# LOW RATE VIDEO FRAME INTERPOLATION – CHALLENGES AND SOLUTION

*Hezerul Abdul Karim, Michel Bister, M. U. Siddiqi*

Faculty of Engineering, Multimedia University, 63100 Cyberjaya, Selangor, Malaysia

**ABSTRACT**

The low frame-rate specific challenges are illustrated by choosing one well-performing algorithm in high frame rate and applying it to low frame-rates. The performance is illustrated by comparing original algorithm, algorithm adapted to low frame rate particularities, and simple averaging. To overcome the particular challenges of low frame rate, two algorithms were developed and compared on objective and subjective basis and shown to provide elegant solution to the specific challenges of low frame-rate video interpolation.

## 1. INTRODUCTION

In recent years several frame interpolation algorithms have been developed ([1]-[7]). Most of them concentrate on high frame rate video. In this case, motion estimation can be achieved by simple block matching technique. Much less work has been done on low frame rates. In this paper, the basic concept of motion estimation in frame interpolation is described first, followed by the frame interpolation challenges in low rate video frames. The algorithms used for frame interpolation are then explained. Results, discussions and conclusion are given at the end part of the paper.

Motion estimation is a process of determining the movement of objects within a sequence of image frames. Block matching is a widely used technique for translation movement estimation. In block matching method for frame interpolation, a block of pixels from the previous frame is matched to a displaced block of pixels in the search area in the next frame. The block that gives minimum matching error will be assigned a displacement value called motion vector, which is used to interpolate the missing pixel in the frame to be interpolated.

To illustrate the challenges of interpolation at low rate video frames, we took a successful algorithm for high frame rate and applied it to low-frame-rate with and without adapting the parameters. Then, a new contribution is presented.

When the frame rate is reduced by a factor of N, the search space has to be increased. This means $S_x$xN and $S_y$xN, hence the risk for false matches between unrelated regions in the search space. Therefore the block size has to be increased by N. This means $B_x$xN and $B_y$xN. With these adjustments, the complexity increases by $N^4$. This large amount of computation has to be performed in the available time, which has increased by N. The increase of complexity by $N^4$ because of the speed reduction by N is known as the combinatorial explosion problem.

## 2. OVERVIEW OF ALGORITHMS USED

Several algorithms to interpolate the low rate video frames will be evaluated. These algorithms are Averaging, Castagno [1] and Adapted Castagno. A novel approach of multi-resolution motion estimation (MRME) is developed and made adaptive (AMRME) to overcome observed artifacts.

The simplest method to do frame interpolation is by Averaging. The previous frame and next frame are averaged at every pixel location in the image to give the interpolated frame.

The Castagno algorithm [1] was designed for a frame rate of 50 frames per second (fps). It performs block matching that uses block size of 3x5 and the search space is limited to 5x9. The motion vector is estimated for every pixel in the interpolated frame. For current pixel, the motion vector is found by searching around 3x3 neighborhood of the previous pixel's motion vector. Hence, a total of 9 motion vectors are evaluated using weighted mean absolute difference (MAD). The motion vector that gives minimum weighted MAD is chosen to be the motion vector for the current pixel. This is repeated for every pixel. When all the pixels' motion vectors have been estimated, a motion vector field is produced, which are post-processed using 3x3 median filter to remove inconsistent motion vectors. Interpolation is performed based on these motion vectors.

The Adapted Castagno method is basically the same method as Castagno but with the following modifications: 1) Instead of evaluating only nine motion vectors, all of the motion vectors in the search range are evaluated. 2) The block and the search range are adapted to a low frame rate of 5 fps, which is ten times lower than the frame rate of Castagno. The search space is increased to 50x90, about ten times the original search space of Castagno. The block size to do block matching is set to 15x9 pixels, three times the block size of Castagno, which should be ten times according to the explanation before. However, it was found that increasing the block size further than this did not improve the results and it is reasonable to the size of the search space. This setting is achieved by calculating the fastest motion between previous and next frame of a sequence (e.g. Suzie sequence). Clearly, the computational complexity for this method is higher than before as more motion vectors need to be evaluated (4641 motion vectors for every pixel). In this method, weighting of the MAD is not implemented. Weighting makes the algorithm favour vectors that are closer to (0,0). Hence, it will not choose the large motion vector. Favouring motion

vector (0,0) makes the performance of the algorithm almost similar to Averaging algorithm during the frame with fast motion.

In the MRME algorithm that we propose, the image is filtered using an averaging low-pass filter and sub-sampled to produce successively reduced-resolution versions. Using QCIF images (176x144 pixels), five levels of resolution are considered, the lowest one having 11x9 pixels. Motion is estimated at the coarsest resolution first to find the global motion. Block size used is 9x9 and full search is used. To improve the consistency of the motion field, it is filtered with a 3x3 median filter. This motion field is then used as an initial estimate in the next finer resolution.

At lower levels, the search space for each pixel is set to 3x3 around the initial estimate. The block size is maintained at 9x9 at each level. The estimate produced at each level is again submitted to a 3x3 median filter before passing on to the next level. In this way, the motion vectors are refined through the pyramid levels until the highest resolution at the lowest level. The motion vectors are then used to interpolate the luminance levels. An advantage of the MRME algorithm is that while motion is detected at high resolution, large motion can be detected with small search space on low-resolution image, requiring only a small block size. In the high-resolution image, a small search space is also used around the previous estimate, hence also requiring only a small block size.

Problems arise in the multi-resolution pyramid algorithm due to its rigid structure [8]. The same problems arise in MRME, and can be alleviated by making the search space adaptive, as illustrated in Figure 1,which illustrates an object sliding from left to right. To visualise the pyramid better, only two levels have been represented.

On the higher level (level k+1), the movement is already estimated. The pixel in pattern is estimated to move by 1 unit in horizontal direction, (0,1). Other pixels have (0,0) motion vectors except the middle right pixel, which is at the boundary. The lower level (level k) has twice number of pixels of the higher level. One pixel at level k+1 is formed by four pixels at level k, so the initial motion vectors at level k are twice the motion vectors of level k+1.

Motion estimation is performed based on the initial estimate. Motion vectors shown in Figure 1 for level k are the resulting motion vectors after the motion estimation on the initial estimate. For four pixels (in pattern) in the middle, the initial estimate is (0,2). In case of MRME, the search range is 3x3 around this initial estimate. So, the motion vector candidates for the top left of this pixel group are (-1,1), (-1,2), (-1,3), (0,1), (0,2), (0,3), (1,1), (1,2) and (1,3). Motion vector (0,2) will be chosen as its match.

The problem occurs for pixel P, who has initial motion vector of (0,0). The correct motion vector for pixel P is (0,2). The motion vector candidate range is $(-1 \le a \le +1, -1 \le b \le +1)$. It is clear that this search range is insufficient to match the correct pixel in the next frame. It should be $(-2 \le a \le +2, -2 \le b \le +2)$ for the pixel P to be matched.

The solution to this problem that has been developed in this work is based on an adaptive search space, based on the previously used 3x3 space, but extended to encompass the initial motion estimates of the four neighbouring pixels.

The initial estimate of the motion vector (a,b) has been influenced by its three neighbours who make up the pixel on the next higher level. The rigidity of the pyramid structure did not allow for a border between white and pixels in pattern to be located at any other place but at the limits between the next higher pixels. The problem occurs because the movement of pixel P is larger than the 3x3 search space. This measurement was well detected at lower resolution levels. However at this low resolution the border could not be accurately located because it is mis-aligned with the pyramid grid by one-pixel. The original 3x3 search space of pixel P made it impossible to match it with its matched in the next frame.

However, now we extend the search space of pixel P to ensure that the initial estimates of the four neighbouring pixels are included. The search space for pixel P is extended to make it a rectangle that encompasses the initial estimate of the motion vector for pixel R. So, the search space for pixel P is extended from $(-1 \le a \le +1, -1 \le b \le +1)$ to $(-1 \le a \le +1, -1 \le b \le +2)$. Hence, the correct motion vector for pixel P, (0,2), is included in this new search space. With this extended search space, it is indeed possible to find a motion vector that matches the pixel P with its match in the next frame.

The difference between the MRME and the AMRME algorithms is that the search space for each pixel is set to 3x3, and then extended to encompass the initial estimates of its four neighbours. In this way, pixels that are located just at a (rigid) boundary of the pyramid structure do not get isolated in their movement from the pixels that are just on the other side of the boundary, and the rigidity of the pyramid structure is overcome.

## 3. RESULTS AND DISCUSSION

The results for the algorithms are compared objectively and subjectively. The 20 original image sequences at 30 fps is compressed using H.263+ to 10-fps. Down sampling is performed on this 10-fps to get the 5-fps image sequence. The 5-fps is then interpolated to 10-fps using the algorithms discussed. The interpolated 10-fps is finally compared with the originally compressed 10-fps in terms of mean square error (MSE) and picture quality. Figure 2 shows the typical results of interpolation from 5 to 10 fps for Miss America, Susie and Coastguard compressed image sequences. Table I shows the objective and subjective evaluation for only 3 image sequences.

The objective evaluation (MSE) reveals that the AMRME is the best compared to the other algorithms. Only in a few cases is MRME slightly better. For the Container sequence, Averaging is better - which can be explained by the very slow motion in that particular sequence. For the Stefan sequence, Adapted Castagno is slightly better. These results are to be expected, since the algorithms were gradually improved, from Castagno to Adapted Castagno to MRME to AMRME, using the MSE as measuring stick for improvement. The (nice) surprise comes from the fact that the algorithm development was consistently done with one single sequence (Suzie), but the results are consistent when using other sequences, and even other formats (SIF and CIF).

The subjective evaluation is more mitigated, although the average is still in favour of the AMRME algorithm. Adapted Castagno never comes out as the best algorithm. Averaging and

Castagno are preferred in 4 cases (out 19), while all the other cases carry MRME or AMRME as preferences. When MRME is preferred, the difference is usually marginal, while when AMRME is preferred the difference is usually more significant.

The reason why the subjective evaluation is less overwhelmingly in favour of AMRME is probably due to the artifacts. MSE is a kind of "average" error, which does not care about local big errors, as long as the global impression is good, while subjective evaluation is very much affected by local big errors like artifacts. This is illustrated in Figure 3, which shows two corrupted versions of an image of the Suzie sequence. In (a), the error is limited to 6 pixels which were put to zero, while in (b) the error introduced is a gaussian error over the whole image with average zero and variance 0.0002. The MSE is 7.2367 and 13.1458, respectively. Hence, the MSE is clearly in favor of the first image, while the subjective evaluation is clearly in favor of the second one. The artifacts introduced in the Adapted Castagno, MRME and AMRME are of a nature like the error in (a), while the blur introduced by the Averaging and Castagno algorithms are more of the nature of the error in (b).

The global preference, even in the subjective evaluation, is still in favour of the AMRME algorithm, which shows that the improvement introduced compared to the other algorithms more than offsets the disturbing artifacts.

AMRME computational load is much less than Adapted Castagno algorithm. Computational load is of the order of $N_x$ x $N_y$ x $B_x$ x $B_y$ x $S_x$ x $S_y$, whereby ($N_x$, $N_y$) is the image dimension, ($B_x$, $B_y$) is the block size, and ($S_x$, $S_y$) is the search space. In the case of MRME, the image dimension to be taken into account is the sum of the image dimension at each level, namely: (144 x 176) + (72 x 88) + (36 x 44) + (18 x 22) + (9 x 11), with $B_x = B_y = 9$ and $S_x = S_y = 3$. The resulting estimate of calculations to interpolate ONE image is 58,164,480 operations for Castagno, 15,878,903,040 for Adapted Castagno (273 times more than Castagno!), and 24,610,311 for MRME (42% of Castagno). For AMRME, the search space is not fixed and possibly larger than for MRME, so while the number of operations will be large than for MRME, no upper limit can be given. The ratios for the computational load are 1000:1 and 3:1 for Adapted Castagno and AMRME compared to Castagno, respectively. Although AMRME is slower possibly due to the fact that the lower-resolution images have to be calculated before starting the motion estimation, it is to be remembered that Castagno was designed for 50-to-75 fps (hence 40 ms time to do the calculation in real-time). AMRME was designed for 5-to-10 fps (hence 200 ms time to do the calculation in real-time), hence AMRME is still doing better than Castagno is.

## 4. CONCLUSION

Interpolation at low frame rate is a great challenge. Most existing algorithms interpolate at high frame rate (e.g. Castagno). The algorithms have to be adapted to assess fast motion (resulting in large frame-to-frame displacement) occurring at low frame rate. Classical block matching introduces combinatorial problems. Small block size and small search area cannot detect fast motion (e.g. Castagno). On the other hand, large block size and large search area produce mismatches, which lead to artifacts and speed reduction (Adapted Castagno).

The MRME algorithm is proposed and implemented. It estimates the movement first at lower resolution (smaller search space), and then successively increases the resolution and performs search only in a subset of the search space around the search solution from the previous hierarchical level. It manages to reduce artifacts and long computation time introduced by the Adapted Castagno algorithm.

An improved version of MRME algorithm, called the AMRME algorithm, is also proposed. The AMRME algorithm reduces artifacts further because of its adaptive search space, which is obtained by stretching the search space to include the search space of the current, previous, next, top and bottom pixels. The AMRME algorithm is superior to Averaging, Castagno, Adapted Castagno, and MRME algorithms subjectively and objectively.

## 5. REFERENCES


[1] R. Castagno, P. Haavisto, and G. Ramponi, "A method for motion adaptive frame rate up-conversion'', *IEEE Trans. on Circuits Syst. Video Technol. vol. 6, no. 5*, pp. 436-445, 1996.

[2] K. A. Bugwadia, E. D. Petajan, and N. N. Puri, "Progressive-scan rate up-conversion of 24/30 Hz source materials for HDTV'', *IEEE Trans. Consum. Electron. vol. 42, no. 2*, pp 312-321, 1996.

[3] D. W. Kim, J. T. Kim, and I. H. Ra, "A new video interpolation technique based on motion-adaptive subsampling", *IEEE Trans. Consum. Electron.*, *vol. 45, no. 3*, pp 782-787, 1999.

[4] R. Krishnamurthy, J. W. Woods, and P. Moulin, "Frame interpolation and bi-directional prediction of video using compactly encoded optical-flow fields and label fields'', *IEEE Trans. on Circuits Syst. Video Technol.*, *vol. 9, no. 5*, pp 713-726, 1999.

[5] W. R. Sung, E. K. Kang, and J. S. Choi, "Adaptive motion estimation technique for motion compensated interframe interpolation'', *IEEE Trans. Consum. Electron.*, *vol. 45, no. 3*, pp 753-761, 1999.

[6] B. T. Choi, S. H. Lee, and S. J. Ko, "New frame rate up-conversion using bi-directional motion estimation'', *IEEE Trans. Consum. Electron.*, *vol. 46, no. 3*, pp 603-609, 2000.

[7] C. L. Huang, and T. T. Chao, "Motion-compensated interpolation for scan rate up-conversion'', *Optical Engineering*, *vol. 35, no. 1*, pp 166-176, 1996.

[8] M. Bister, J. Cornelis, and A. Rosenfeld, "A critical view of pyramid segmentation algorithms'', *Pattern Recognition Letters*, *vol. 11*, pp 605-617, 1990.

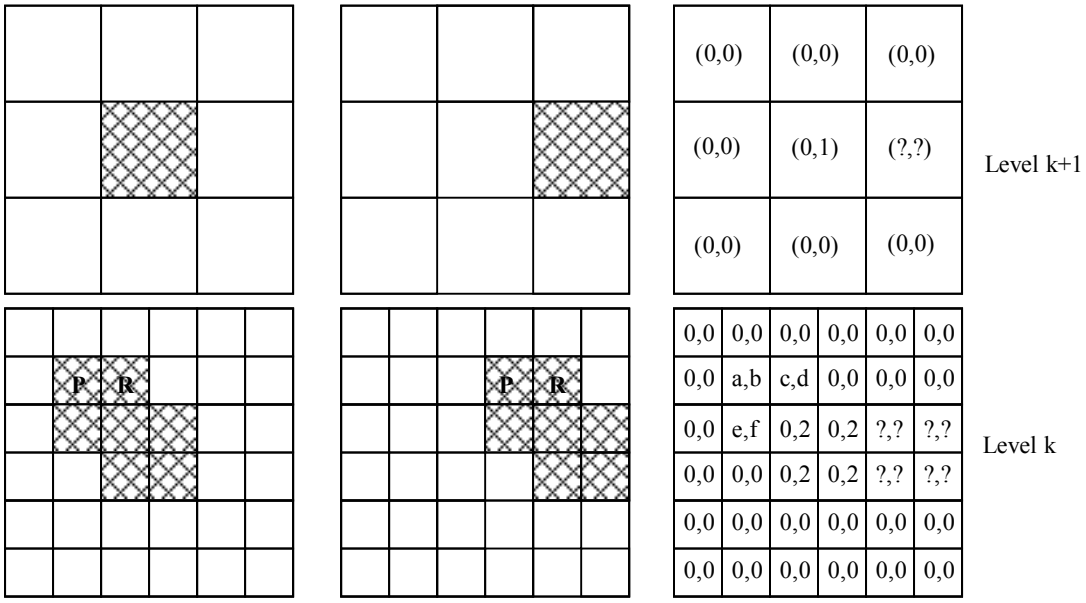


Fig. 1. AMRME illustration

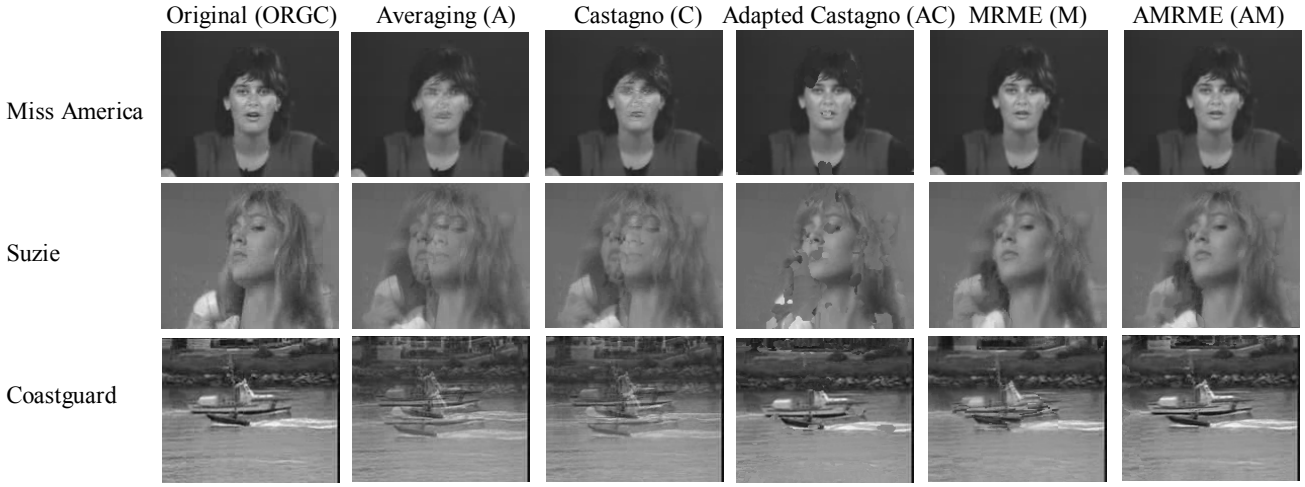


Fig. 2. Typical results for interpolation

(a) (b)

Fig. 3. Suzie image with (a) strong local error (b) small global error

Table I. Objective and subjective evaluation

| | Objective | | | | | |
|---|---|---|---|---|---|---|
| Image | **ORGC** | A | C | AC | M | AM |
| 1.MA[a] | **10.08** | 16.01 | 9.22 | 91.87 | 7.84 | 7.71 |
| 2.Sz[a] | **19.63** | 103.40 | 92.58 | 124.51 | 75.45 | 66.69 |
| 3.Cgd[a] | **77.67** | 230.20 | 205.22 | 316.28 | 225.33 | 150.00 |
| | Subjective | | | | | |
| Image | **ORGC** | A | C | AC | M | AM |
| 1.MA[a] | **3.07** | 2.67 | 2.60 | 0.73 | 3.20 | 3.27 |
| 2.Sz[a] | **2.67** | 1.80 | 2.00 | 1.13 | 2.67 | 2.33 |
| 3.Cgd[a] | **3.07** | 2.67 | 2.60 | 0.87 | 1.87 | 2.53 |

[a] 176×144 – QCIF, MA-Miss America, Sz-Suzie, Cgd-Coastguard, ORGC–Original compressed