeta_x^2 + B\zeta_y^2 + C\zeta_x\zeta_x + D\zeta_x + E\zeta_y + F \quad (1)$$

Where $SAD_i(\zeta)$ is the estimated SAD value of the ith block. The ζ_x and ζ_y values are the co-ordinates of the estimation, centered at the best motion vector at whole-pixel resolution (they vary from 1.0 to -1.0) with the co-ordinate (0,0) being the best motion vector at whole-pixel resolution. Therefore for the quarter-pixel case used in our experimental H.264 encoder, the parameters ζ_x and ζ_y take the values [-1,-$^3/_4$,-$^1/_2$,-$^1/_4$,0,$^1/_4$,$^1/_2$,$^3/_4$,1].

After the motion vector at full pixel resolution is obtained, the SAD values of its nearest neighbours are made available (calculated or retrieved) giving 8 nearest SAD neighbours from which the parameters A, B, C, D, E and F can be es-

Authorized licensed use limited to: UNIVERSITY OF BRISTOL. Downloaded on January 23, 2009 at 05:50 from IEEE Xplore. Restrictions apply.

timated. These neighbours are shown in figure 1. For the subsequently described techniques, the 8 neighbours are either labeled near neighbours (with even indices) or far neighbours (odd indices).

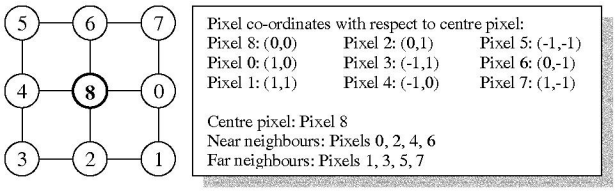

Fig. 1. Illustration of position of neighbour pixels

There are 9 potential points for the estimation of the 6 parameters within eq. 1. The system is therefore overdescribed and there are therefore many possible estimation methods. Chiew [3] presented three models for this parameter estimation. The first method used an underdetermined model based just on the near neighbours (defined as the Near-Neighbours Model (NNM)). A second method used an Overcomplete System Model (OSM) that used all the 9 neighbour points and a pseudo matrix inverse method for obtaining A-F. These methods were proved to give inferior results to our chosen method; the Complete-System Model (CSM).

3. THE COMPLETE SYSTEM MODEL (CSM) FOR PARABOLIC PARAMETER ESTIMATION

The CSM parameter estimation model has been found to be the most efficient computationally and in terms of compression [3]. In this model, the parameters A, B, D, E and F are calculated from eqs. 2 [the values of S are defined in [3]]. Eqs 2 are derived from the simple insertion of neighbour values into eq. 1 shown in figure 1. The value of C is firstly set to zero in the under-determined case (NNM). However within the CSM method the value of C is chosen from the set C_1, C_3, C_5, C_7 where C_k is the value of C found with the complete system of equations using points 0, 2, 4, 6, 8 and k (as defined in figure 1). Then the value of C is chosen using eq. 3, where $\hat{S}_{ki}$ is the estimate of S_i using eq. 1 with parameter set A, B, C_k, D, E, F. This model determines which of the far-neighbours best fits the model and ignores the other 3, thus removing the effect of outliers. Its complexity is similar to that of the NNM method whilst also guaranteeing the existence of a minimum point.

$$A = -S_8 + \tfrac{1}{2}(S_0 + S_4);\ B = -S_8 + \tfrac{1}{2}(S_2 + S_6);$$
$$D = \tfrac{1}{2}(S_0 - S_4);\ E = \tfrac{1}{2}(S_2 - S_6);\ F = S_8 \qquad (2)$$

$$C = \underset{k=1,3,5,7}{arg\,min} \sum_{i=1,3,5,7} \left| S_i - \hat{S}_{ki} \right| \qquad (3)$$

4. OBTAINING THE SURFACE MINIMUM

A parabolic surface (i.e. second order polynomial) as defined in eq. 1 will only have one minimum in continuous space. However this minimum is not guaranteed to be within the (-1,1)×(-1,1) area. Therefore analytical methods are not appropriate. The technique adopted in [3] is to evaluate the value of $S(x, y)$ for all sub-pixel locations within the (-1,1)×(-1,1) area. This achieves the desired result but has an associated high computational complexity due to its exhaustive search and the large number of multiplications associated with eq. 1.

We now present three methods for calculating the minimum of the parabolic surface without an exhaustive search.

4.1. Obtaining Parabolic Minimum Methods 1 & 2

The basis of this method is to start at an origin and check the $S()$ value of its near (4-connected) neighbours (at quarter pixel accuracy). If any of these values are less than the value of $S()$ at the origin then move the origin to the position of the new value and then repeat until the minimum is found.

This method is similar to the block based gradient descent motion compensation scheme utilised within the H.263 standard reference software and defined by Liu and Feig[4]. However, this is only using interpolated values whereas our method uses the estimated SAD values derived from the parameterised parabolic surface (eq. 1). This method is not guaranteed to find the minimum value of $S(k)$ within the (-1,1)×(-1,1) area as the descent of the gradient can get caught in local minima. This problem is reduced using an 8-connected search. Method 1 using an 8-connected search is defined as method 2.

4.2. Obtaining Parabolic Minimum Method 3

A two stage algorithm checks the value of the 8-connected $^1/_2$ pixel resolution positions (and the origin) relative to the origin. The final minimum of $S()$ is then the minimum of an 8-connected $^1/_4$ pixel resolution check centred on the minimum at the $^1/_2$ pixel resolution (including the minimum at the $^1/_2$ pixel resolution).

4.3. Results for Obtaining Minimum Methods

Figure 2 shows the results of using the three methods described above. The top figure shows an insignificant difference in rate-distortion efficiency between using any of these methods. However, when examined closely (see inset) it can be seen that all three methods have a slight cost involved when compared to the full search. The cost for each method is variable over the rate distortion curve but is approximately 0.2% of the rate at a fixed PSNR value. This cost is the highest for method 1 using a 4-connected search followed by the 8-connected search of method 2 and finally method 3. However,

Authorized licensed use limited to: UNIVERSITY OF BRISTOL. Downloaded on January 23, 2009 at 05:50 from IEEE Xplore. Restrictions apply.

the speed of these methods shown in the bottom figure of figure 2 shows the significant speed improvement using method 1. As the compression performance for all the methods is insignificant compared to the full search (top figure) method 1 is preferred.

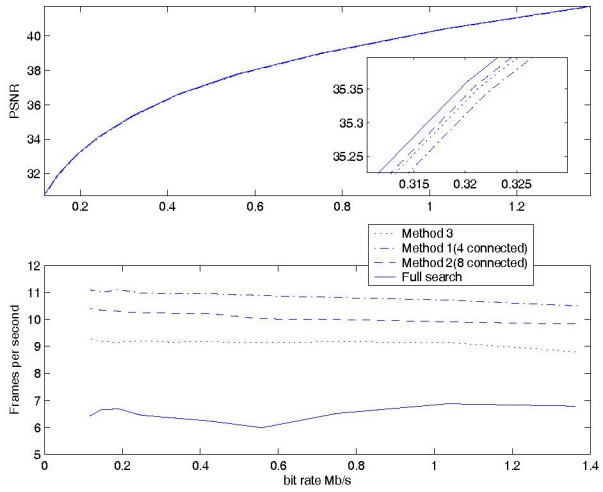

Fig. 2. Results for obtaining the parabolic minimum methods (Foreman Sequence: 100 frames, CIF). Top: Rate distortion results, Bottom: Encoder speed in frames per second.

5. INTERPOLATED FALL-BACK

The results from method 1, provides a reduction in bitrate when compared to the simple full pixel case. However, this method is measurably worse (in terms of rate distortion performance) compared to the fully interpolated case (as shown in figure 3).

We therefore adopt the concept of the interpolated fallback. This involves a check that measures the quality of the result from method 1. The motion vector from method 1 is kept without any further method if the result of the fallback check is positive. However if the fallback check is negative then the usual interpolation method is used. This has an impact on computationally efficiency. The greater the use of interpolation, the better the bit rate but higher the computational load. This therefore allows a trade off between computational efficiency and compression according to a quality threshold.

5.1. Fallback check

The fallback check should reflect how good method 1 (4-connected) is at achieving a SAD minimum similar to that which would have been obtained by interpolation. **Fallback check 1:** The CSM method for obtaining the correct parameters for the parabolic minimum surface does not give an exact match for the far neighbours. i.e. $S()$ in eq. 1 does not always equal the actual SAD values for the far neighbours. Fallback

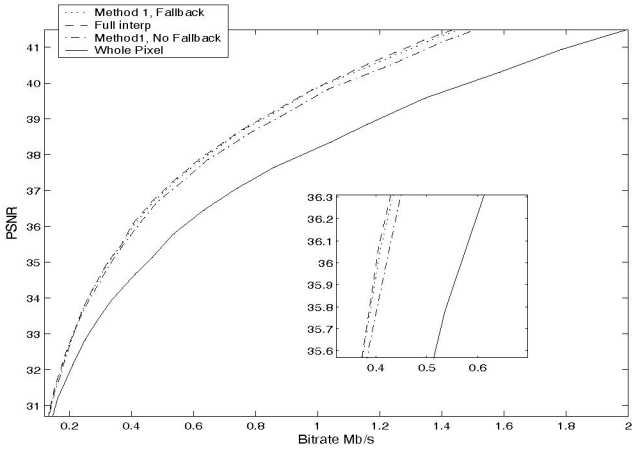

Fig. 3. Comparison of results for sub-pixel methods: Foreman

check 1 is therefore a measure of how well the parabolic surface model fits the SAD values for the far neighbours. This is achieved by obtaining a measure of the divergence from the model $DivMod$. $DivMod$ is the average difference between the predicted value of $S()$ (using eq. 1) with the actual SAD values for the far neighbours (see eq. 5). Fallback check 1 is therefore a check to see if this value is over a threshold (see eq. 5).

$$DivMod = \sum_{i=1,3,5,7} \left| S_i - \hat{S}_{ki} \right| \quad (4)$$

$$Fallbackcheck1: \quad DivMod/(B_h \times B_w) > thresh1 \quad (5)$$

Where S_i is the predicted SAD values at far neighbour positions 1,3,5 and 7. $\hat{S}_{ki}$ is the actual SAD values (same positions). B_w and B_h indicate the mode block size.

B_w and B_h are included to normalise the effect of different sized blocks. Figure 4 shows how the variation of the threshold changes the percentage of interpolation fallback and its associated compression and timings. The threshold sets the trade off between compression and speed. **Fallback check 2:** A more direct method of checking how good method 1 is, is by getting the actual interpolated SAD value at the quarter pixel motion vector position indicated by the minimum found by method 1. The absolute difference between the actual interpolated SAD value at this point and the predicted SAD value (S() in eq. 1) using method 1 is compared to a threshold (as in eq. 5) to form Fallback check 2. As with fallback check 1, there is also a trade off between compression and speed according to the value of the threshold. This is also shown in figure 4. N.B. the cost function of each block is based on the actual interpolated SAD value. This significantly reduces the bitrate compared to using the estimated SAD value.

Authorized licensed use limited to: UNIVERSITY OF BRISTOL. Downloaded on January 23, 2009 at 05:50 from IEEE Xplore. Restrictions apply.

6. FALLBACK CHECK RESULTS

Figure 4 shows the results of the two fallback check methods described above. The top line of the top graph shows the bit rate of a system without any interpolation and the bottom line of the same graph shows the bit-rate of the system with full interpolation. Between these two lines the two threshold methods decrease the bit rate as the proportion of interpolation increases. However this is a much greater increase than from a random selection of interpolation. The lower graph shows the frame rate of the systems with the top and bottom lines similarly delimiting the non and full interpolation modes. The decreasing graph shows how both systems become quicker the less interpolation is performed.

Fallback check 1 was seen to give moderately better results and was chosen. As bitrate conservation was chosen as a priority the threshold of 2.0 (threshold 1) was chosen giving a 40% proportion of interpolation (approximately). As shown in figure 4, this gives a significant reduction in bit rate compared to the non-interpolated case but also has a reasonable speed-up (8fps). This method and threshold were used for all subsequent experiments (in the tables below).

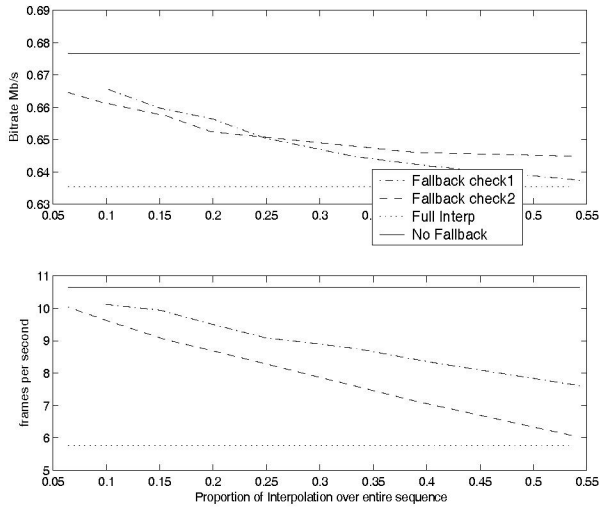

Fig. 4. Comparison of threshold methods. Foreman CIF × 100. QP = 26, almost exactly the same PSNR over bitrate range

	Full Interp	FBack	No FBack	Whole Pixel
Foreman	7.3034	8.5351	10.5877	14.9931
Akiyo	9.7557	11.7719	12.2695	15.6095
Coast	7.1907	7.6270	12.1715	16.2123
Hall	10.902	13.197	13.7286	17.8572
Stefan	6.4978	7.2846	10.3925	14.9289
Mobile	5.8278	8.0765	11.3227	15.1986

Table 1. The speeds of methods in frames per second: CIF×100

	Fallback	No Fallback	Whole Pixel
Foreman	0.4004	1.5703	10.9342
Akiyo	0.0027	0.2316	0.7219
Coast	0.2333	0.7102	8.5228
Hall	0.0000	0.0378	0.4380
Stefan	0.1249	1.9357	16.5843
Mobile	0.2066	3.5145	21.3099

Table 2. Difference between rate distortion curves (measured in % PSNR difference over logarithmic scale [5]) compared to full interpolation curve: CIF×100

7. CONCLUSION

Quarter pixel motion compensation provides a key contribution to the compression performance of H.264. However, the interpolation required to give an accurate estimation for quarter pixel motion compensation is very computationally intensive. This is compounded by the exhaustive search used by the H.264 encoder to check all possible prediction modes. Interpolation free quarter pixel motion estimation is able to give a substantial reduction in computational complexity. This is shown in table 1 where the speed improvement of the non fallback method over full interpolation ranges from 25% to 60%. However, as is shown in figure 3 this method does not match the performance (in a rate-distortion sense) of a fully interpolated system. The fall back methodology introduced in this paper improves the rate distortion performance to a level close to full interpolation with only a slight increase in complexity (compared to the non fallback system). Table 2 illustrates this, comparing the difference in rate distortion curves of the three methods.

8. REFERENCES

[1] G. Giunta, "Fine estimators of 2D parameters and application to spatial shift estimation," *IEEE Trans. Signal Processing*, vol. 46, no. 12, pp. 3201–3207, Dec 1999.

[2] G. Giunta, "Fast estimators of time delay and Doppler stretch based on discrete-time methods," *IEEE Trans. Signal Processing*, vol. 46, pp. 1785–1797, July 1998.

[3] T.K. Chiew, *Rapid Segmentation for Video Processing*, Ph.D. thesis, The University of Bristol, 2004.

[4] L.K. Liu and E. Feig, "A Block-Based Gradient Descent Search Algorithm for Block Motion Estimation in Video Coding," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 6, no. 4, pp. 419–423, 1996.

[5] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," *VCEG document VCEG-M33*, March 2001.

Authorized licensed use limited to: UNIVERSITY OF BRISTOL. Downloaded on January 23, 2009 at 05:50 from IEEE Xplore. Restrictions apply.