A Spatial Scalable Video Coding with Selective Data Transmission using Wavelet Decomposition

by

Lakshmi Veerapandian

Bachelor of Engineering (Information Technology)
University of Madras, India. 2004.

Thesis submitted for the degree
of

Master of Engineering Science
in
School of Electrical and Electronic Engineering

The University of Adelaide

March 2010

6. Results and Discussions

This chapter presents the results of the simulations and relevant discussion. The results are organised into three categories: resolution scalability, error correction and selective data transmission.

Sample Video Data

Video samples were shot specifically for this research. Three sample videos were taken covering the basic aspects of video diversity, such as

- Panning movement of a fast moving foreground with a slow moving background: CarYellow sample

- A slow moving foreground, with a fast moving background object: SwanAndBird sample.

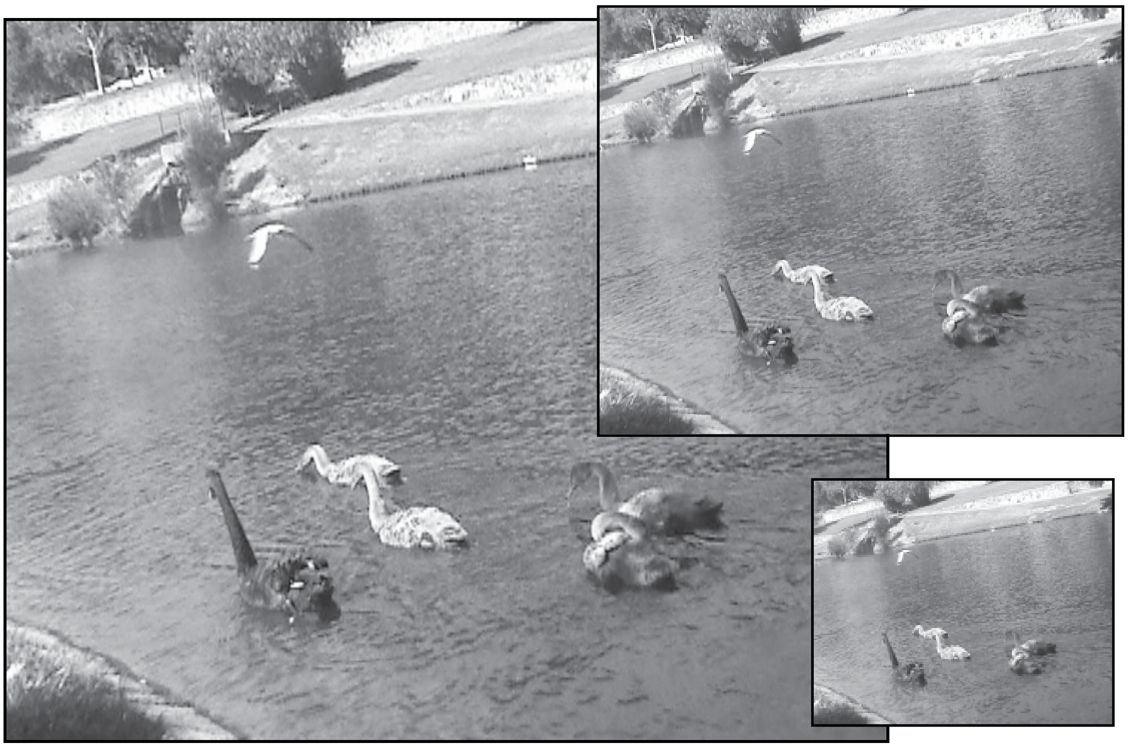
- A slow moving background and foreground: WhiteCar sample

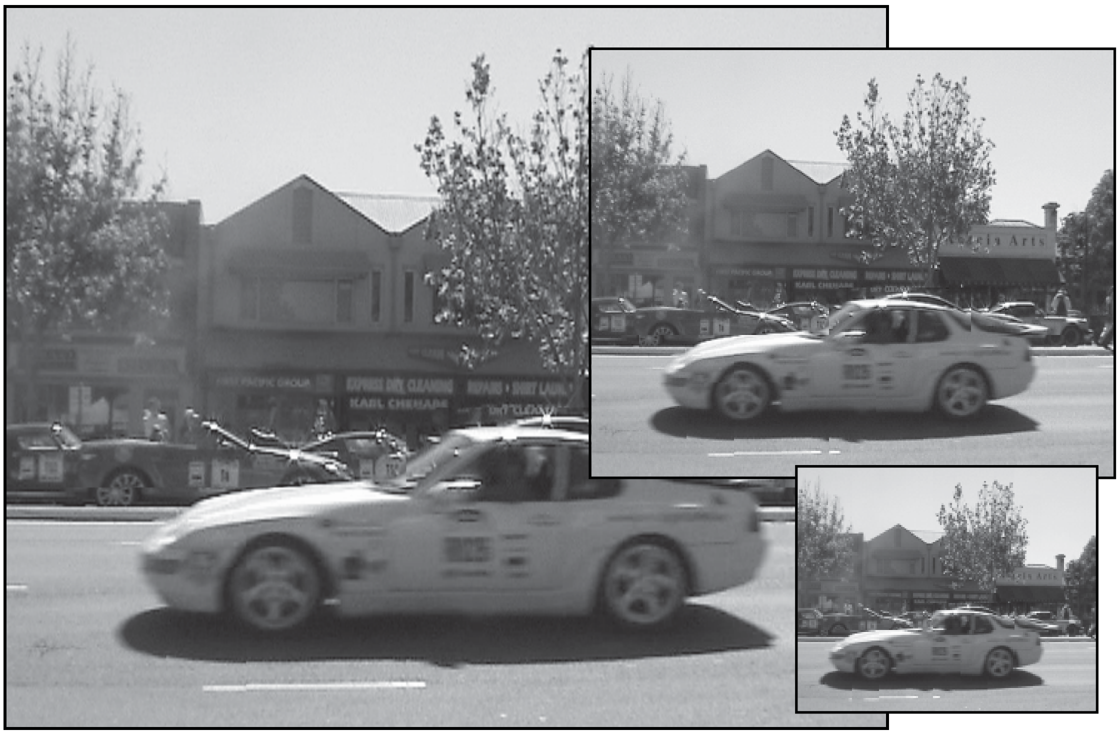
6.1 Resolution Scalability

Three resolutions of size 144x176, 288x352 and 576x704 are extracted as result of wavelet decomposition and multiresolution motion estimation. Using top-down approach, the motion estimation is first performed in resolution level 144x176 (R1) and the resultant motion vectors are scaled to resolution level 288x352 (R2) and 576x704 (R3). The images in Figure 6.1, represent the three resolutions extracted from the video samples: (a) CarYellow, (b)WhiteCar and (c) SwanAndBird videos.

The images in Figure 6.2 (b) represent our resolution-scaled images formed using multiresolution motion estimation without any error resilience. It is then compared against the original resolution images formed through direct motion estimation without scaling, as in Figure 6.2 (a). The images represent samples taken from the three video sequences. From the Figure 6.2(a) and 6.2(b), it is clear that the images formed through resolution scaling have similar visual quality to the original resolution images, but have some error, marked in circles, which are propagated due to motion translation. The error correction and selection transmission function are discussed in the next sections.

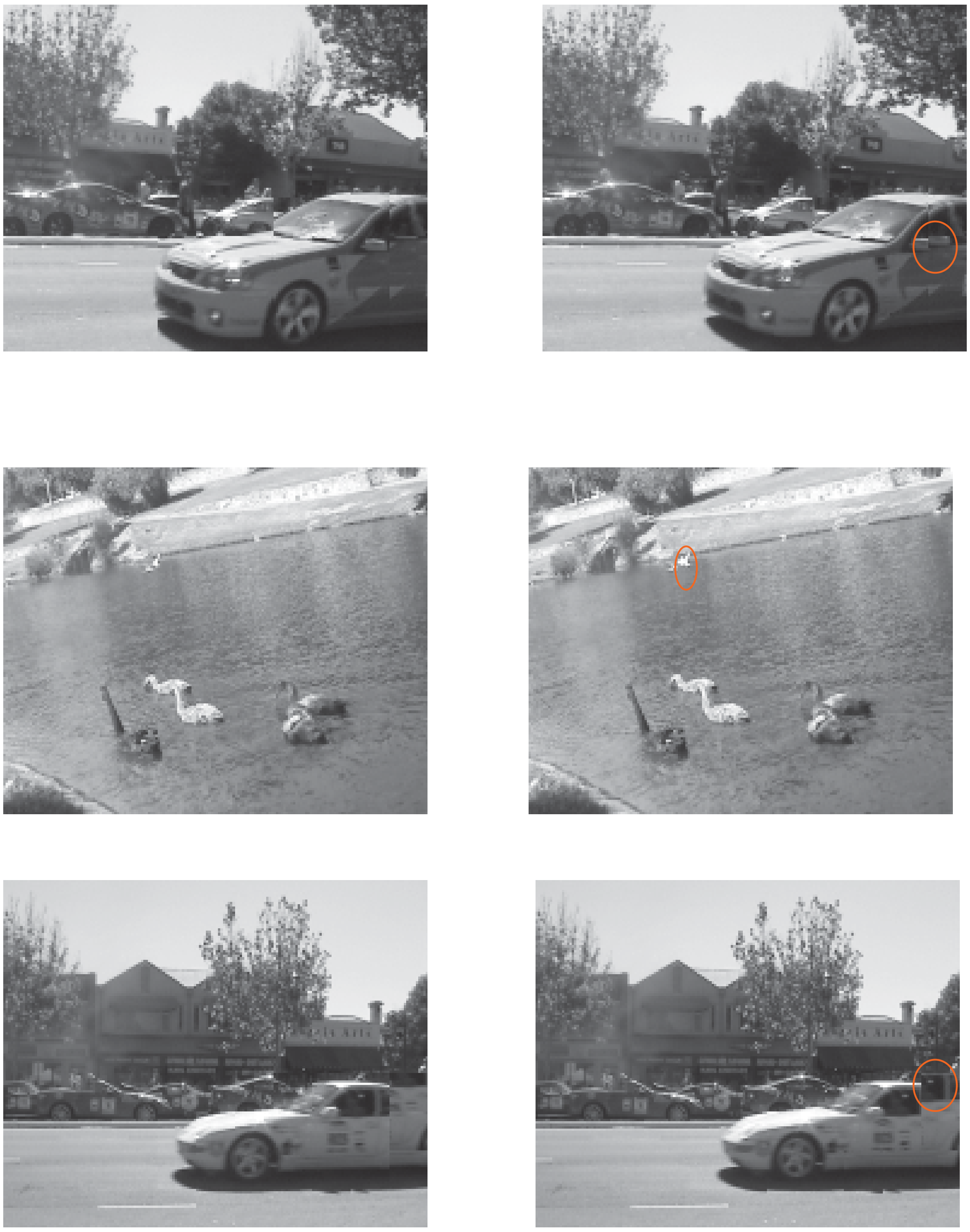
(a) CarYellow sequence


(b) SwanAndBird sequence



(c) WhiteCar sequence

Figure 6.1: The three extracted resolutions from three sample sequences



(a) Direct motion estimate images without scaling

(b) Resolution-scaled images

Figure 6.2: Comparison of resolution-scaled images (without error correction) against images formed through direct motion estimation without scaling

6.2. Error Resilience and Selective Data Transmission

The errors propagated due to motion vector translation from lower resolution are handled using the proposed threshold method. The same threshold method is be used for selective data transmission by varying the two threshold parameters. The extracted resolution images are then subjected to error resilience and tested for selective data transmission in three different threshold modes.

1. Quality threshold constrained for low bandwidth
2. Quality threshold constrained for high bandwidth
3. Quality threshold relaxed for high quality and high bandwidth

This approach was tested on two different motion sequences, CarYellow and SwanAndBird sequences, as discussed below.

6.2.1 Quality Threshold Constrained for Low Bandwidth

This threshold mode is used when the video is required in low quality during low bandwidth situations, as in mobile phone applications. Therefore, both the parameters Threshold and WorstBlock Factor take are set to meet these constraints.

For CarYellow sample, the quality Threshold is constrained to a nominal value around 15000, which is about 40% of the global motion exhibited by the sample image. So the mismatch blocks that fall under this threshold is represented in Figure 6.3.

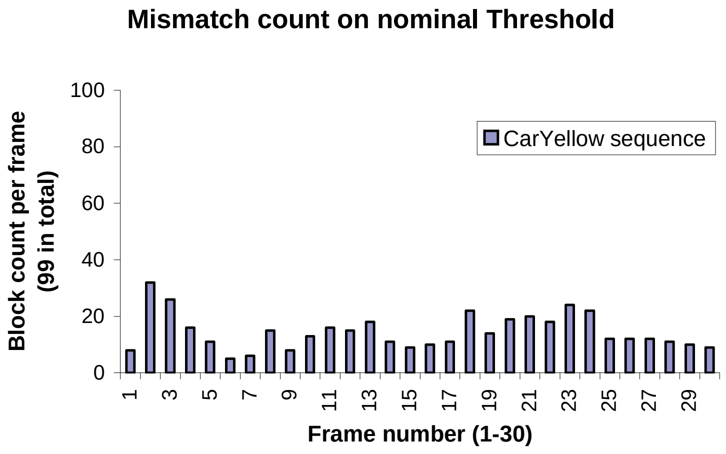


Figure 6.3: Number of blocks identified as mismatches using a nominal Threshold

Only the worst blocks are replaced, to meet the bandwidth constraint. This is achieved using the WorstBlock Factor. The WorstBlock factor is assigned a nominal value, which is of 60%. Hence only the blocks, which exceed both thresholds, labeled as A1 in Figure 6.4, are subjected to replacement. The nominal threshold plot for the CarYellow sample image for selective data transmission is presented below.

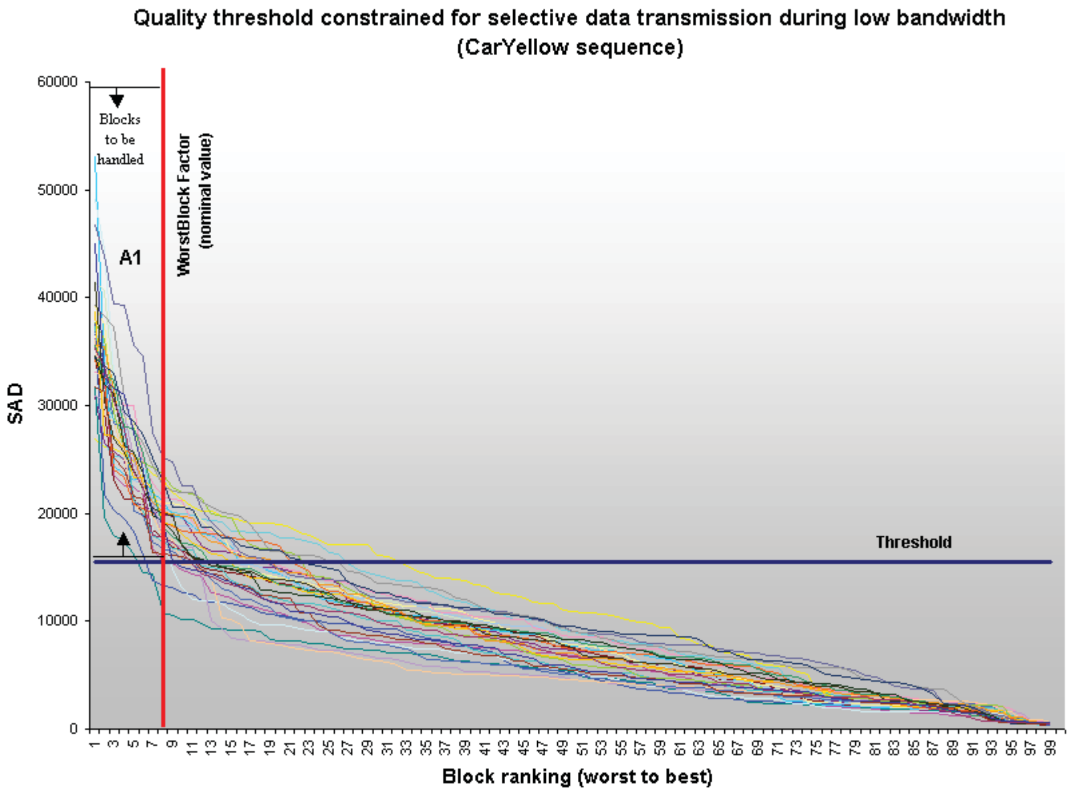
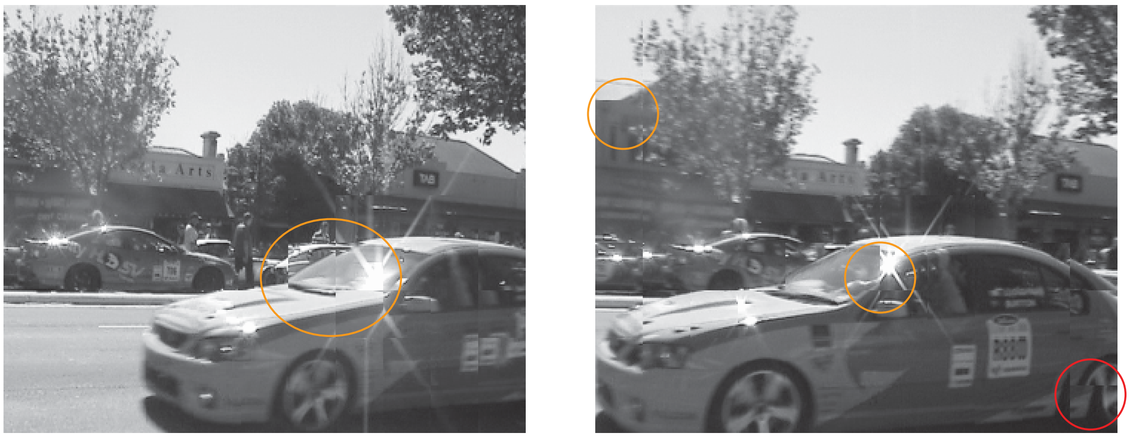


Figure 6.4 Quality threshold constrained for low bandwidth (CarYellow sequence)

As the required quality of video is low, the threshold bar, the blue line in Figure 6.4, is moved upwards so that the threshold limitations applies to only a small number of blocks. Since there are few blocks, the error resilience is limited, thereby, affecting the image quality to a less significant level.

In a low bandwidth scenario, the WorstBlock factor represented as a red line is moved very close to y-axis as they take a nominal value. The blocks in top-left hand corner, Section A1, are the worst mismatch blocks and have to be corrected. To make sure that this approach correctly identifies the worst mismatches let have a look the following outputs, Figure 6.5 and 6.6.

(a) Original Images


(b) Multiresolution motion compensated images without any error correction

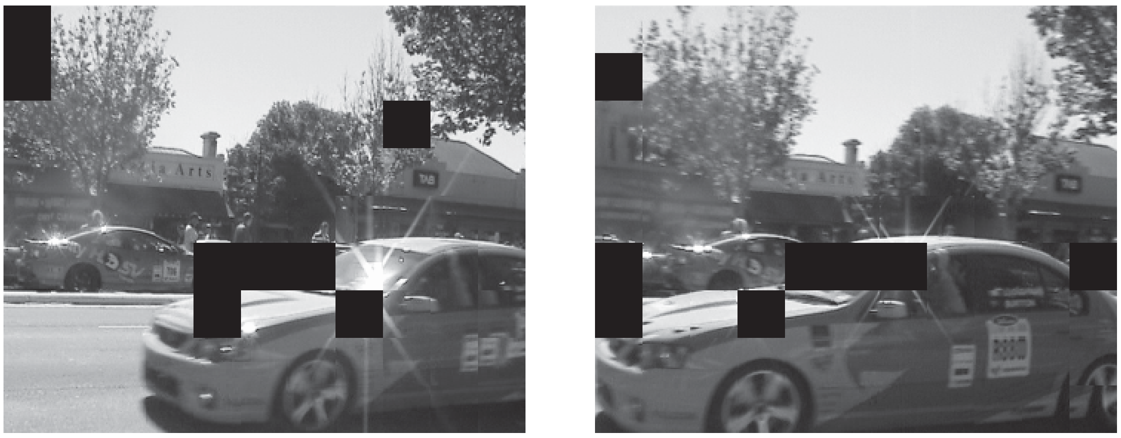

(c) Showing only blocks which are propagated without correction

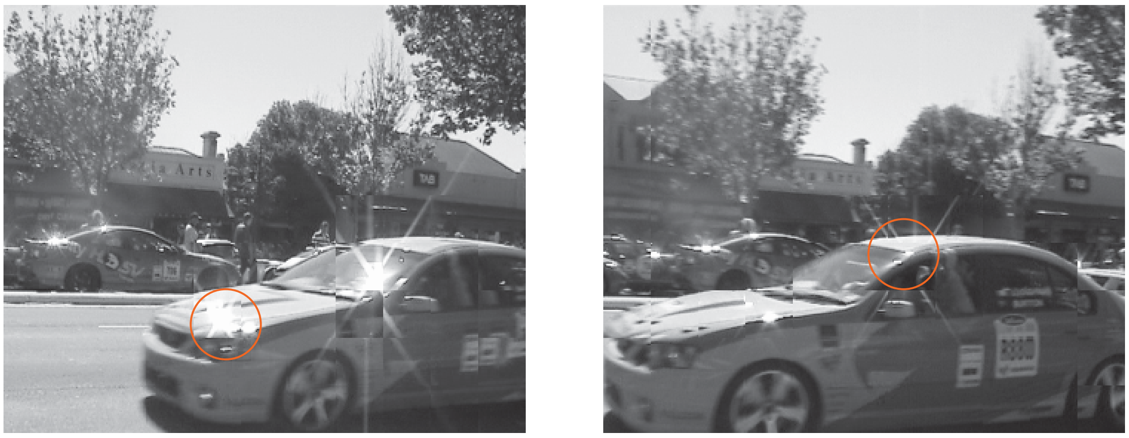
Figure 6.5: Mismatch identification during nominal Threshold and in low bandwidth scenario

In Figure 6.5, two sample frames, Frame 23 and 27, of CarYellow sequence from R2 (288x352) resolution are studied. Figure 6.5 (a) represents the original image, which has to be predicted and Figure 6.5 (b) represents the multiresolution motion estimated image but without the application of error correction. It can be seen that the estimated image is quite close to the original image except at certain places, which are highlighted by orange and red circles. The missing black coloured blocks in Figure 6.5 (c) represent the mismatches identified under this threshold mode. It clearly shows that our approach has identified almost all the mismatches shown in Figure 6.5 (b). Some of them are not identified, like the mismatch marked in red in Frame 27, since the WorstBlock factor only takes a nominal value due to the bandwidth constraint. The overall good performance of this threshold mode is attributed to the fact that the area A1 coincides with the replaced area, Section A mentioned in Section 5.2.5.

Error Handlers

The identified mismatches are now subjected to (1) block replacement and (2) motion vector replacement. The block replacement method is represented by Figure 6.6 (b), and the motion replacement by Figure 6.6 (c); the orange circles highlight the prediction error. It can be seen that in comparison with original image, Figure 6.6 (a), the mismatches are properly handled by the block replacement method better than the motion vector replacement. In block replacement method the mismatch blocks are intracoded with contributes to the accurate motion depiction, which cannot be obtained to that extent in motion vector replacement. The quality of the replaced blocks is also increased in the block replacement method. The performance comparisons of these two methods are detailed later in this chapter.

(a) Original Images


(b) Block replacement

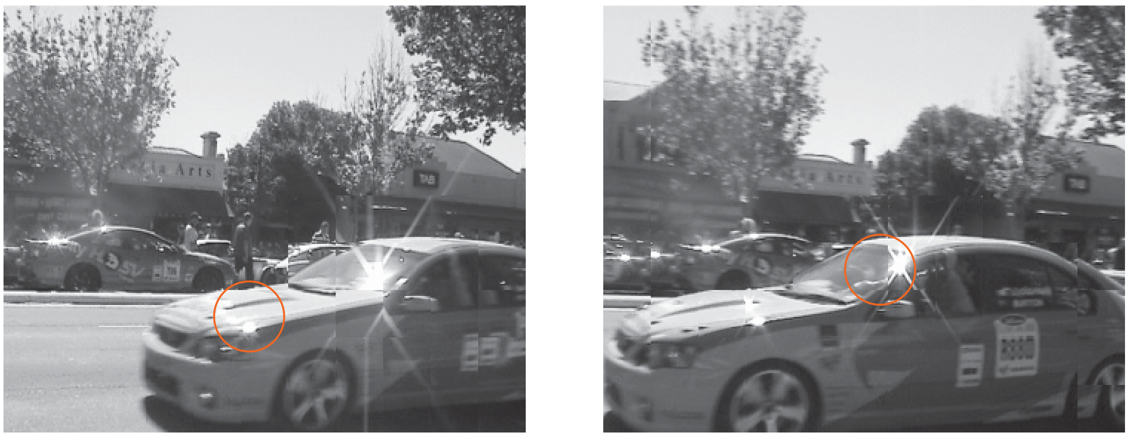

(c) Motion vector replacement

Figure 6.6: Updated blocks using block and motion vector replacement method in Threshold mode 1 (CarYellow sequence).

SwanAndBird Sequence

A different sequence, SwanAndBird sequence, was now considered. This is to test the behavior of our approach in different motion sequences. Compared to the previous sample video, the SwanAndBird exhibits a slow motion.

Here, the Threshold parameter takes a nominal value and the mismatch blocks that are under this threshold limit is represented graphically in Figure 6.7. This mismatch count is quite similar to the count obtained in CarYellow sequence.

As in this test case we have a low bandwidth limitation, the WorstBlock factor also takes an appropriately low value. The threshold plot for this sequence is represented in Figure 6.8. As seen in the Figure, the global motion of this sample sequence is reasonably similar to the previous sequence and their threshold plots match closely as well. The area that falls under both these two parameters is marked as A1S. The blocks in this section alone are subjected to error correction.

Figure 6.9 represents two sample frames from SwanAndBird sequence with the identified mismatch blocks. Since this sequence has little motion, all the mismatches are correctly identified in this threshold mode. The identified mismatches are handled using block and motion vector replacement and their sample outputs are shown in Figure 6.10; the square indicate the prediction error.

The block replacement method, Figure 6.10 (b), once again outperforms the motion vector replacement method, Figure 6.10 (c). Thus the estimated image is almost identical to the original image. This demonstrates that it is possible to achieve high precision images even in the nominal level.

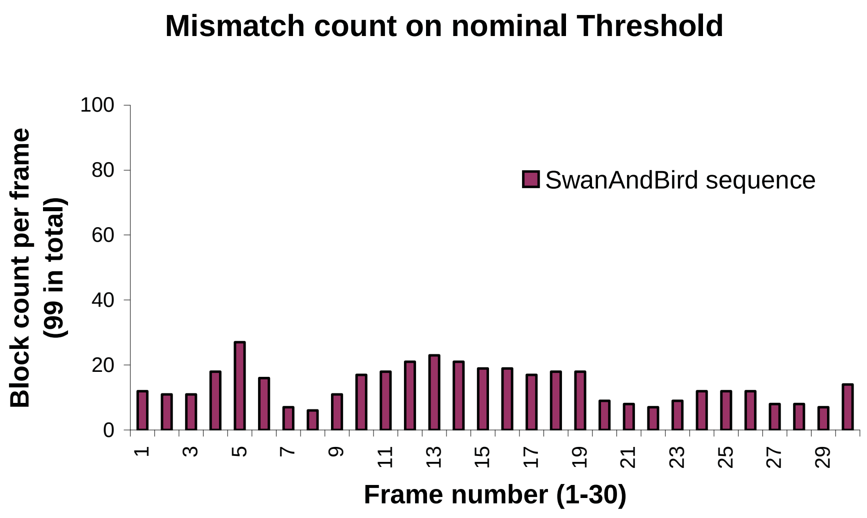


Figure 6.7: Number of blocks identified as mismatches using a nominal Threshold (SwanAndBird sequence)

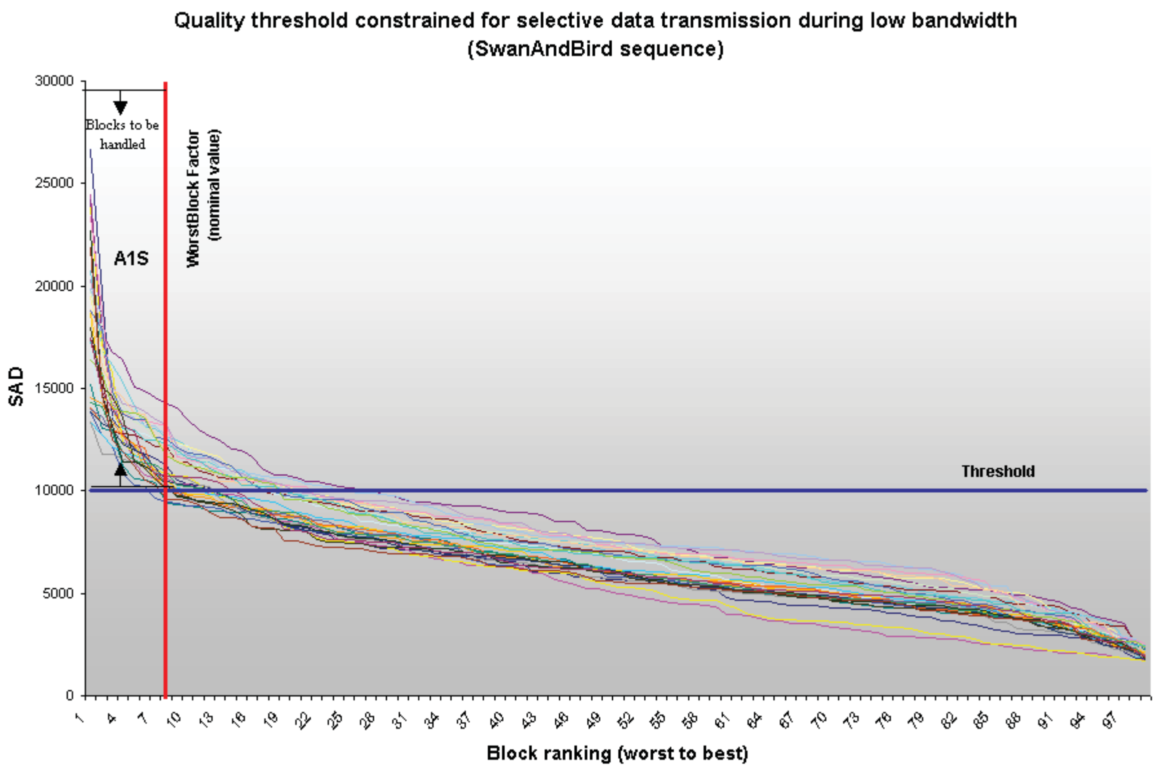
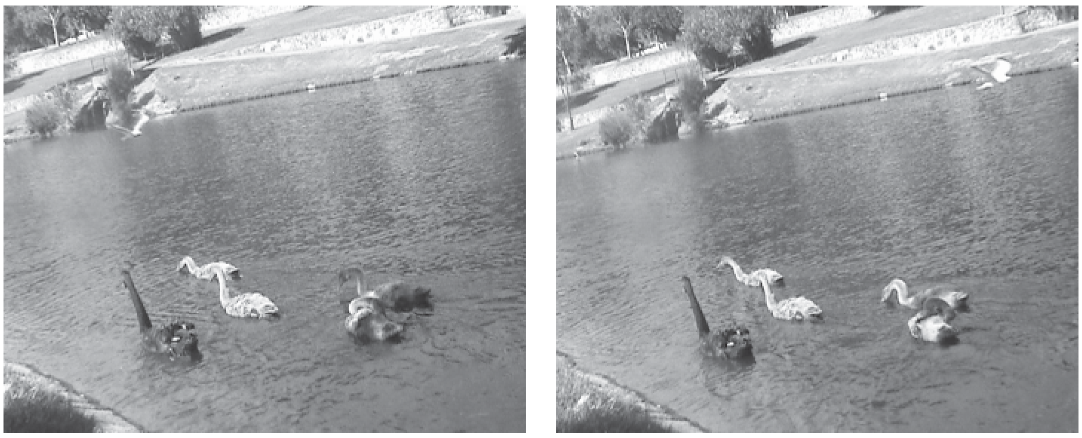
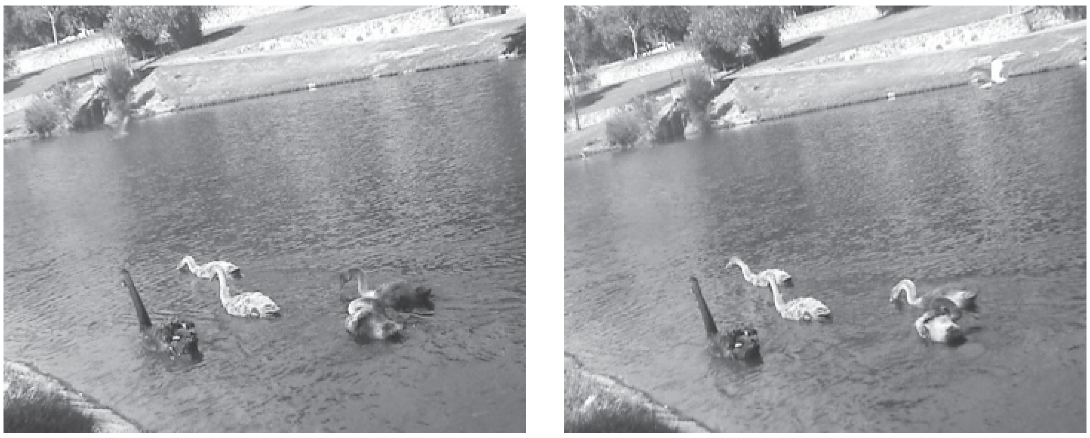


Figure 6.8: Quality threshold constrained for low bandwidth (SwanAndBird sequence)


(a) Original Images


(b) Multiresolution motion compensated images without any error correction

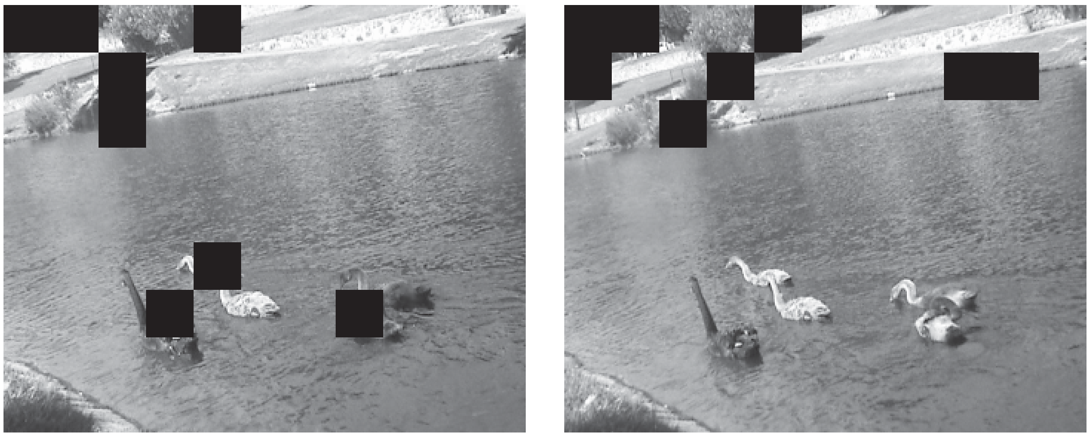
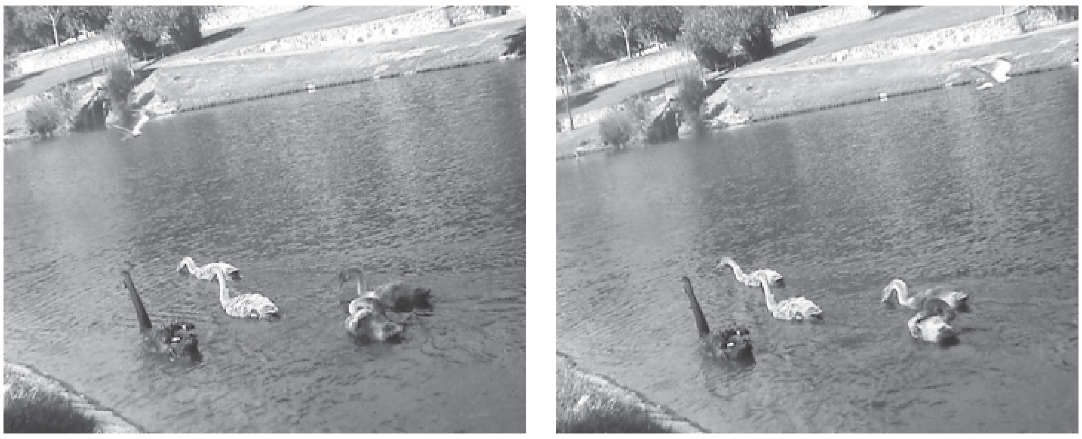
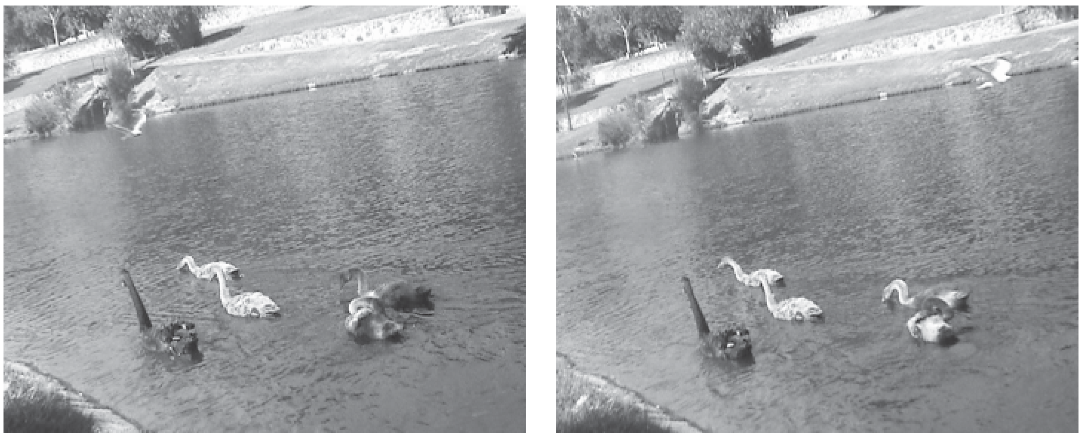

(c) Showing only blocks which are propagated without correction

Figure 6.9: Mismatch identification during nominal Threshold and in low bandwidth scenario (SwanAndBird sequence)


(a) Original Images


(b) Block replacement

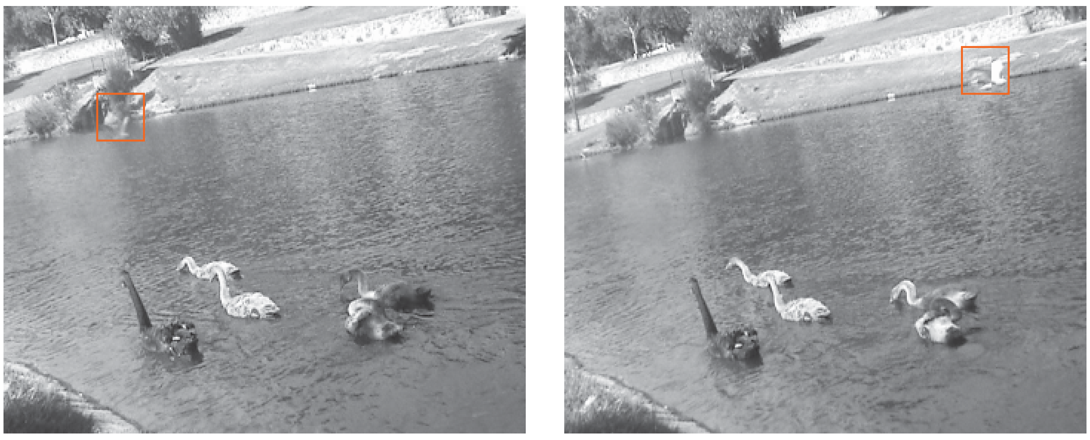

(c) Motion vector replacement

Figure 6.10: Updated blocks using block and motion vector replacement method in Threshold mode1 (SwanAndBird sequence)

6.2.2 Quality Threshold Constrained for High Bandwidth

This test case applies to the situation where there is no limitation to the available bandwidth but full quality video is not justified. The Threshold, which controls the quality of the frames, is maintained at a nominal value while the WorstBlock factor is stretched to the maximum limit as it represents the bandwidth condition.

The threshold plot of this test case for CarYellow sequence is shown in Figure 6.11. Since there is no bandwidth, the replacement area A2 can now be extended to include Section B, of the threshold plot shown in Figure 5.17, in addition to Section A. So, the entire mismatch blocks in area A2 are selected for error handling.

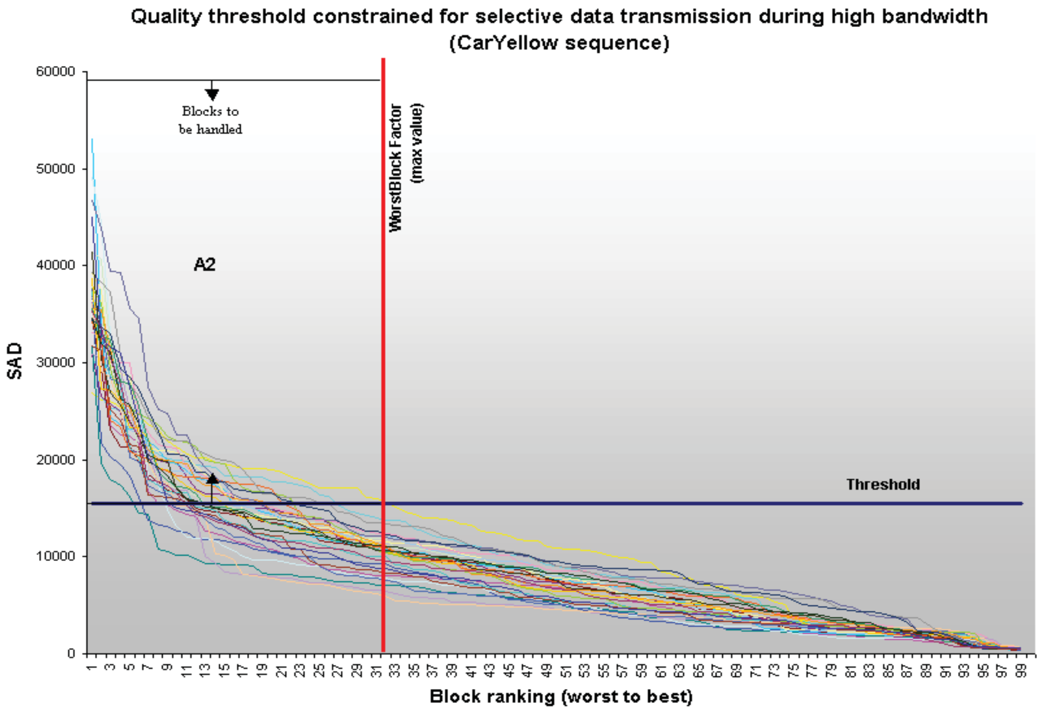
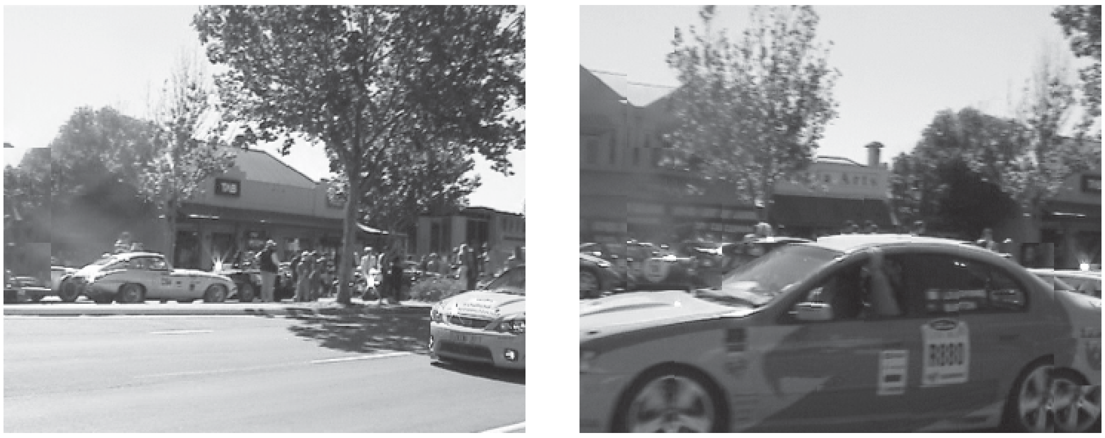


Figure 6.11: Quality threshold constrained for high bandwidth
(CarYellow sequence)

It can be seen from Figure 6.12 (c) that there is a steady increase in the number of mismatch blocks due to the increase in WorstBlock factor. This helps to have a better error resilient image, even this would not, at times, be sufficient to produce an error free image because few of the blocks would be below the threshold line. This mainly depends on the type of motion exhibited by the video sequence. For a slow motion video, as in the SwanAndBird sequence, this is more than sufficient to cover all the mismatches.

(a) Original Images


(b) Multiresolution motion compensated images without error correction

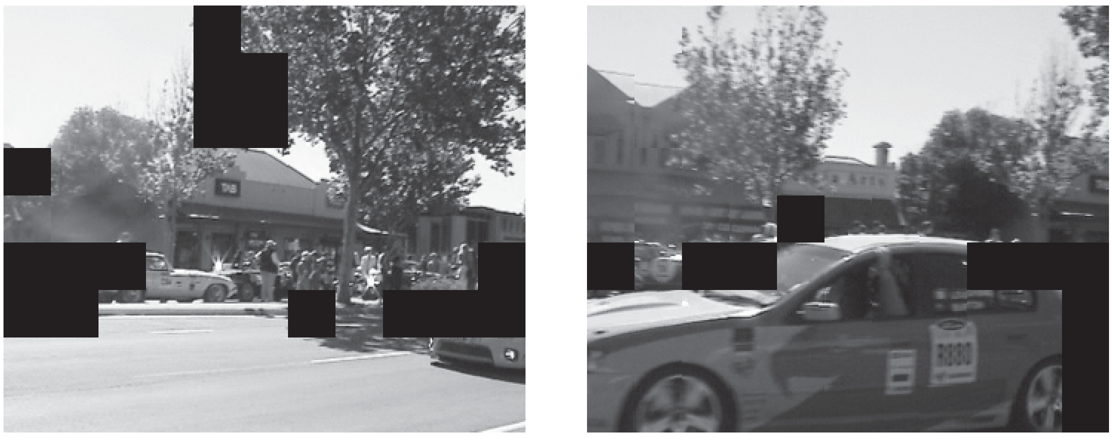
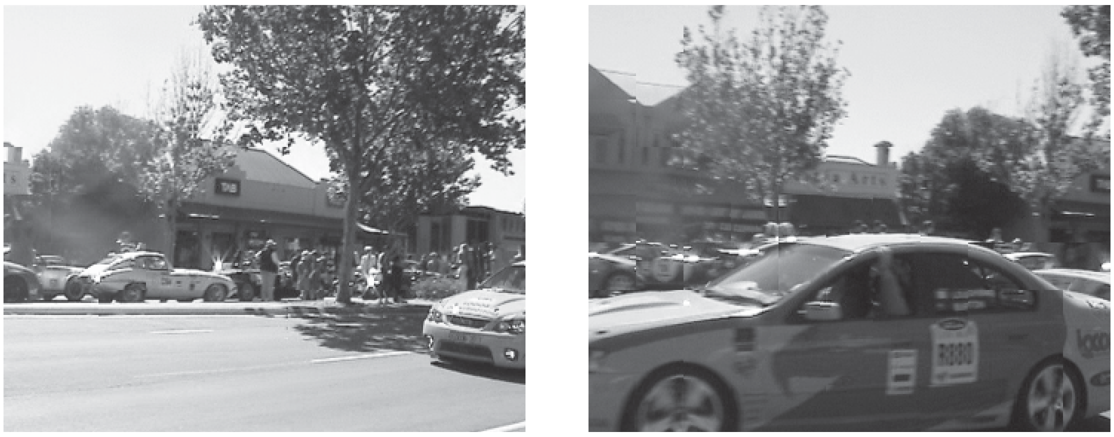

(c) Showing only blocks which are propagated without correction

Figure 6.12: Mismatch identification during nominal Threshold and in high bandwidth scenario (CarYellow sequence)

(a) Original Images


(b) Block replacement

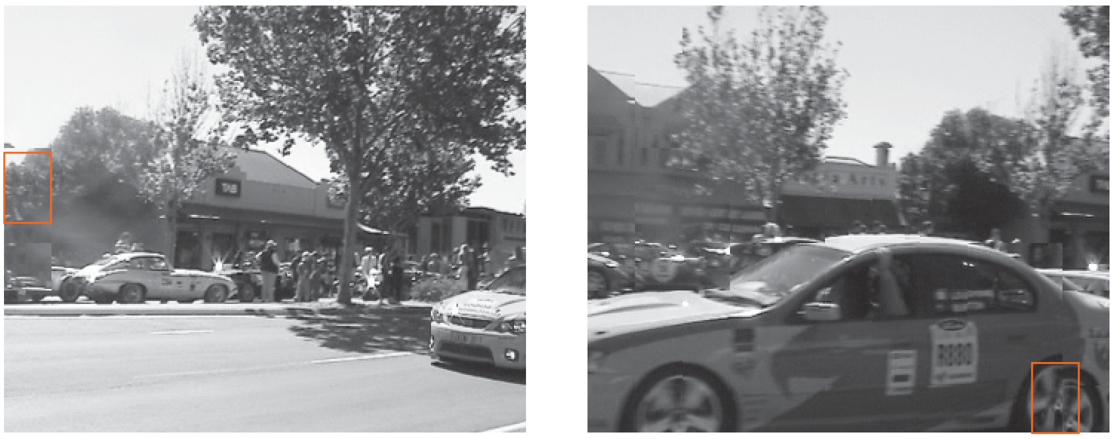

(c) Motion vector replacement

Figure 6.13: Updated blocks using block and motion vector replacement method in Threshold 2 (CarYellow sequence)

Error Handlers

Figure 6.13 shows sample images of CarYellow sequence for the evaluation of the block and the motion vector replacements of this test case. Comparing the two corrected images against the original image it is clear that block replacement method does a better job of error correction than motion vector replacement alone.

SwanAndBird Sequence

Applying the same test case environments of nominal Threshold and maximum WorstBlock factor to SwanAndBird sequence we obtain the threshold plot shown in Figure 6.14. Similarly in order to put up with higher bandwidth situation the replacement area A2S now accommodates the combined area of A and B, of the threshold plot (Figure 5.17). Therefore the number of blocks to be handled increases as seen Figure 6.15 (c).

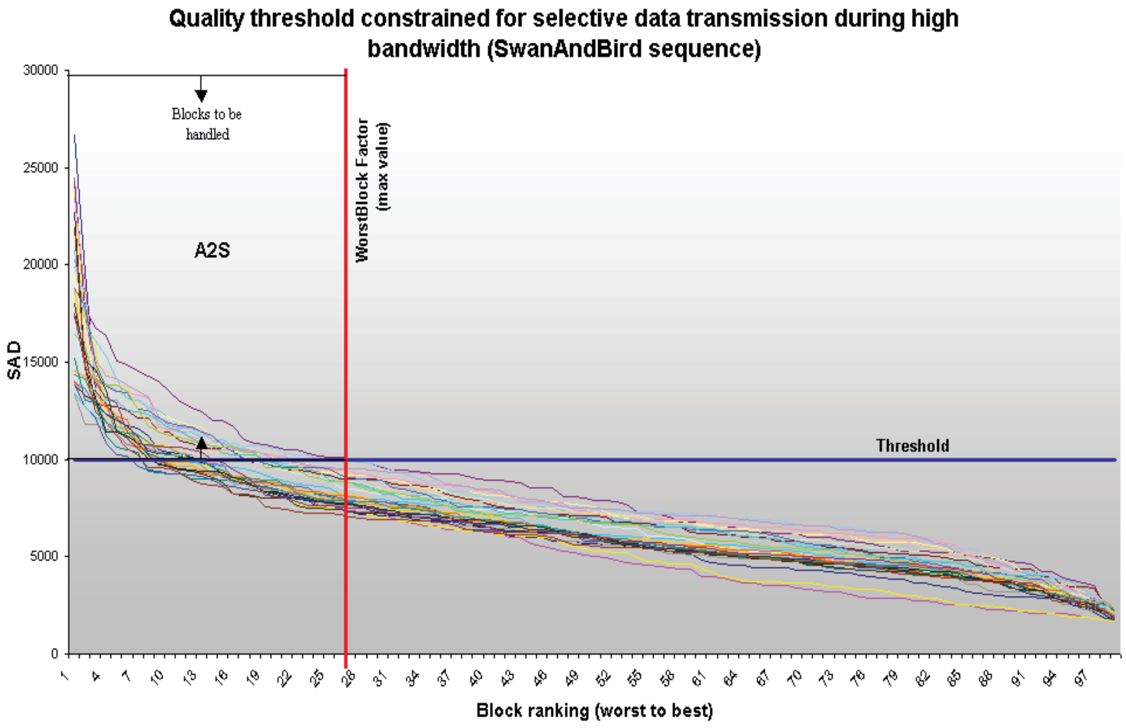
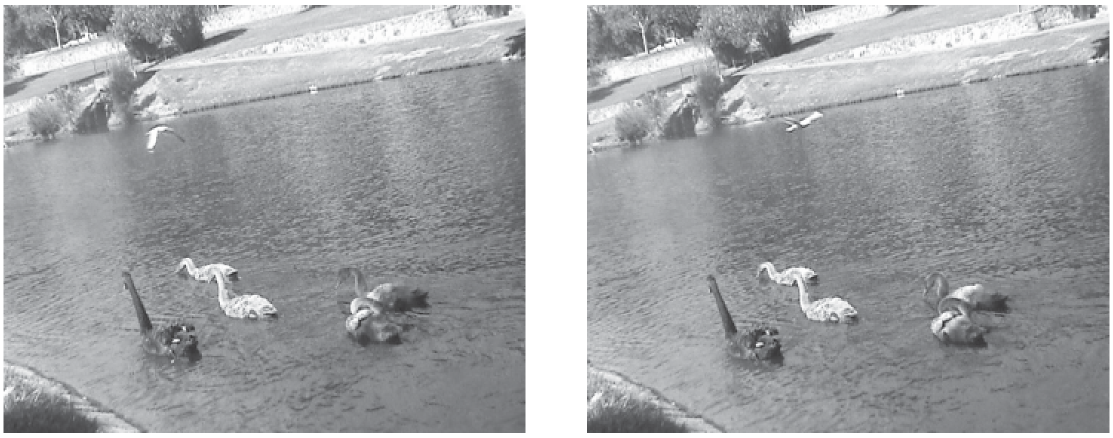
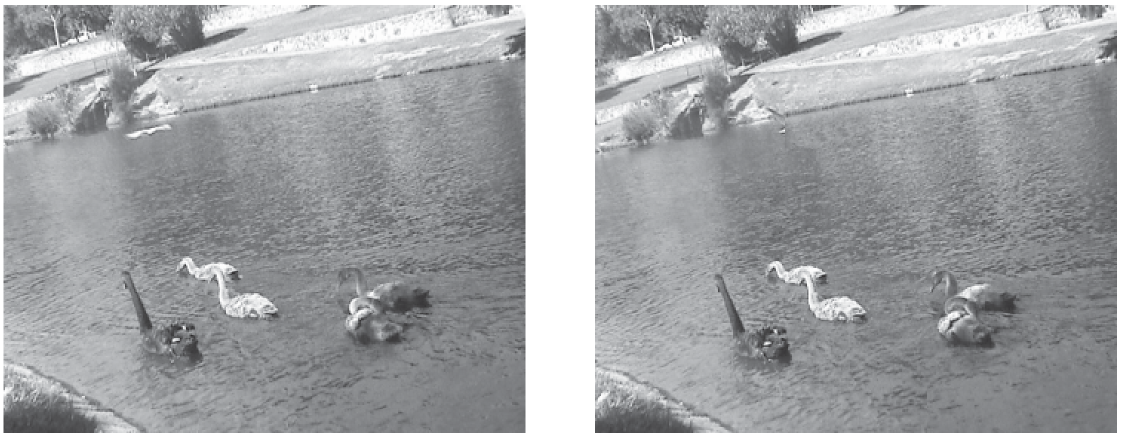


Figure 6.14: Quality threshold constrained for high bandwidth (SwanAndBird sequence)

As this sequence exhibits only few motion changes, all the mismatches are contained in the replacement area A2S. The block and motion vector replacement for this sequence are presented in Figure 6.16. Here again, the mismatches are correctly handled by the block replacement approach.


(a) Original images


(b) Multiresolution motion compensated image without error correction

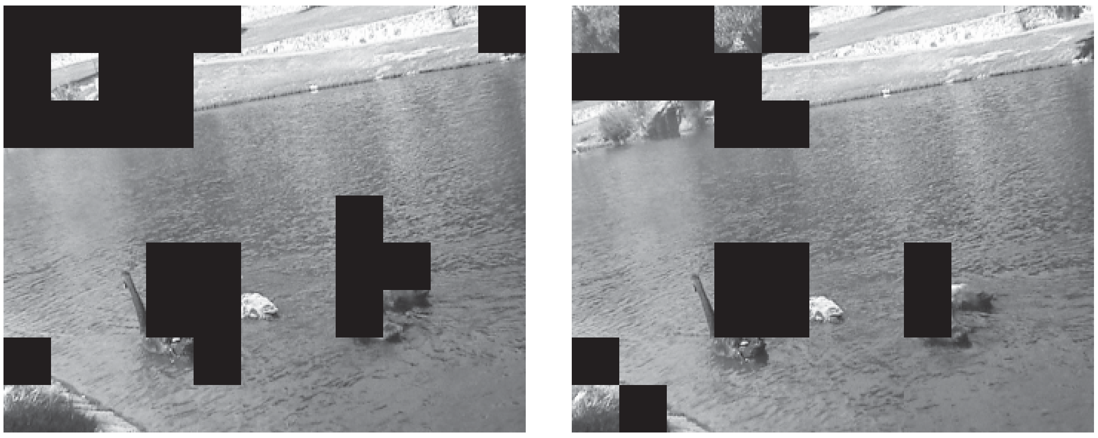
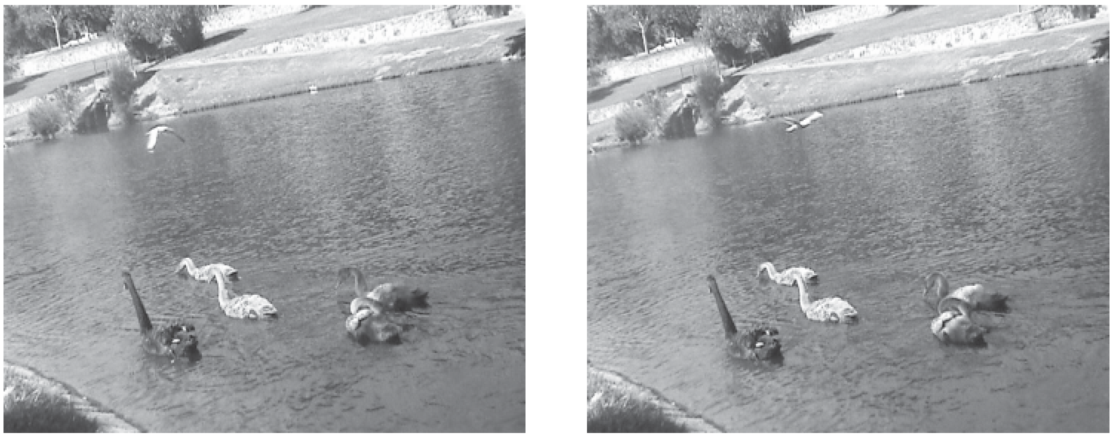
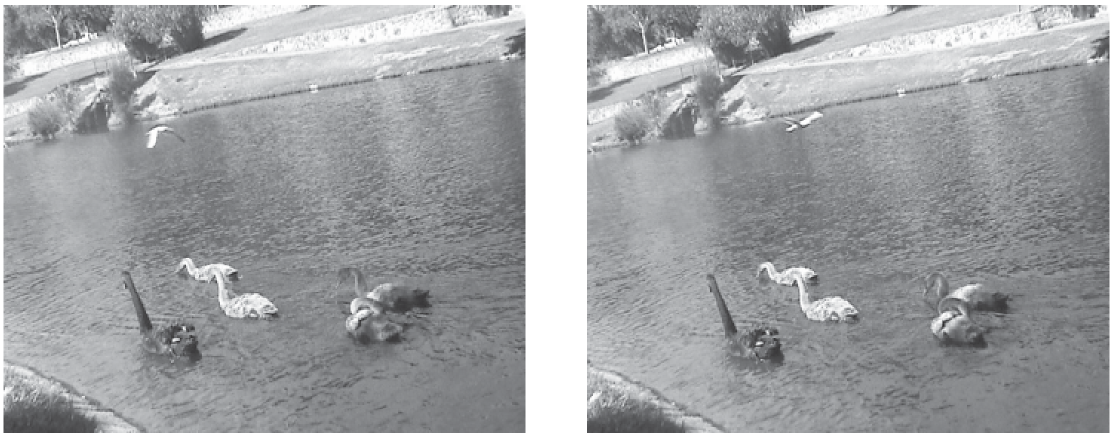

(d) Showing only blocks which are propagated without correction

Figure 6.15: Mismatch identification during nominal Threshold and in high bandwidth scenario (SwanAndBird sequence)


(a) Original images


(b) Block replacement

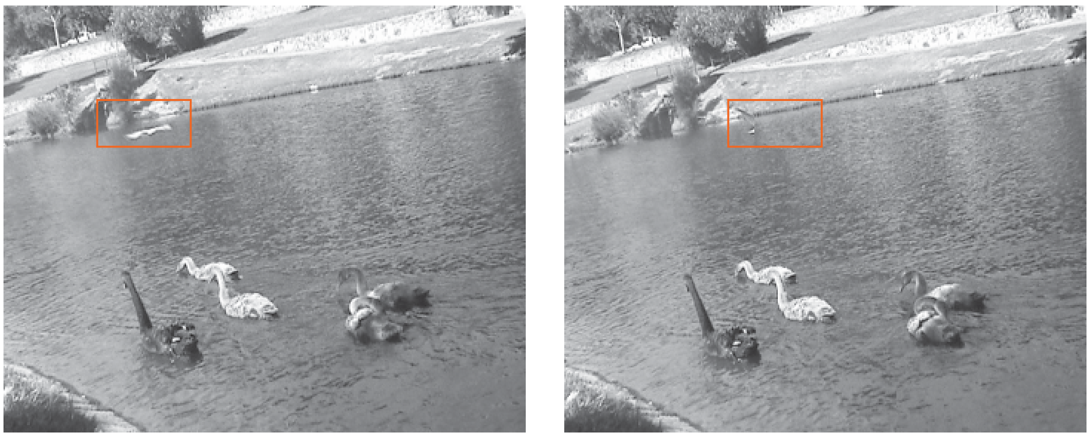

(c) Motion vector replacement

Figure 6.16: Updated blocks using block and motion vector replacement method in Threshold 2 (SwanAndBird sequence)

6.2.3 Quality Threshold Relaxed for High Quality and High Bandwidth

This test case is used when video is requested from a high-end device like HDTV through a high-speed network. In order to satisfy these demands, the Threshold parameter is shifted to a lower value to include as many blocks as possible and the WorstBlock factor takes the highest possible value. The corresponding threshold plots of both the sequences are represented in the Figure 6.18 and 6.19.

As the Threshold limit is pushed to a lower value, the replacement area, A3 and A3S, now include whole of Section C and majority of Section D. With the WorstBlock factor taking a higher value, Section A and B is also included in the replacement area. So the block count reaches a high value which nearly amount to 80% of the total frame. The block count for the two sequences is shown in Figure 6.17.

All the blocks in the replacement area, which are to be replaced, are highlighted in Figure 6.20 (c) and 6.22 (c). Even though it can be seen that not all the blocks are erroroneous blocks, by comparing the original and predicted images, they are selected for replacement due to the need for high-quality video images and also because it can be supported by a higher bandwidth network.

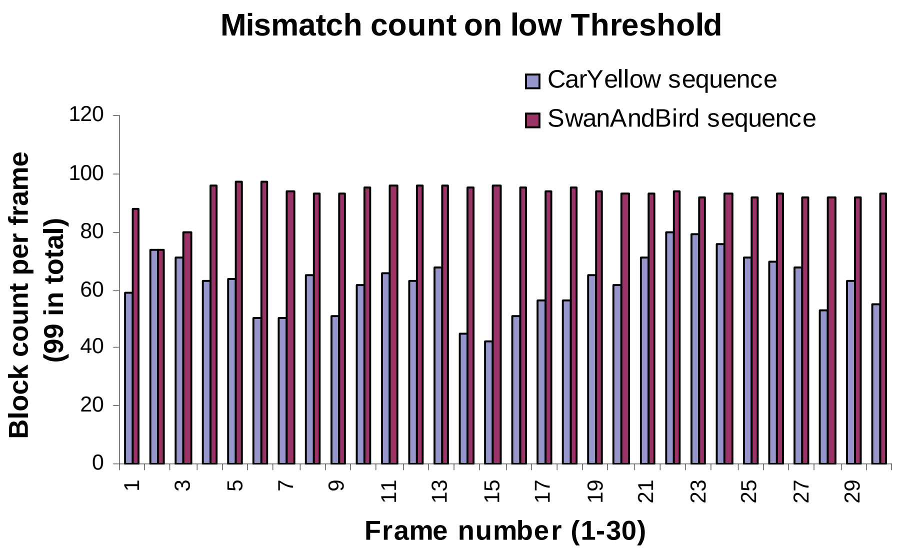


Figure 6.17: Number of blocks identified as mismatches using a low Threshold in both the sequence

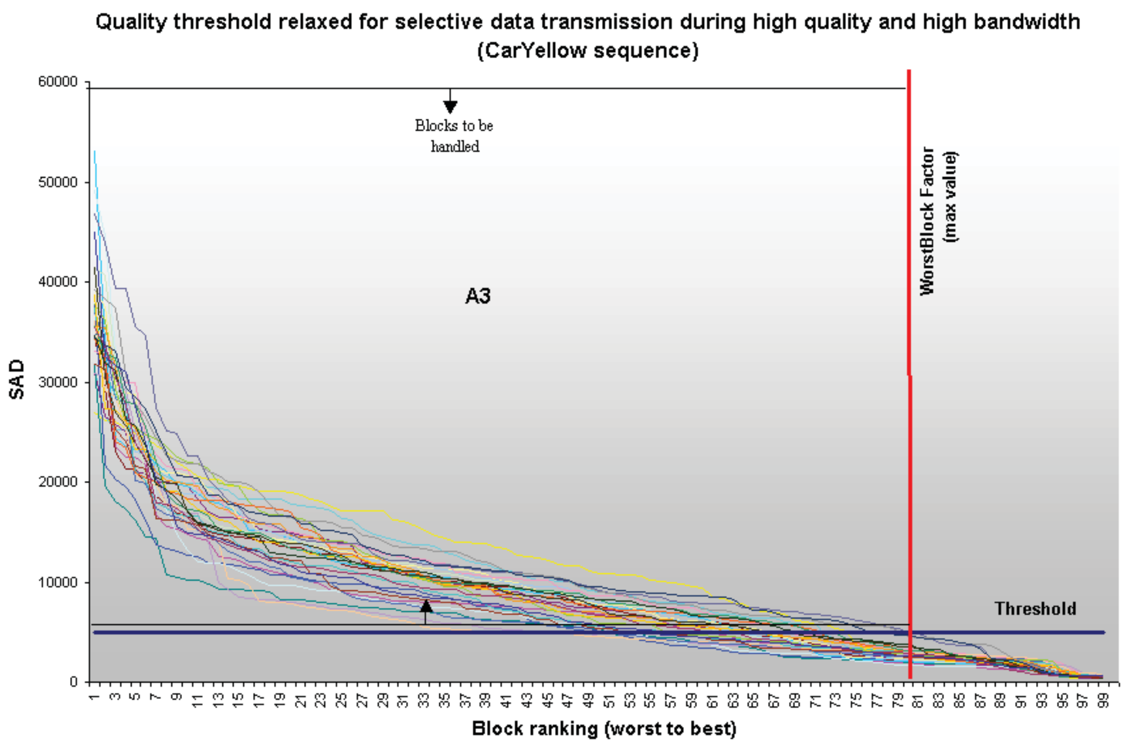


Figure 6.18: Quality threshold relaxed for high quality and high bandwidth (CarYellow sequence)

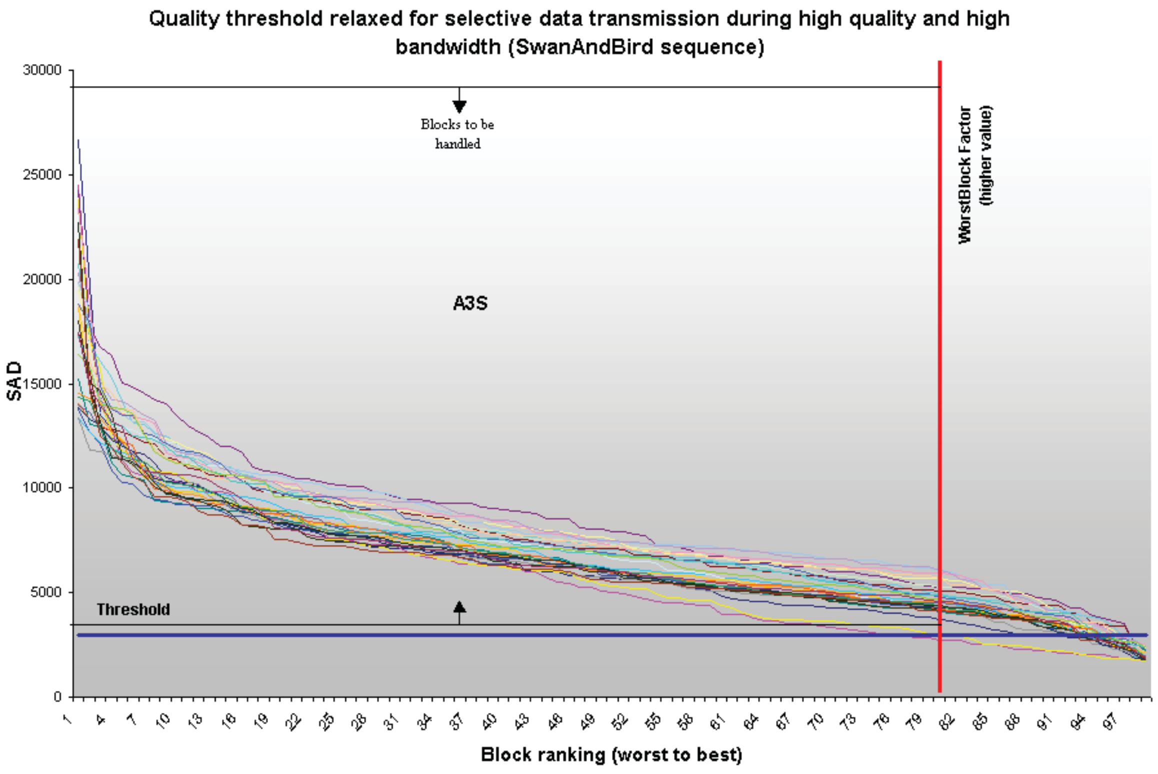
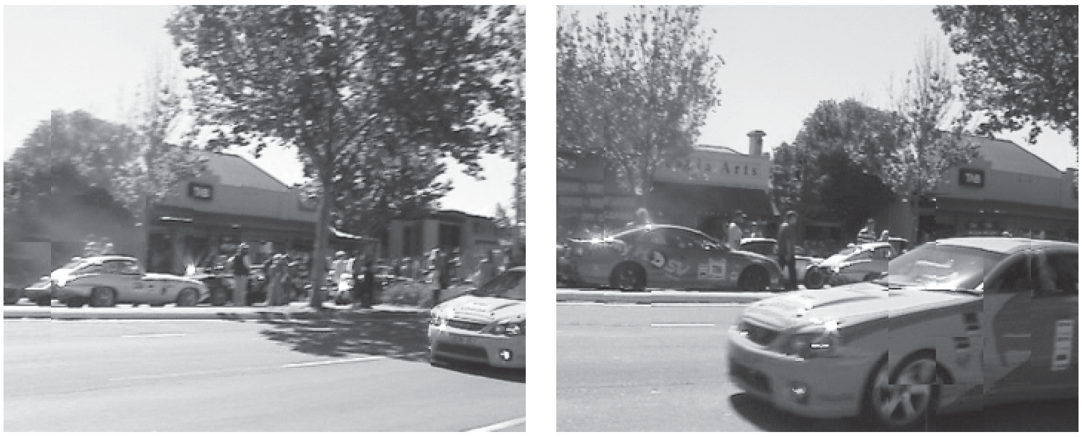
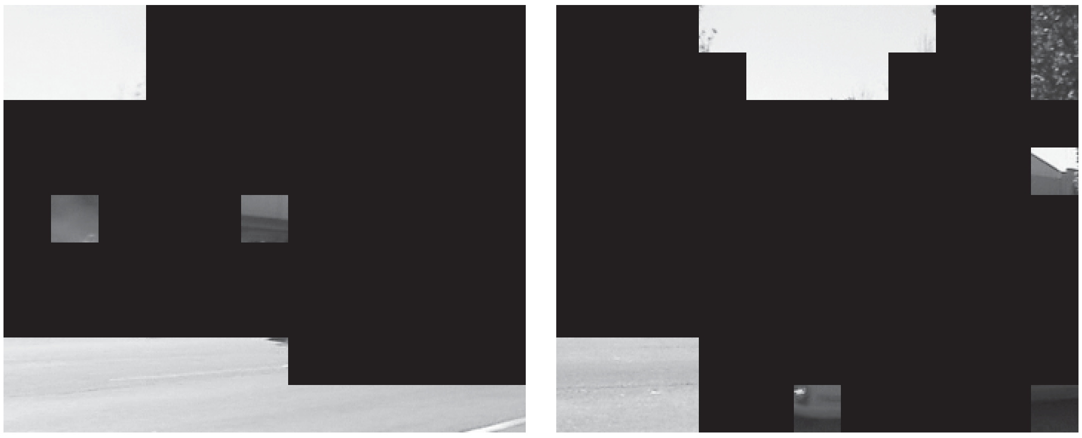


Figure 6.19: Quality threshold relaxed for high quality and high bandwidth (SwanAndBird sequence)

(a) Original images



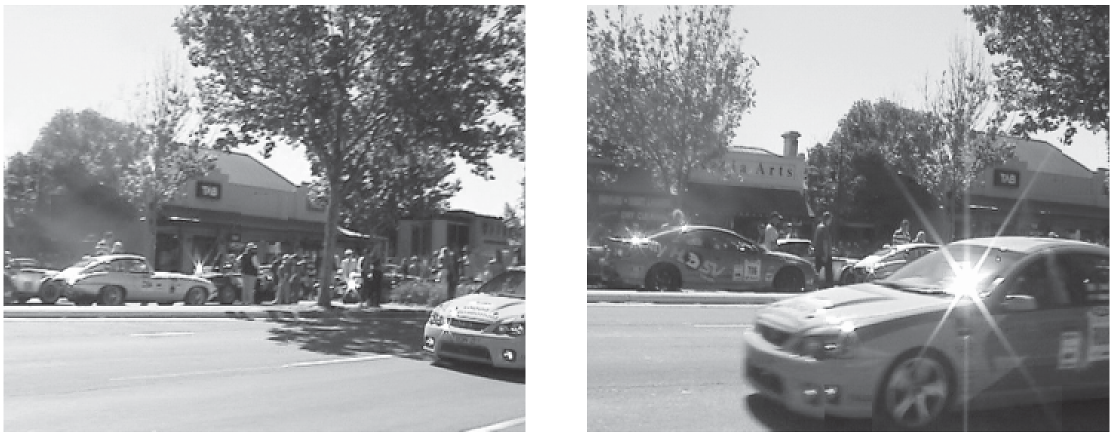
(b) Multiresolution motion compensated image without error correction



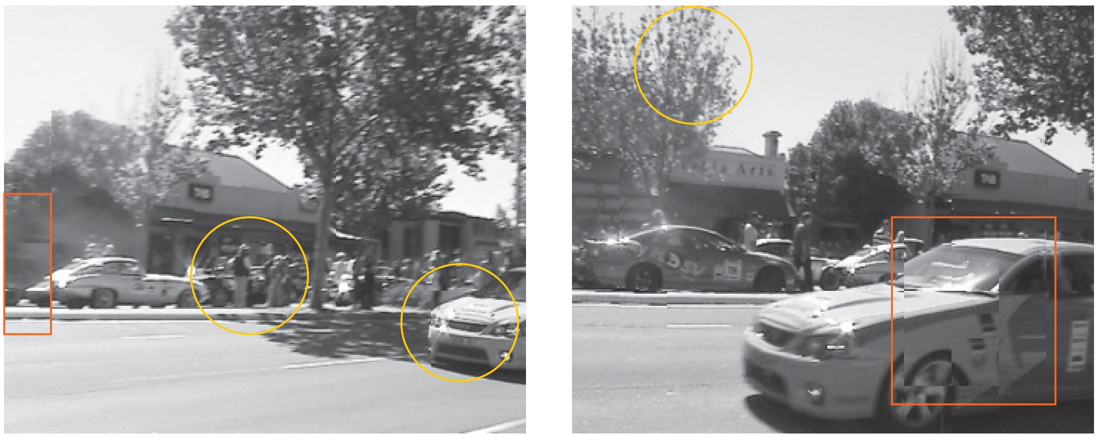
(c) Showing only blocks which are propagated without correction

Figure 6.20: Mismatch identification during low threshold and in high bandwidth scenario (CarYellow sequence)

(a) Original images

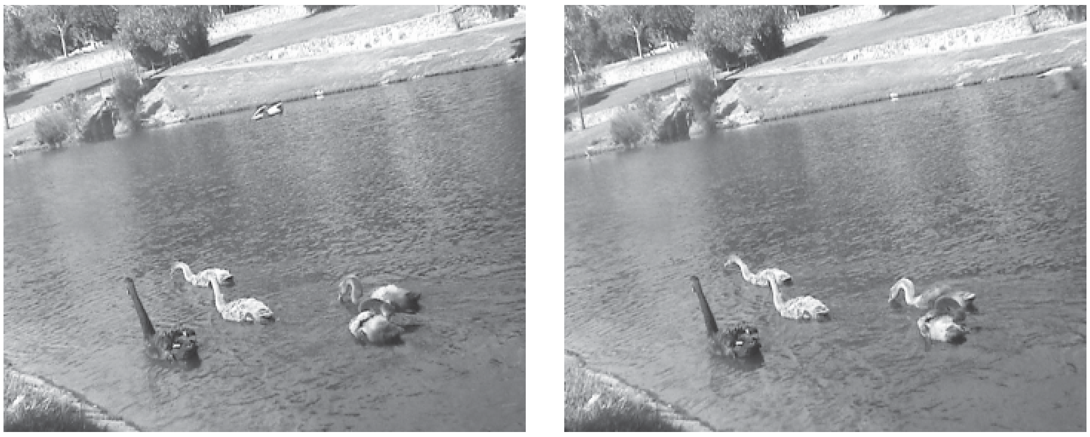


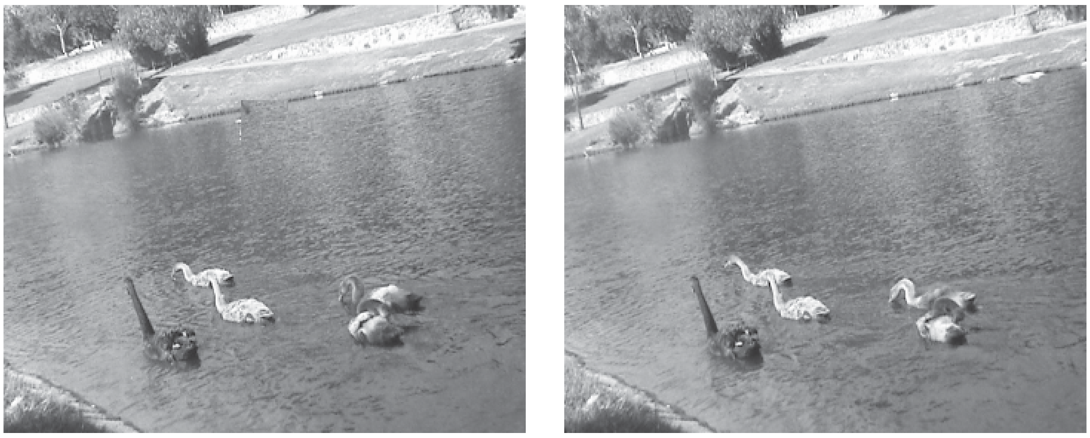
(b) Block replacement



(c) Motion vector replacement

Figure 6.21: Updated blocks using block and motion vector replacement method in Threshold mode 3 (CarYellow sequence)


(a) Original images


(b) Multiresolution motion compensated image without error correction

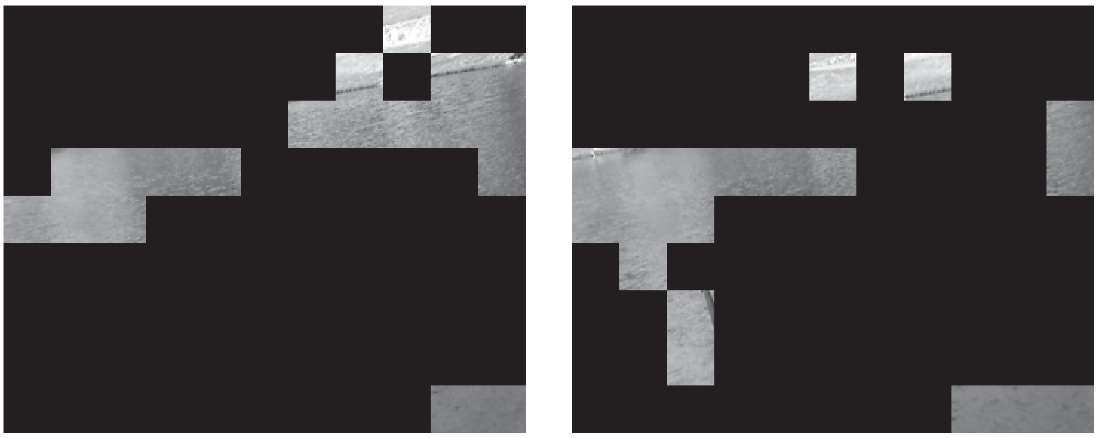
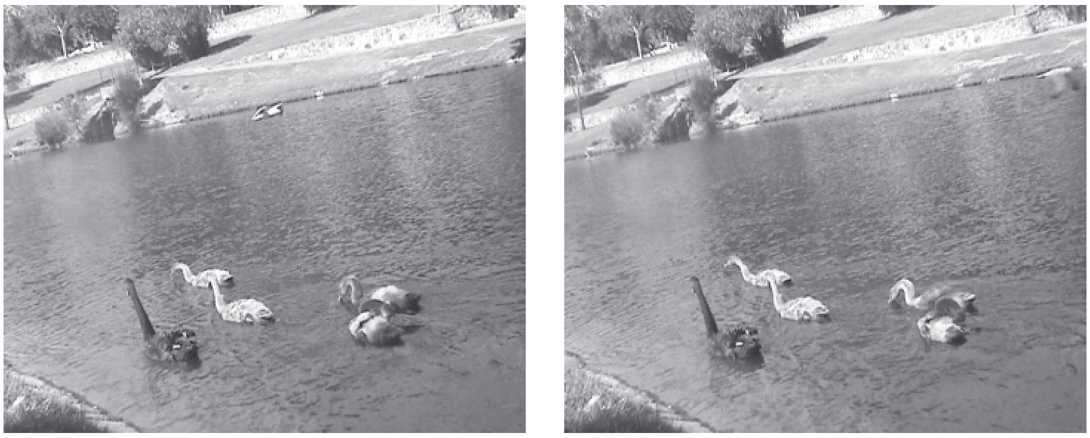
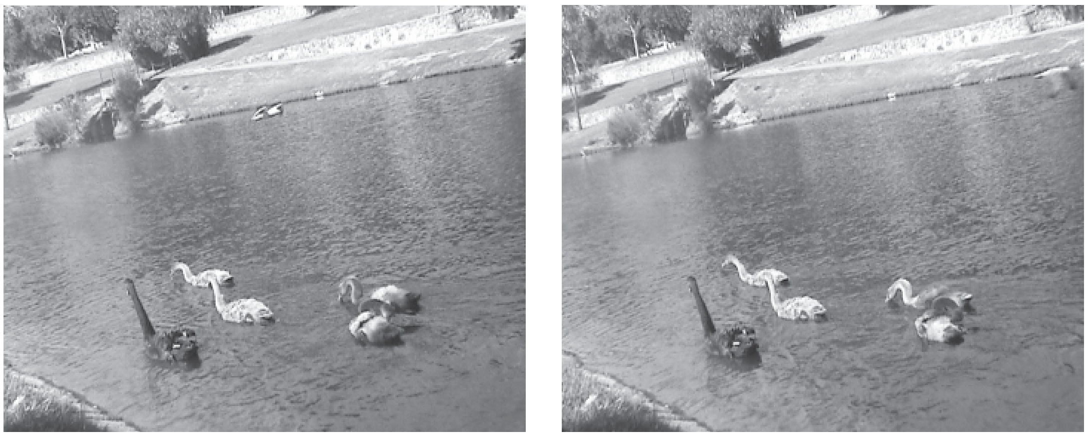

(c) Highlighting error blocks in motion compensated images

Figure 6.22: Mismatch identification during low threshold and in high bandwidth scenario (SwanAndBird sequence)


(a) Original images


(b) Block replacement

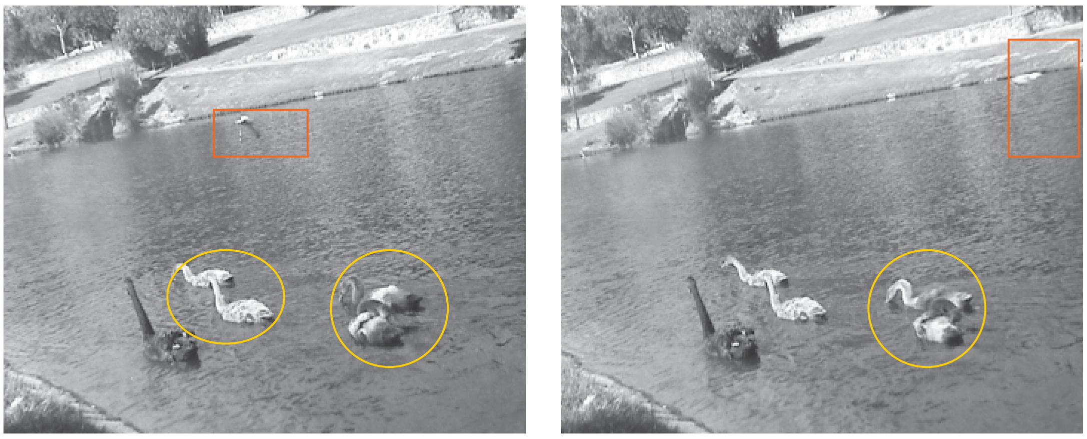

(c) Motion vector replacement

Figure 6.23: Updated blocks using block and motion vector replacement method in Threshold mode 3 (SwanAndBird sequence)

The blocks of A3 and A3S are subjected to block and motion vector replacement Figure 6.21 and 6.23; the yellow circles indicate the quality variation. These blocks are appropriately enhanced by block replacement method compared to vector replacement as in all the previous test cases. In addition to correcting their motion, the block replacement method also improves the quality of the block. The quality enhancement can be easily spotted in this test case due to the replacement of a large number of blocks.

6.3 Bandwidth Estimation

Using wavelet decomposition the video frames were decomposed into three different resolutions, R1 (144x176), R2 (288x252) and R3 (576x704). This enables to adapt to different bandwidth requirements. For example, using R1 in lower bandwidth situation and using R3 when the bandwidth constraint is relaxed. The decomposed images are further subjected to multiresolution motion estimation using variable block sizes (Section 5.1.2.4.1). Hence, we have the following based on the resolution level.

Table 6.1: Size of macroblocks in different resolutions

Frame Resolution	**Macroblock Size**
R1 (144x176)	16x16
R2 (288x352)	32x32
R3 (576x704)	64x64

The multiresolution motion estimation used applies *top-down* approach for estimating the initial motion estimation. In addition further scaling is done, as per our proposed approach, based on the bandwidth availability and the required video quality.

The *first approach (motion vector replacement)* is to resend new motion estimate for the mismatches. For each mismatch macroblock one motion vector is resend, which is estimate by repeating the motion estimation process. A motion vector which represents by 2 signed integer/fractions uses 16 bits. Even though it only requires 16 bits, it incurs high computation expense. As the motion estimation process has to be repeated again for establishing the new vectors after the initial top-down multiresolution motion estimate.

The motion vector V(h,v) represents the horizontal motion (h) as signed integer in the range of {-p …0…+p} and the vertical motion (v) in the range of {-p,…0 … +p}, where S is the search range (±p). It may also use fraction if the motion estimation algorithm uses half pixel accuracy approach or similar. If a picture P comprises of 16x16 bocks, for each block (x,y) the motion estimate would be (h, v) and the prediction would result in the form of P(x+h, y+v). Generally the motion vector overhead is large for very small blocks. Although this factor becomes negligible as 0.031 and 0.0078 bits/motion vector for larger blocks like 16x16 and 32x32 respectively (Arnold *et al.* 2007).

Table 6.2: Motion vector estimate in bit representation

	No. of bits required	**Bits/motion vector**	**Total bits**
Motion Vector	2	8	16 bits

The *second approach (block replacement)* involves performing macroblock intra-coding. This requires resending the original macroblock as a replacement for the ones in error, thereby having no computational overhead. As variable-block size approach is used, the different resolutions require their respective size of blocks to be resent. The bits per pixel (bpp) also vary based on the compression ratio. A typical macroblock can use up to 0.5 to 2.0 bpp based on wavelet or DCT compression at the rate the 4:1 and 16:1 respectively. The following outlines the total bits required per block based on the resolution and compression ratio.

Table 6.3: Macroblock estimate in bit representation

Resolution	**Block Size**	**bpp (based on compression ratio)**	**Total bits per macroblock**
R1 (144x176)	16x16	0.5 – 2.0	128 – 512 bits
R2 (288x352)	32x32	0.5 – 2.0	512 – 2k bits
R3 (576x704)	64x64	0.5 – 2.0	2k – 8k bits

Applying the above motion vector and block estimate to the CarYellow frames of resolution R1 (144x176), we have

Initial motion vector estimate

= Total block count * bits per motion vector (6.1)

In our motion vector replacement approach,

Motion vector resend estimate (in bits)

= Mismatch count * bits per motion vector (6.2)

Therefore,

Total Motion vector estimate

= Initial motion vector estimate + Motion vector resend estimate (6.3)

In our block replacement approach,

Block resend estimate (in bits)

= Mismatch count * block size * bpp (6.4)

Total block estimate

= ((Total block count – Mismatch count) * bits per motion) + block resend estimate

(6.5)

When implementing for the bandwidth constraint and relaxed scenarios as in Section 6.2.1 and 6.2.3 respectively we have the following results. During bandwidth constraint situation we have low mismatch count and the mismatch count is high for relaxed bandwidth.

Table 6.4: Motion vector and block estimate for different bandwidth requirements

Bandwidth requirement	**Average mismatch count per frame**	**Total Motion Vector estimate (in bits)**	**Total Block estimate in bits (@ 1 bpp)**
Constraint bandwidth (section 6.2.1)	15	1.8 k	5 k
Relaxed bandwidth (section 6.2.3)	60	2.5 k	16 k

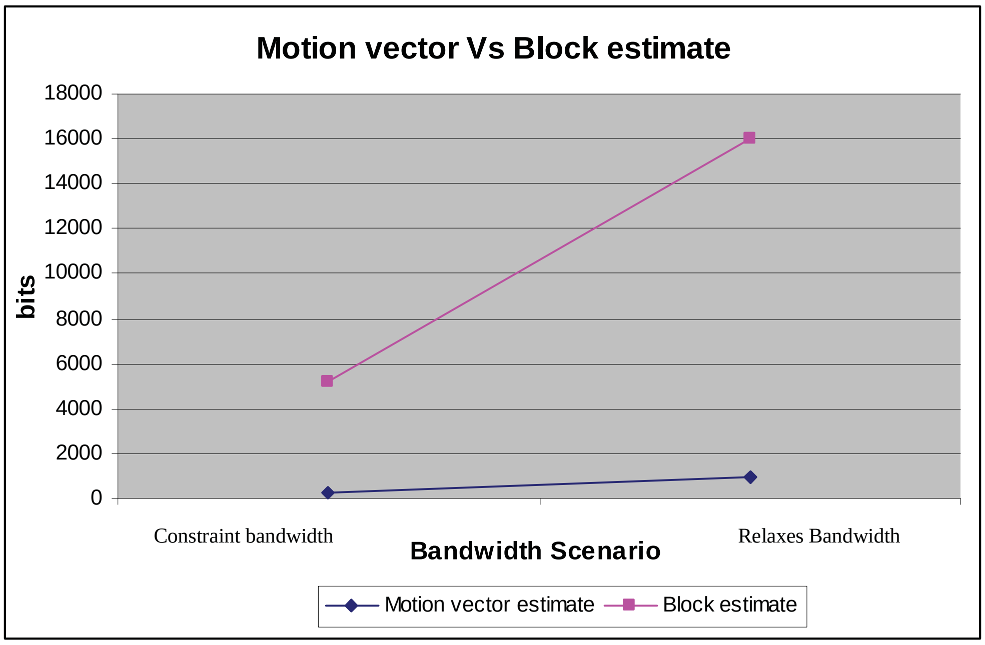


Figure 6.24: Motion vector estimate versus Block estimate in bit representation

Even though motion vector replacement method uses fewer bits in both the scenarios, it requires high computation cost due to repetition of motion estimate. Also, the quality of the frames produced is not good in comparison with the block replacement method Section 6.5.1. Although block replacement method uses more bits it is well within the bandwidth constraints. This method also has good results in terms of visual quality making it the adopted approach.

6.4 Computational Complexity

A brief insight into the computational complexity of the proposed algorithm is presented here. This thesis proposes two different scalable approaches to scale video to different bandwidth and quality requirements. Some of the techniques used in the proposed algorithm are computationally inefficient. This was done because the thesis attempted only to demonstrate the core concept behind the proposed scalable approach and not to produce a cost-effective algorithm. Therefore, it would be inappropriate to do a comparative study of the proposed algorithm against existing codecs, many of which are proprietary.

The different components involved in the proposed algorithm are outlined in Section 5.2 and 5.1 as per the Figure 5.13. The proposed algorithm uses wavelet transform for multi-level content representation as it provides better performance than discrete cosine transforms (Woods and O'Neil 1986, Vetterli 1984, Mallat 1989a, and Xiong *et al.* 1999). The type of wavelet used was Haar for its simplicity and it can be replaced by biorthogonal or Daubechies wavelets for improved performance (Topiwala 1998). Since 3D wavelets are computationally complex, 2D wavelet was adopted in the proposed approach.

The motion estimation in the proposed approach is done by using block-matching scheme for its simplicity. This can be replaced by a more efficient approach such as the pixel or frequency based approaches. In addition, the motion estimation search uses the computationally expensive exhaustive search for producing accurate results. This search mechanism can be optimised by using a cost-effective fast search algorithm. In the proposed algorithm temporal redundancy is best utilised by using variable block-size multiresolution motion estimation (Zhang and Zafar, 1992) which is less computationally complex than the constant block-size approach (Uz *et al.* 1991). In the same note, the proposed algorithm avoids bottom-up multiresolution motion estimation approach as it is computationally expensive. The top-down approach is best used as the initial motion estimate is performed on lower resolution images saving computational cost (see Section 5.1.2.4.1).

Finally, the algorithm proposes two error correction methods, intra-coding (block replacement) and inter-coding (motion vector replacement), to achieve video scalability. Drawing a computational analysis against these two approaches is difficult. In both the methods, the initial motion estimation is performed by using top-down multiresolution motion estimation approach, which is an inter-coding technique. The initial motion estimate is then scaled to the appropriate resolution. As in the block replacement approach, intra coding of the macro blocks is done only after the motion vector estimate turns out to be inefficient. Therefore block replacement method uses expensive inter-coding initially and computationally inexpensive macroblock coding. In the motion vector replacement approach, the motion vectors are re-estimated after the initial estimate turns out to be inefficient. The motion vector approach requires repeating the expensive motion estimation process twice. This incurs additional computational cost for the motion vector replacement method. Also, the motion vector replacement method produces poor result in terms of quality compared to the block replacement method as shown in Section 6.5.1. Furthermore, the number of blocks resent can be adjusted based on the available bandwidth. Thus making block replacement method as the suitable solution.

6.5 Performance Evaluation

Video quality can be measured using quantitative measures like *PSNR* or using subjective measure, which uses an expert observation. The performance quality of our work is measured using the quantitative peak signal-to-noise ratio (PSNR) measure. It measures the difference between the original and estimated image. It is given as

$$PSNR = 20\log_{10}\frac{255}{MSE} \qquad (6.6)$$

where,

MSE - the mean square error,

$$MSE = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(f(i,j) - f'(i,j))^2 \qquad (6.7)$$

N, M – represents the size of the image

f(i,j) and f′(i,j) denotes the pixel values at the position (i,j) in the original and the estimated frame respectively

PSNR is measured in decibels (dB). If the PSNR value is higher then the quality of the estimated image is better in comparison with the original image. So in general, higher the PSNR value, higher is the performance of the method involved.

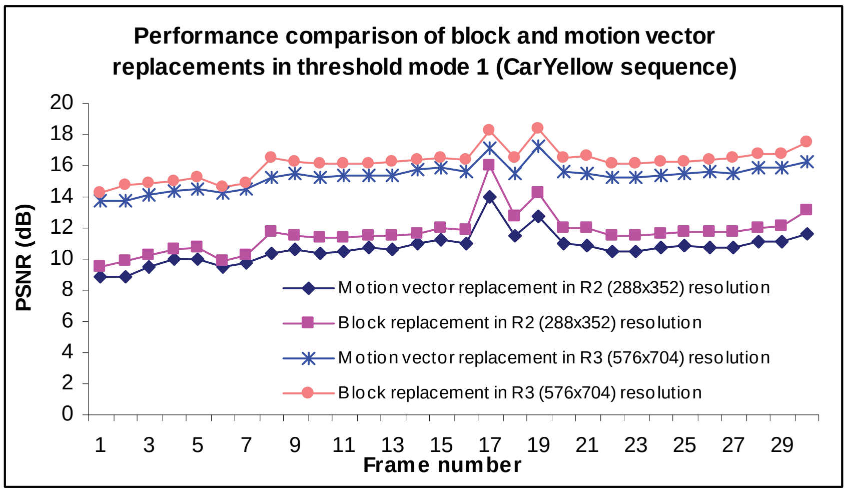
6.5.1 Block Replacement Vs Motion Vector Replacement

In our approach, we have performed two types of error handlers: block replacement and motion vector replacements. Although we have already seen their results in sample outputs presented in Section 6.2, their performance in terms of PSNR would be valuable support.

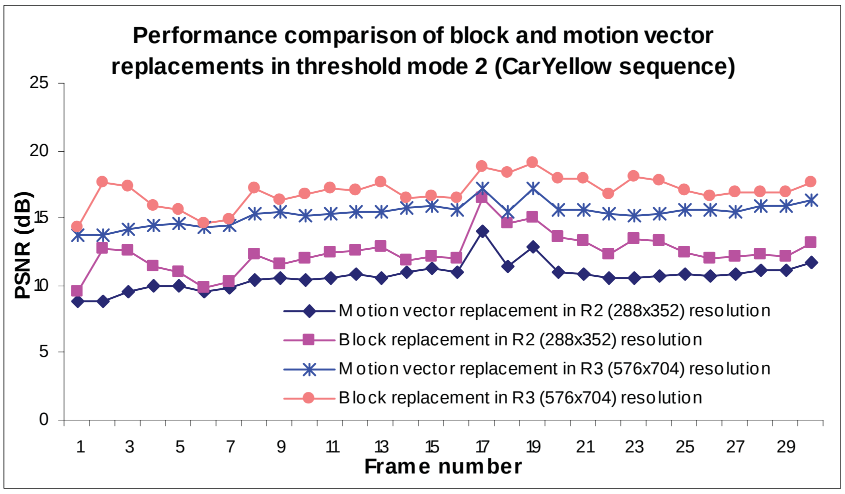
Figure 6.25 and 6.26 presents the performance comparison of block replacement and motion vector replacements in CarYellow and SwanAndBird sequences. The sub-Figures (a) to (c) represent their performance across the three modes of selective data transmission. In each of these plots, the performance of the two error handlers across two resolutions, R2 (288x352) and R3 (576x704) resolutions, are represented.

It can be seen that the block replacement method gives a higher performance in both the video sequence irrespective of the mode of transmission. This is because in block replacement, the error blocks are replaced by their actual blocks, intracoded, which as a result improves the quality of the blocks and also maintains their motion intact. In the other method, the motion vectors of the error blocks are re-estimated in a reduced search area and the resultant new motion vector is used in its place. The usage of another motion vector is not helpful at all times as it only gives a probable estimate of the motion exhibited by the block. Not only does the latter method have high computational load, due to repetition of the motion estimation process for the error blocks, it ends up providing poor error correction. The block replacement method has significantly lower computational load as it only resends the erroneous blocks. The only downside is that it increases the amount of data by intracoding. Even this can be counterbalanced using our selective data transmission model, where the amount of data is controlled according to the available bandwidth. So when the bandwidth limit is low, only a small number of blocks are replaced, as in model, and when there no bandwidth limit large number of blocks is replacement, as in mode 3. Since majority of the blocks are replaced in mode 3, the resultant images have highest quality compared to the ones in mode 1. That is why the block replacement outputs in resolution R2 and R3 have a very high PSNR value in mode 3, Figure 6.25 (c) and 6.26 (c).

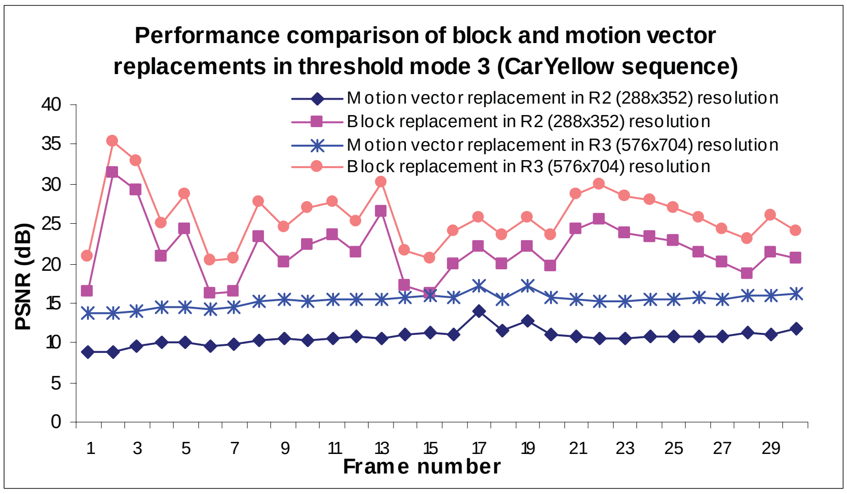
From the presented evaluation and performance comparison, it is apparent that the block replacement method clearly outperforms motion vector replacement through its simplicity and efficiency. Therefore the block replacement method would be our error correction selection for the other following performance evaluation.



(a) Mode 1: Quality Threshold constrained for low bandwidth

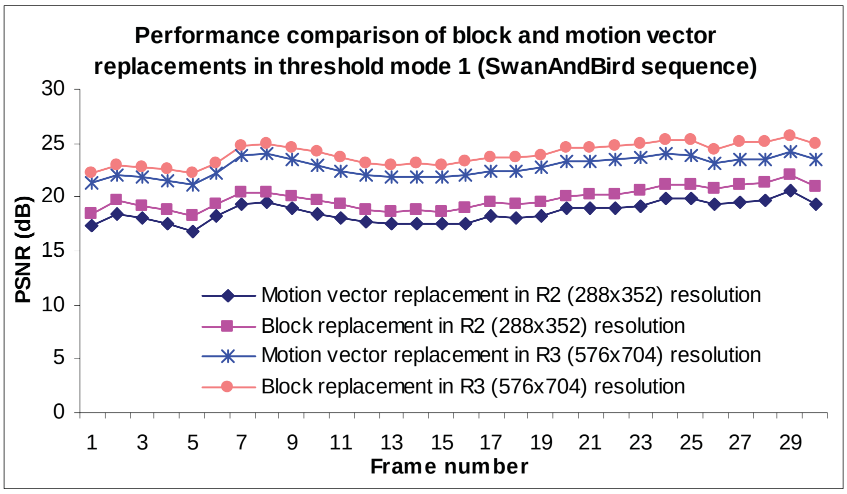


(b) Mode 2: Quality Threshold constrained for high bandwidth

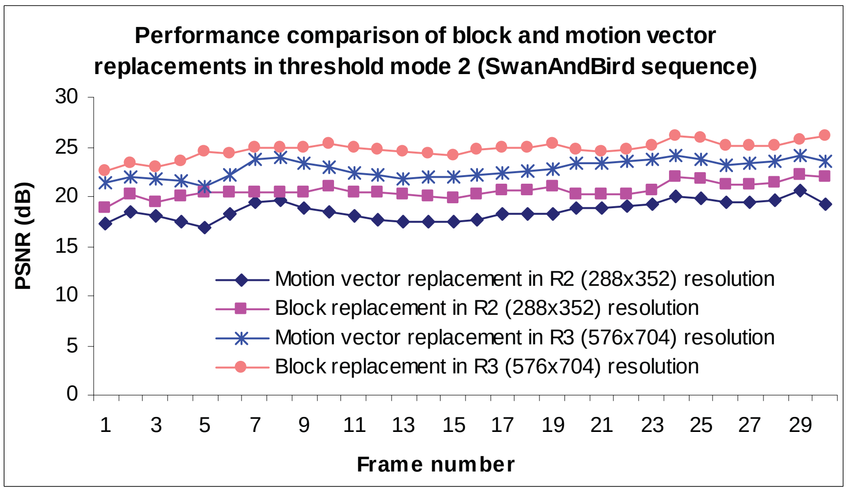


(c) Mode 3: Quality Threshold relaxed for high quality and high bandwidth

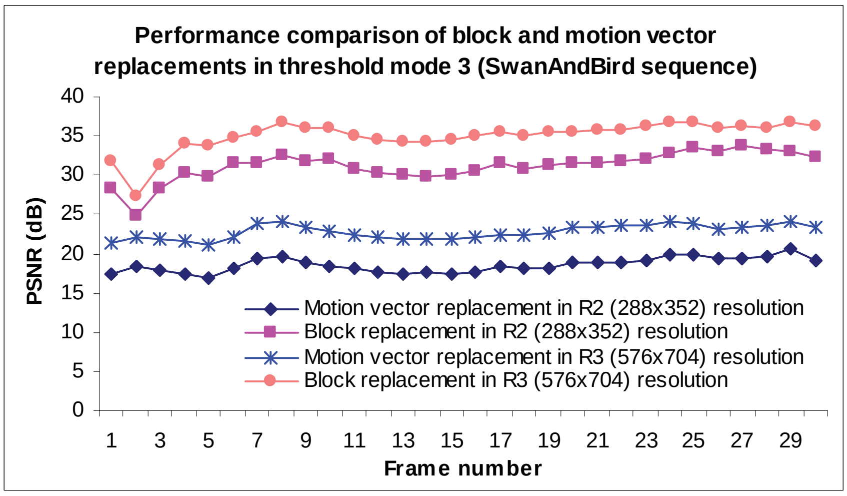
Figure 6.25: Performance comparison of block and motion vector replacement methods across three selective data transmission modes in CarYellow sequence



(a) Mode 1: Quality Threshold constrained for low bandwidth



(b) Mode 2: Quality Threshold constrained for high bandwidth



(c) Mode 3: Quality Threshold relaxed for high quality and high bandwidth

Figure 6.26: Performance comparison of block and motion vector replacement methods across three selective data transmission modes in SwanAndBird sequence

6.5.2 Performance of the Three Different Modes of Selective Data Transmission

In Section 6.2, we have seen the three different modes of selective data transmission. The performances of those three modes are evaluated here. Figure 6.27 and 6.28 present the comparison of the three threshold modes in CarYellow and SwanAndBird sequences respectively. For our evaluation we have used images from R2 (288x352) resolution and block replacement method of error correction.

The threshold mode 3, with low Threshold and high WorstBlock factor, gives the best performance of the three modes in both the video samples. This is due to the fact that in threshold mode 3, the majority of the blocks are subject to replacement. The more blocks are reduced, the greater is the quality of the images and more improved are the motion factor of the block. As it was designed for high quality video images in higher bandwidth situations, it was expected that this mode would perform better than the others, and the results bear this out.

In threshold mode 1, as there is a limitation in the available bandwidth and video quality, the resultant video should have a low performance to adhere to the above limitations. It can be seen from the Figure that threshold mode 1 has the least performance of the three. Even though mode 1 has a lower performance of the three, it gives the best results compared to the other modes under the bandwidth and quality constraints.

The threshold mode 2 has a better performance than the mode 1, because it used in higher bandwidth scenario, but lower than mode 3 as it has restriction on the video quality. Since it uses the appropriate threshold its performance is much closer to mode 1 which also uses the same threshold value.

This evaluation has demonstrated that our selective data transmission method works appropriately according to its chosen modes.

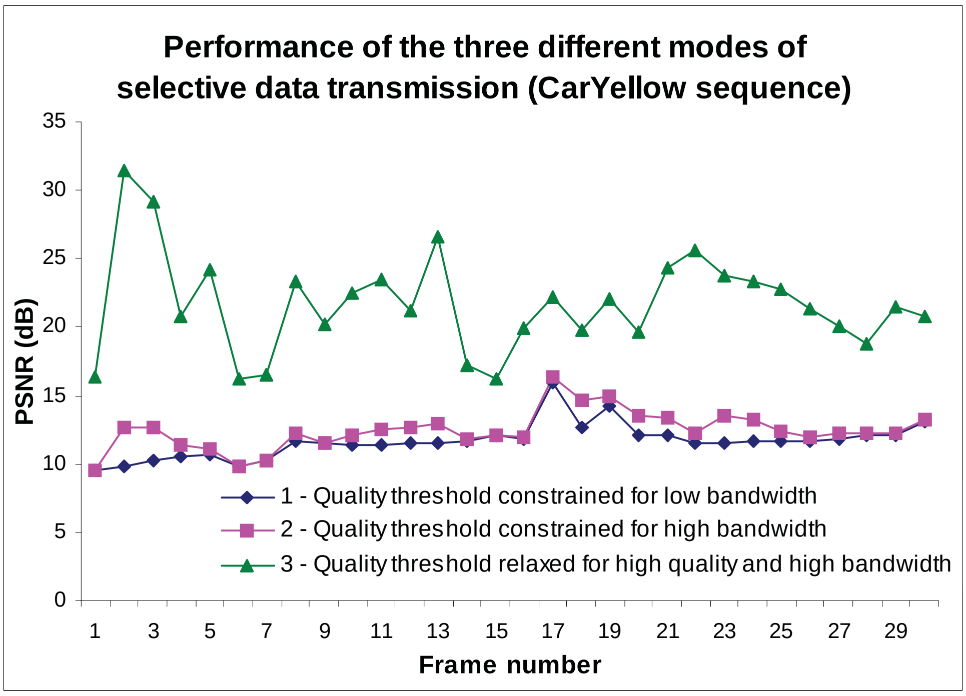


Figure 6.27: Performance of the three modes of selective data transmission in CarYellow sequence.

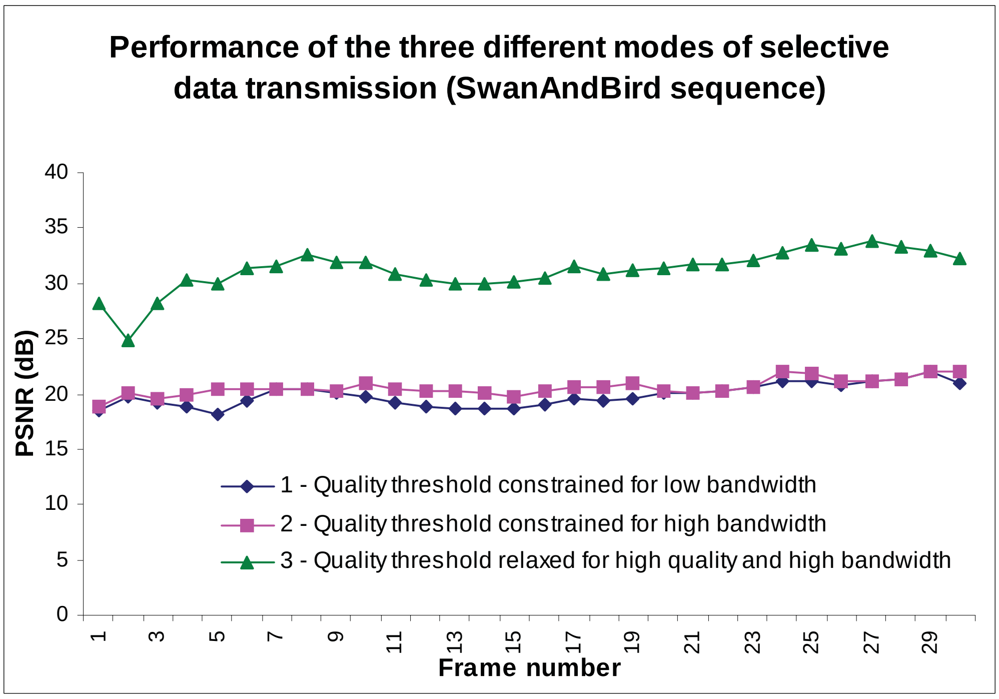


Figure 6.28: Performance of the three modes of selective data transmission in SwanAndBird sequence.

6.5.3 Performance in Different Motion Sequences

The performance of our approach in different type of motion sequence is considered here. For our evaluation we have taken two different motion sequences: CarYellow and SwanAndBird sequences. The CarYellow sequence with a slow moving background and fast moving foreground object captured using a panning movement has a fast and complex motion, while the latter with a slow moving foreground and a fast moving background object has an overall slow and simple motion.

We have tested our sample sequences against two of our selective transmission modes representing two extremes: mode 1 with nominal values and mode 3 with higher values, Figure 6.29 and 6.30 respectively. In both the modes our approach gives a higher performance in SwanAndBird compared to CarYellow sequence. This performance is consistent except at the second and third frame in Figure 6.30 (mode 3), where drastic change is introduced in the first sequences and also because of difference in the number of block replacements. That is, in SwanAndBird sequence the number of blocks replaced is lower in frame 2 and 3 compared to other frames of that sequence, and is almost equal to the block count of CarYellow sequence, refer Figure 6.17.

The overall better performance in SwanAndBird sequence is mainly due to its simple motion structure. Therefore, it can be concluded that our approach gives very high performance in sequences with simple motion and gives relatively good performance in complex motion sequence.

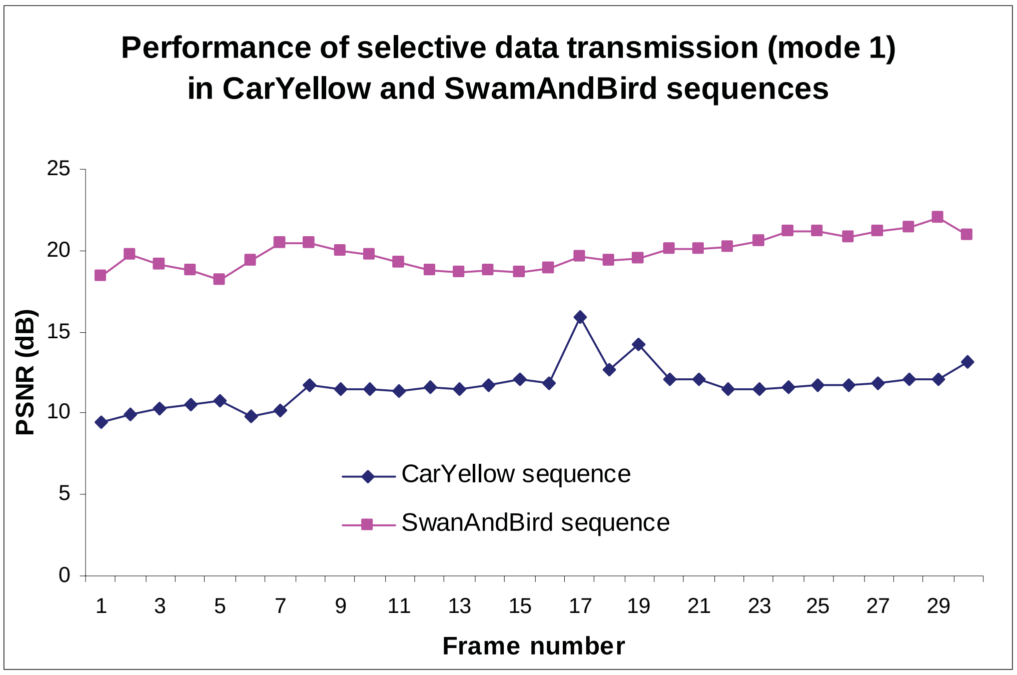


Figure 6.29: Performance of selective data transmission of low quality video during low bandwidth (mode 1) in CarYellow and SwanAndBird sequences.

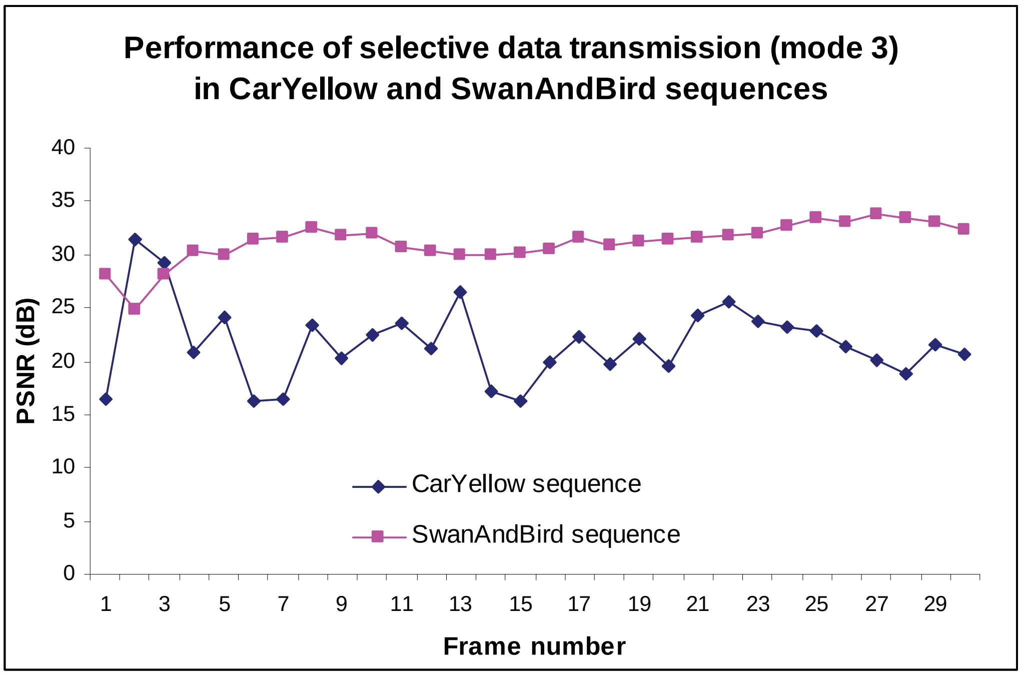


Figure 6.30: Performance of selective data transmission of high quality video during high bandwidth (mode 3) in CarYellow and SwanAndBird sequences.

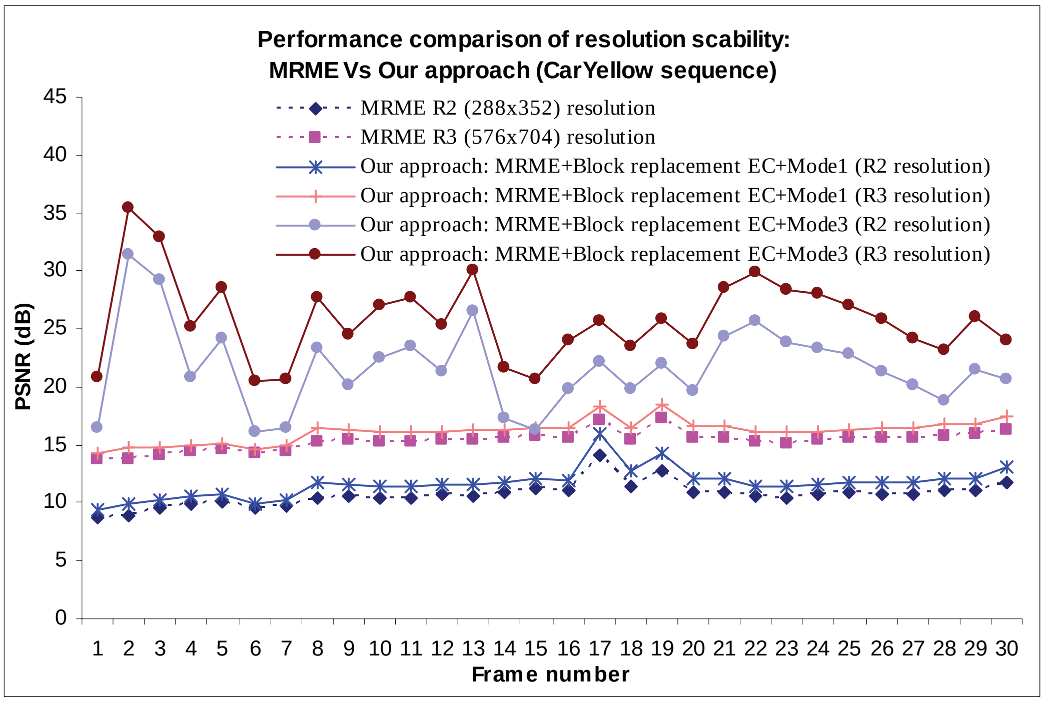
6.5.4 Resolution Scalability

In our approach, we have extracted three different resolutions, R1 (144x176), R2 (288x352) and R3 (576x704), by using wavelet decomposition and multiresolution motion estimation (MRME). The images were then subjected to error correction of block replacement. The resolution scaled images were then used for selective data transmission based on the network and user requirements.

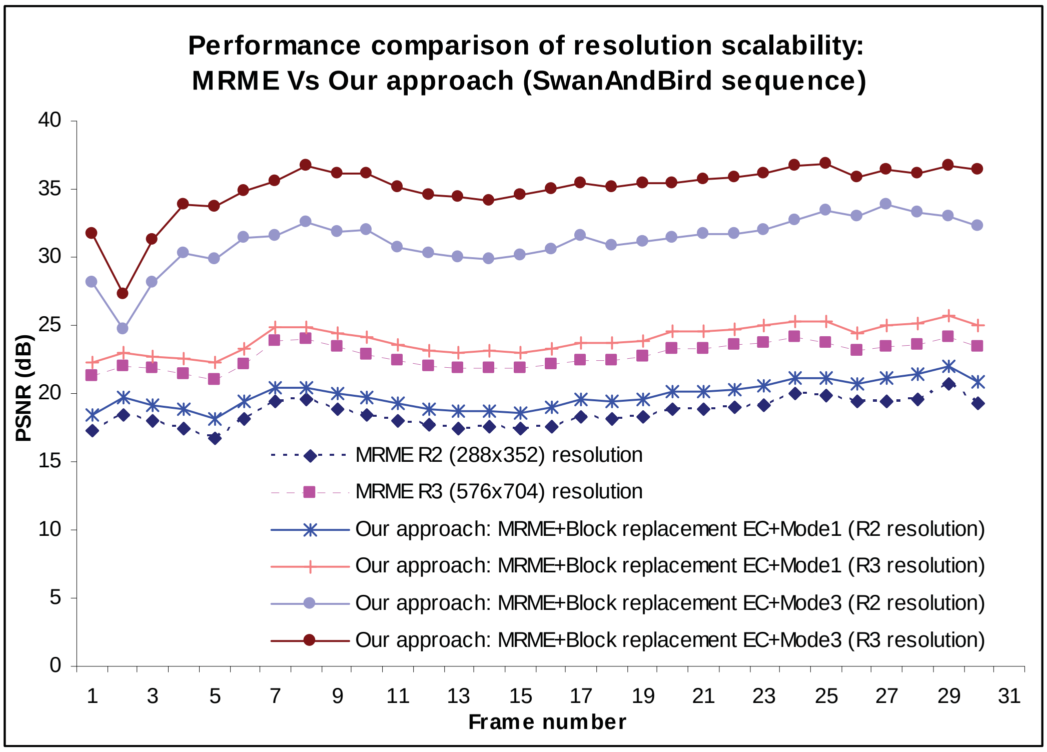
Here, we present the comparison of our approach against the resolution scaled images formed using multiresolution motion estimation without any error correction. We have used R2 and R3 images from CarYellow and SwanAndBird sequence for our evaluation, Figure 6.31 For illustration of selective data transmission we have represented our approach in two extreme test cases: mode 1 and mode 3. Mode 1 is for transmission of low quality video in low bandwidth scenario whereas mode 3 is for high quality video in high bandwidth. Mode 3 has been demonstrated to the have better performance than Mode 1 (see Section 6.5.2).

Even in mode 1 our approach performs much better than the conventional MRME approach in both the resolution levels. In Mode 3 our approach produces an excellent performance with very PSNR, thereby outperforming other approaches. This performance is consistent in both the video sequences.

Therefore our approach of resolution scalability combined with block replacement error correction and selective data transmission clearly outperforms the MRME approach in any resolution mode irrespective of the type of motion sequence.



(a) CarYellow sequence



(b) SwanAndBird sequence

Figure 6.31: Performance comparison of resolution scaled images using multiresolution motion estimation (MRME without error correction) Vs our approach (MRME + Block replacement Error Correction (EC)) in selective transmission mode 1 (low video quality and low bandwidth) and mode 3 (high video quality and higher bandwidth)

7. Conclusion

In our work we have presented a spatial scalable video coding based on wavelet transform and multiresolution motion compensation that is capable of selectively transmitting bit streams based on the user and network constraints. We have used a simple and effective approach to perform the scalable functions. The wavelet transform has been employed in this work in order to take full advantage of its flexible scalable and multiresolution representation. The use of 2D transform instead of 3D has been highly beneficial in providing less constraint on memory usage and simplified coding. The combination of multiresolution representation and multiresolution motion estimation provides easy motion scalability and high coding efficiency. The variable block size MRME approach helps to provide easy and accurate motion translations across all resolutions. Furthermore the computational cost is greatly reduced by limiting the prediction process to a smaller image subset, using a top-down approach. As the motion compensation process is carried out in the wavelet domain rather than the spatial domain, the motion features can easily be added to the different versions of the video with great flexibility and with all the advantage of the in-band prediction approach. In addition, our approach provides error resilience for the error generated during the motion translation or any drift error. Instead of providing error correction for all the blocks, as done in all the conventional methods, our framework is designed to provide error correction for only the affected blocks. From our simulation result we imply that it is best to intracode the error blocks rather than correcting the motion prediction, which requires repeating the motion estimation process and also leads to quality degradation. By using the suggested block replacement approach, we can have many advantages: the quality of the image can be improved in addition to providing correct prediction, and high computational load can be avoided. Moreover this replacement can be adjusted based on our requirements.

The most important feature of our work is the selective data transmission function. Using this function, the data required for transmission can be controlled, be it for error resilience or for bit rate control. The two control parameter used for this purpose, helps to limit the total required data to a minimum, so that only the necessary information is sent to meet a highly-constraint network. The control parameters are flexible and can be varied depending on quality and bandwidth requirements. So, when the user requirement increases, the control parameter can be set to maximum limit to allow for more data transmission to obtain a higher version of the video, in any levels of scalability, quality or resolution.

As we have used a wavelet representation, the motion compensated image can be encoded in an embedded form. The selective data function can also be included in the embedded encoding, which gives added advantage.

Thus our spatial scalable framework provides great scalable features and good error resilience. The simulation results also confirm the same. The framework outperforms the traditional multiresolution motion estimation approach. The selective data transmission function also provides excellent adaptation based on the network and user needs.

Future Work

Our present video coding framework only provides spatial scalability, so in our future extension we would like to develop a fully scalable video codec which provides all three scalable options: spatial, temporal and quality. In addition, we would like to test the embedded bit stream representation using the EBCOT coder. Moreover, we would like to include our selective data transmission function to the embedded structure to achieve maximum adaptability and flexibility. Even though the proposed selective transmission approach provides greater performance, we would like to further refine it and improve its performance across all types of motion sequence. Finally we would like to perform a comparison of our fully scalable video codec against the existing video standards and the JPEG2000 image standard.