# 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

# Reference

Ahmad, I., Wei, X., Sun, Y., and Zhang, Y.-Q., (2005), 'Video transcoding: An Overview of Various Techniques and Research Issues', *IEEE Transactions on Multimedia*, vol. 7, no. 5, October, pp. 793-804.

Ahmed, N., Natarajan, T., and Rao, K.R., (1974), 'Discrete Cosine Transform', *IEEE Transactions on Computers*, vol. C-23, January, pp. 90-93.

Al-Mualla, M., Canagarajah, C.N., and Bull, D.R., (2002), '*Video Coding for Mobile Communications: Efficiency, Complexity, and Resilience*', Academic Press, California, USA.

Andreopoulos, Y., van der Schaar, M., Munteanu, A., Barbarien, J., Schelkens, S., and Cornelis, J., (2002), 'Open-loop, in-band motion-compensated temporal filtrating for objective for objective full-scalability in wavelet video coding', *ISO/IEC JTC1/SC29/WG11, m9026, MPEG 62nd meeting*, Shanghai, China, October.

Andreopoulos, Y., van der Schaar, M., Munteanu, A., Barbarien, J., Schelkens, S., and Cornelis, J., (2003), 'Fully-scalable wavelet video coding using in-band motion compensated temporal filtering', in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2003*, Hong Kong, vol. 3, March, pp. 417-420.

André, T., Cagnazzo, M., Antonini, M., and Barlaud, M., (2007), 'JPEG2000-compatible scalable scheme for wavelet-based video coding', *EURASIP Journal on Image and Video Processing*, vol. 2007, no. 30852, January.

Antonini, M., Barland, M., Mathieu, P., and Daubechies, I., (1992), 'Image coding using wavelet transform', *IEEE Transaction on Image Processing*, vol. 1, no. 2, April, pp. 205-220.

Arnold, J., Frater, M., and Pickering, M., (2007), *'Digital television: technology and standards'*, John Wiley & Sons, Hoboken, New Jersey.

Bjork, N., and Christopoulos, C., (1998), 'Transcoder architecture for video coding', *IEEE Transactions on Consumer Electronics*, vol. 44, no. 1, February, pp. 88-98.

Bosveld, F., Lagendijk, R., and Biemond, J., (1992), 'Hierarchical video coding using a spatio-temporal subband decomposition', in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, San Francisco, CA, USA, vol. 3, March, pp. 221-224.

Bottreau, V., Benetiere, M., Felts, B., and Pesquet-Popescu, B., (2001), 'A Fully Scalable 3D subband video Codec', in *Proceedings of the IEEE Conference on Image Processing (ICIP) 2001*, Greece, vol. 2, October, pp. 1017-1020.

CCIR-601 (1990), 'International Consultative Committee for Radio (CCIR) Recommendation 601-2: Encoding parameters of digital television for studios'. In *Digital Methods of Transmitting Television Information*, Section 11 F, pp. 95-104.

Chang, S.-F, and Vetro, A., (2005), 'Video Adaptation: Concepts, Technologies and Open Issues', in *Proceeding of the IEEE*, vol. 93, no. 1, January, pp. 148-158.

Chen, P., and Woods, J.W., (2002), 'Video coding for digital cinema', in *Proceeding of IEEE International Conference on Image Processing*, Rochester, NY, September, pp. 744-752.

Chen, P., and Woods, J.W., (2004), 'Bidirectional MC-EZBC with lifting implementation', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 10, October, pp. 1183-1194.

Clerckx, T., Verdicchio, F., Munteanu, A., Andreopoulos, Y., Devos, H, Eeckhaut, H, Christiaens, M, Stroobandt, D, Verkest, D, and Schelkens, P, (2004), 'Complexity scalable motion compensated temporal filtering', in *Proceeding of the SPIE*, vol. 5558, August, pp. 116-127.

Conklin, G., and Hemami, S., (1997), 'Multi-Resolution Motion Estimation', *Proceeding of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97)*, vol. 4, April 22-24, pp. 2873.

Dogan, S., and Sadka, A.H., (2002), 'Video transcoding for inter-network communications', in *Compressed Video Communications* by Sadka, A.H., John Wiley and Sons Ltd., Chichester, UK, Chapter 6, pp. 215-256, 2002.

Dufaux, F., and Moscheni, F., (1995), 'Motion estimation techniques for digital TV: A review and a new contribution', *Proceedings of the IEEE*, vol. 83, no.6, June, pp. 858-875.

Ebrahimi, T., van der Schaar, M., and Ohm, J.-R., (2002), 'The status of interframe wavelet coding exploration in MPEG', *International Organisation for Standardisation, ISO/IEC JTC1/SC29/WG11, n4928, MPEG 61st Meeting*, Klagenfurt, Austria, July.

Eisert, P. and Girod, B., (1997), 'Model-based 3D motion estimation with illumination compensation', in *Proceedings of International Conference on Image Processing and its Applications*, Dublin, Ireland, July, pp. 194-198.

Ghanbari, M., (1999), 'Video Coding: An Introduction to Standard Codec', *Institution of Electrical Engineers*, IEE Telecommunications series; 42, London.

Girod, B., and Horn, U., (1997), 'A scalable codec for Internet video streaming', in *Proceedings of the International Conference on Digital Signal Processing*, Santorini, Greece, vol. 1, pp. 221-224.

GSM Arena (2000), GSM handset specification (online), http://www.gsmarena.com/, Accessed: September 2005.

Haskell, B.G., Puri, A., and Netravali, A.N., (1997), '*Digital Video: An Introduction to MPEG-2*', Kulwer Academic Publishers, Boston.

Holschneider, M., (1995), '*Wavelets, an analysis tool'*, Clarendon Press Oxford.

Hu, Z., van der Schaar, M., and Pesquet-Popescu, B., (2004), 'Scalable Motion Vector Coding for MC-EZBC', in *Proceeding of EUSIPCO*, Vienna, Austria, September, pp.657-660.

ISO/IEC 11172, Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1.5 M bit/s, *Motion Pictures Expert Group (MPEG), International Organisation for Standardisation ISO/IEC JTC1/SC29/WG11*. http://www.chiariglione.org/mpeg/standards/mpeg-1/mpeg-1.htm

ISO/IEC 13818, Generic coding of moving pictures and associated audio information, *Motion Pictures Expert Group (MPEG), International Organisation for Standardisation ISO/IEC JTC1/SC29/WG11*. http://www.chiariglione.org/mpeg/standards/mpeg-2/mpeg-2.htm

ISO/IEC 14496, Coding of audio-visual objects, *Motion Pictures Expert Group (MPEG), International Organisation for Standardisation ISO/IEC JTC1/SC29/WG11, N4668.* http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm

ISO/IEC 14496-2:2001/Amd 2:2002, Information technology - Coding of audio-visual objects - Part 2: Visual - Amendment 2: Streaming video profile, *Motion Pictures Expert Group (MPEG), International Organisation for Standardisation ISO/IEC JTC1/SC29/WG11*.

ISO/IEC 15444-1:2000, Information technology – JPEG 2000 image coding system: Part 1: Core coding system (ITU-T Rec. T.800), *International Standard*. http://www.jpeg.org/public/fcd15444-1.pdf

ITU-T (1990), Video codec for audiovisual services at px64 kbps, *International Telecommunication Union – Telecommunication Standardisation Sector (ITU-T) Recommendation H.261*

ITU-T (1996), Video coding for low bit-rate communication at rates less than 64 kbps, *International Telecommunication Union – Telecommunication Standardisation Sector (ITU-T) Recommendation H.263*, version 1, 1996; version 2, 1997

Jain, A.K., (1989), ‘*Fundamentals of Image Processing*’, Prentice-Hall, Englewood Cliffs, NJ.

Johnson, A.W., Sikora, T., Tan, T.K., and Nagan, K.N., (1994), ‘Filters for drift reduction in frequency scalable video coding schemes’, *Electronics Letters’*, vol. 30, no. 6, March, pp. 471-472.

Karlsson, G., and Vetterli, M., (1987), ‘Subband coding of video signals for packet switched networks’, in *Proceeding of SPIE Visual Communications and Image Processing II*, Cambridge, MA, USA, vol. 845, October, pp. 446-456.

Kim, B.-J., and Pearlman, W., (1997), ‘An embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees (SPIHT)’, in *Proceedings of the Data Compression Conference,* Snowbird, UT, USA, March, pp. 251-260.

Kim, B.-J., Xiong, Z., and Pearlman, W., (2000), ‘Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)’, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 8, December, pp. 1374-1387.

Kim, S, Rhee, S., Jeon, J.G., and Park, K.T., (1998), ‘Interframe coding using two-stage variable block-size multiresolution motion estimation and wavelet decomposition’, *IEEE Transactions on Circuit and Systems for Video Technology*, vol. 8, no. 4, August, pp. 399-410.

Kronander, T., (1989), ‘Motion compensated 3-dimensional wave-form image coding’, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Glasgow, Scotland, UK, vol. 3, pp. 1921-1924.

Kronander, T., (1990), ‘New results on 3-dimensional motion compensated subband coding’, in *Proceedings of the Picture Coding Symposium (PCS),* Cambridge, MA, USA, pp. 8-10.

Li, W., (2001), ‘Overview of fine granularity scalability in MPEG-4 video standard’, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, March, pp. 301-317.

Lippman, A., and Butera, W., (1989), ‘Coding image sequences for interactive retrieval’, *Communications of the ACM*, vol. 32, no. 7, July, pp. 852-860.

López, M.F., Rodríguez, S.G., Ortiz, J.P., Dana, J.M., Ruiz, V.G., and García, I., (2005), ‘Fully Scalable Video Coding with Packed Stream’, in *Proceedings of the SPIE Image and Video Communications and Processing*, eds., Said, A., Apostolopoulos, J.G., San Jose, CA, USA, vol. 5685, January, pp. 378-389.

Mallat, S., (1989a) ‘A theory of multiresolution signal decomposition: the wavelet representation’, *IEEE Transactions on Pattern and Machine Intelligence*, vol. 11, no. 7, July, pp. 674-693.

Mallat S., (1989b) ‘Multifrequency channel decompositions of images and wavelet models’, *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. 37, no.12, December, pp. 2091-2110.

Mandal, M.K., Chan, E., Wong, X., and Panchanathalu, S., (1996), ‘Multiresolution motion estimation techniques for video compression’. *Optical Engineering*, vol. 35, no.1, January, pp. 128-136.

Marpe, D., George, V., Cycon, H.L., anf Barthel, K.U., (2004), ‘Performance evaluation of Motion-JPEG2000 in comparison with H.264/AVC operated in pure intra coding mode’, in *Wavelet Applications in Industrial Processing*, *Proceedings of SPIE*, vol. 5266, Providence, RI, USA, October, pp.129-137.

Mohan, R., Smith, J.R., and Li, C., (1999) 'Adapting multimedia internet content for universal access', *IEEE Transactions on Multimedia*, vol. 1, no.1, March, pp. 104-114.

Naveen, T., Bosveld, F., Woods, J., and Lagendijk, R., (1995), 'Rate constrained multiresolution transmission of video', *IEEE Transaction on Circuits and Systems for Video Technology*, vol. 5, no. 3, June, pp. 193-206.

Netravali, A.N., and Haskell, B.G., (1995), '*Digital Pictures: Representation, Compression and Standards'*, Second edition, Plenum Press, New York.

Nosratinia, A., and Orchard, M.T., (1995), 'A multiresolution framework for backward motion compensation', In *SPIE Proceedings on Electronic Imaging*, San Jose, CA, February.

Nosratinia, A., Davis, G., Xiong, Z., and Rajagopalan, R., (1999), 'Subband Image Compression', in *Wavelet, Subband and Block Transforms in Communications and Multimedia*, Akansu A.N., and Medley M.J., eds., Kluwer Academic Publishers, Boston, MA.

Ohm, J.-R., (1994a), 'Three-dimensional subband coding with motion compensation', IEEE Transactions on Image Processing, vol. 3, no. 5, pp. 559-571, September 1994.

Ohm, J.-R, (1994b), 'Motion-compensated 3-D subband coding with multiresolution representation of motion parameters', in *Proceedings of the IEEE International Conference on Image Processing*, Austin, TX, USA, vol. 3, November, pp. 250-254.

Ohm, J.-R., van der Schaar, M., and Woods, J.W., (2004), 'Interframe wavelet coding – Motion picture representation for universal scalability', *EURASIP Signal Processing: Image Communication*, vol. 19, no. 9, October, pp. 877-908.

Pearson, D.E., (1995), 'Developments in model-based video coding', *Proceedings of the IEEE*', vol. 83, no. 6, June, pp. 892-906.

Rao, K.R., Bojkovic, Z.S., and Milovanovic, D., (2002), *Multimedia Communication Systems: Techniques, Standards, and Networks'*, Prentice Hall, New Jersey.

Rhee, I., Martin, G.R., and Packwood, R.A., (1995), 'A multiresolution block matching technique for image motion estimation', in *IEE Colloquium on Multiresolution Modeling and Analysis in Image Processing and Computer Vision*, London, UK, April, pp.5/1-5/6.

Said, A., and Pearlman W., (1996), 'A new, fast, and efficient image codec based on Set Partitioning in Hierarchical Trees (SPIHT)', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, June, pp. 243-250.

Secker, A., and Taubman, D., (2001), 'Motion-compensated highly scalable video compression using an adaptive 3D wavelet transform based on lifting', in *Proceedings of the International Conference on Image Processing (ICIP) 2001*, Thessaloniki, Germany, vol. 2, October pp. 1021-1032.

Schwarz, H., Marpe, D., and Wiegand, T., (2007), 'Overview of the scalable video coding extension of the H.26/AVC standard', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no.9, September, pp. 1103-1120.

Shanableh, S., and Ghanbari, G., (2000), 'Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats', *IEEE Transactions on Multimedia*, vol. 2, no.2, June, pp. 101-110.

Shapiro, J., (1993), 'Embedded image coding using zerotrees of wavelet coefficients', *IEEE Transactions on Signal Processing*, vol. 41, no. 12, December pp. 3445-3462.

Shen, B., Sethi, I.K., and Vasudev, B., (1999), 'Adaptive motion-vector resampling for compressed video downsampling', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no.6, September, pp. 929-936.

Skodras, A., Christopoulos, C., and Ebrahimi, T., (2001), 'The JPEG 2000 still image compression standard', *IEEE Signal Processing Magazine*, vol. 18, September, pp. 36-58.

Sorell, M., (2005), (Video Images) Lecture notes - Multimedia Communications, *School of Electrical and Electronic Engineering, University of Adelaide*. Accessed: October 2005.

Sun, H., Kwok, W., and Zdepski, J.W., (1996), 'Architectures for MPEG compressed bitstream scaling', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no.2, April, pp. 191-199.

Tan, K.-H., and Ghanbari, M., (1995), 'Layered image coding using the DCT pyramid', *IEEE Transactions on Image Processing*, vol. 4, no. 4, April, pp-512-516.

Taubman, D., (2000), 'High performance scalable image compression with EBCOT', *IEEE Transactions on Image Processing*, vol. 9, no.7, July, pp. 1158-1170.

Taubman, D., (2001), Kakadu software: a comprehensive framework for JPEG2000 developers (JPEG2000 coder), http://www.kakadusoftware.com/, Accessed: October 2005.

Taubman, D., and Marcellin, M., (2002), '*JPEG2000: Image Compression Fundamentals, Standards and Practice*', 2nd ed., Kluwer Academic Publishers, Norwell, USA.

Taubman, D., and Zakhor, A., (1994), 'Multirate 3-D subband coding of video', *IEEE Transaction on Image Processing*, vol. 3, no. 5, September, pp. 572-588.

Tham, J., Ranganath, S., Kassim, A., (1998), 'Highly scalable wavelet-based video codec for very low bit-rate environment', *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 1, January, pp. 21-27.

Topiwala P., (1998), *Wavelet image and video compression*, Kluwer Academic Publishers, Boston, MA.

Turaga, D., and van der Schaar, M., (2002) ‘Wavelet coding for video streaming using new unconstrained motion compensated temporal filtering’, in *Proceedings of the International Workshop on Digital Communications: Advanced Methods for Multimedia Signal Processing (IWDC) 2002*, Capri, Italy, September, pp. 41-48.

Tzovaras, D., Strintzis, M.G., and Sahinoglou, H., (1994), ‘Evaluation of multiresolution block matching techniques for motion and disparity estimation’, *Signal Processing: Image Communication*, vol. 6, no.1, pp. 59-67.

Uz, K., Vetterli, M., and LeGall, D., (1990), ‘A multiresolution approach to motion estimation and interpolation with application to coding of digital HDTV’, in *Proceedings of the IEEE International Symposium on Circuits and Systems*, New Orleans, LA, USA, vol. 2, May, pp. 1298-1301.

Uz, K., Vetterli, M. and LeGall, D., (1991), ‘Interpolative multiresolution coding of advanced television with compatible subchannels’, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, no. 1, March, pp. 86-99.

van der Schaar, M., and Radha, H., (2001), ‘A hybrid temporal-SNR fine-granular scalability for internet video’, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, March, pp. 318-331.

Verdicchio, F., Andreopoulos, Y., Munteanu, A., Cornelis, J., and Schelkens, P., (2003), ‘Architectural Aspects if Scalable Wavelet Video Coding’, in *Proceeding of the Third PA3CT Symposium*, September, pp. 57-59.

Vetro, A., Christopoulos, C., and Sun, H., (2003), ‘Video Transcoding Architectures and Techniques: An Overview’, *IEEE Signal Processing Magazine*, vol. 20, no. 2, March, pp- 18-29.

Vetterli M., (1984), ‘Multidiemsional subband coding: Some theory and algorithms’, *Signal Processing*, vol. 6, April, pp. 97-112.

Wang, A., Xiong, Z., Chou, P., and Mehrotra, S., (1999), ‘Three-dimensional wavelet coding of video with global motion compensation’, in *Proceedings of the Data Compression Conference (DCC)*, Snowbird, UT, USA, March, pp. 404-413.

Wang, Q., Wu, F., Li, S., Xiong, Z., Zhang, Y.-Q., and Zhong, Y., (2001), ‘A new rate allocation scheme for progressive fine granular scalable coding’, in *Proceeding of the IEEE International Symposium of Circuits and Systems*, Sydney, Australia, vol. 2, May, pp. 397-400.

Wang, Y., Ostermann, J., and Zhang, Y.Q., (2002), ‘*Video Processing and Communications*’, Prentice Hall Englewood Cliffs, NJ.

Woods, J.W., ed., (1991), *‘Subband Image Coding’*, Kluwer Academic Publishers, Norwell, USA.

Woods, J.W., and Chen, P., (2002), ‘*Improved MC-EZBC with quarter-pixel motion vectors*’, MPEG document, ISO/IEC JTC1/SC29/WG11 MPEG2002/M8366, Fairfax, May.

Woods, J.W., and Lilienfield, G., (2001), ‘A resolution and frame-rate scalable subband/wavelet video coder’, *IEEE Transaction on Circuits and Systems for Video Technology,* vol. 11, no. 9, September, pp. 1035-1044.

Woods, J.W, and O’Neil, S., (1986), ‘Subband coding of images’, *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 34, October, pp. 1278-1288.

Wu, Y., and Woods, J., (2005), ‘Scalable motion vector coding for MC-EZBC’, in *Proceedings of SPIE Electronic Imaging*, San Jose, CA, January, pp. 335-346.

Xin, J., Lin, C.-H., and Sun, M.-T., (2005), 'Digital Video Transcoding', in *Proceedings of the IEEE (Special Issue on Advances in Video Coding and Delivery)*, vol. 93, no.1, January, pp.84-95.

Xin, J., Sun, M.-T., Chun, K., and Choi, B.S., (2002), 'Motion re-estimation for HDTV to SDTV transcoding', in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS) 2002*, Geneva, Switzerland, vol. 4, May, pp. 715-718.

Xiong Z, Ramchandran K, Orchard M.T., and Zhang Y.-Q., '*A Comparative Study of DCT- and Wavelet-Based Image Coding'*, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 9 No. 5, pp.692-695, August 1999.

Xu, J., Xiong, Z., Li, S., and Zhang, Y.-Q., (2001), 'Three-dimensional embedded subband coding with optimal truncation (3D ESCOT)', *Applied and Computational Harmonic Analysis*, vol. 10, pp.290-315.

Xu, J., Xiong, Z., Li, S., and Zhang, Y.-Q., (2002), 'Memory-constraint 3D wavelet transform for video coding without boundary effects', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 9, September, pp. 812-818.

Ye, J., and van der Schaar, M., (2003), 'Fully scalable 3-D overcomplete wavelet video coding using adaptive motion compensated temporal filtering', *Proceedings of SPIE Video Communication and Image Processing*, June, pp. 1169-1180.

Yin, P., and Wu, M., (2000), 'Video transcoding by reducing spatial reduction', in *Proceedings of IEEE International Conference on Image Processing 2000*, Vancouver, Canada, vol. 1, September, pp. 972-975.

Yin, P., Vetro, A., Liu, B., and Sun, H., (2002), 'Drift compensation for reduced spatial resolution transcoding', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 11, November, pp. 1009-1020.

Youn, J., and Sun, M.-T., (2000), ‘Video transcoding with H.263 bit streams’, *Journal of Visual Communication and Visual Image Representation*, vol. 11, no. 4, December, pp. 385-404.

Youn, J., Sun, M.-T., and Xin, J., (1999), ‘Video Transcoder Architectures for Bit Rate Scaling of H.263 Bit Streams’, in *Proceeding of the seventh ACM international conference on Multimedia (Part 1) 1999*, Orlando, Florida, USA, October, pp. 243-250.

Zafar S., Zhang Y.-Q., and Jabbari B., (1993), ‘Multiscale Video Representation Using Multiresolution Motion Compensation and Wavelet Decomposition’, *IEEE Journal on Selected Areas in Communication*, vol. 11, no. 1, January, pp. 24-35.

Zan J. Ahmad M.O., and Swamy M.N.S., (2006), ‘A Multiresolution Motion Estimation Technique with Indexing’, *IEEE Transactions on Circuits and Systems on Video Technology*, vol. 16, no. 2, February, pp. 157-165.

Zhang, X.M., Vetro, A., Shi, Y.Q., and Sun, H., (2002), ‘Constant quality constrained rate allocation for FGS coded video’, in *Proceedings of SPIE Conference on Visual Communications and Image Processing*, San Jose, CA, USA, January, pp. 817-827.

Zhang Y.-Q., and Zafar S., (1992), ‘Motion-Compensated Wavelet Transform Coding for Color Video Compression’, *IEEE Transactions on Circuits and Systems For Video Technology*, vol. 2, no. 3, September, pp. 285-296.

Zhao, L., Kim, J.W., Bao, Y., Kuo and Kuo, C.-C.J., (2000), ‘Highly scalable differential JPEG-2000 wavelet video codec for Internet video streaming’, in *Proceedings of the SPIE: Applications of Digital Image Processing XXIII*, vol. 4115, December, pp. 130-140.

Zhu, W., Yang, K.H., and Beacken, M.J., (1998), ‘CIF-to-QCIF video bitstream down-conversion in the DCT domain’, *Bells Labs Technical Journal*, vol. 3, no. 3, August, pp.21-29.

# Appendix

## A. Haar wavelet

The haar function (Holschneider 1995) is expressed as

$$g(t) \leftarrow \begin{cases} -1 & for 0 \leq t < 1/2 \\ +1 & for 1/2 \leq t < 1 \\ 0 & otherwise \end{cases}$$

This function was the first wavelet to be used in mathematics and was introduced by Alfred Haar in 1906.

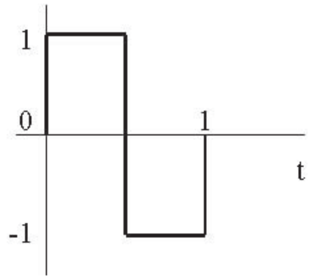


Figure A.1. Haar wavelet in time representation

## B. Code

**1. SpatialScalableVC.m**

```
%Spatial Scalable video coding with selective data transmission
%BlockDiagram
%------------------
%WT->Spatial Scalability->MRME->EC(with Threshold + Selective Data Transmission)
%->1.Block Replacement 2. Motion Vector Replacement
%Simulation Configuration
%-------------------------------
%1. Wavelet Transform
%       - haar wavelet
%       - Two decomposition levels
%2. Motion Estimation
%       - Block estimation – Full search – SAD search criteria
%3. Multiresolution motion estimation
%       - Top-down approach
%       - Variable block-size and search area

clear all
close all
format long g %To make the answer of the 'percentage' store without exponential
iptsetpref('ImshowAxesVisible', 'on'); %to make the axes values visible in the image
                                                    %outputs

% 1. Automated image(s) extraction
% ----------------------------------------
%(with gray and double conversion)
max_frames =31;
[FrameSet] = FrameExtractionFull_carpan(max_frames); %Carpann sample set
% [FrameSet] = FrameExtractionFull_Bird(max_frames); %Bird sample set
% [FrameSet] = FrameExtractionFull_Swan(max_frames); %Swan sample set
 %[FrameSet] = FrameExtractionFull_Wcar(max_frames); %WhiteCar sample set

%Temporal Scalability
%-------------------------
FrameSkipTL1 = 1; %for Temporal level 1 (by skipping one frame)
[Finished_ScalableFramework] = ScalableFramework(FrameSet, FrameSkipTL1);
                              %Temporal Level 1, Variable Resolution Levels
disp('Finished_ScalableFramework');
```

**2. FrameExtractionFull_carpan.m**

```
%Carpann sample sequence
%Input
%------
%maxFrames - max no.of frames to be extracted.
```

```
%Output
%--------
%FrameSet - Collection of required frames

function[FrameSet] = FrameExtractionFull_carpan(maxFrames);

for countmax = 1:maxFrames %Importing Frames from the database
%%%for countmax = 1:FrameSkip:maxFrames
   if countmax < 10
       mystr = strcat('C:\Matlab7\work\carpann\Car00', int2str(countmax), '.bmp')
   else
       mystr = strcat('C:\Matlab7\work\carpann\Car0', int2str(countmax), '.bmp')
   end

    FrameSet(:,:,countmax) = double(rgb2gray(imread(mystr,'bmp')));
          % Resolution of the frame is 720x576 (704x576)
end
```

**3. FrameExtractionFull_Bird.m**

```
%SwanAndBird sample sequence
%Input
%------
%maxFrames - max no. of frames to be extracted.
%Output
%--------
%FrameSet - Collection of required frames

function[FrameSet] = FrameExtractionFull_Bird (maxFrames);

for countmax = 1:maxFrames %Importing Frames from the database
%%%for countmax = 1:FrameSkip:maxFrames
if countmax < 10
       mystr = strcat('C:\Matlab7\work\Bird\bird0', int2str(countmax), '.bmp')
   else
       mystr = strcat('C:\Matlab7\work\Bird\bird', int2str(countmax), '.bmp')
   end
    FrameSet(:,:,countmax) = double(rgb2gray(imread(mystr,'bmp')));
          % Resolution of the frame is 720x576 (704x576)
end
```

**4. FrameExtractionFull_Wcar.m**

```
%WhiteCar sample sequence
%Input
%------
%maxFrames - max no. of frames to be extracted.
```

```
%Output
%--------
%FrameSet - Collection of required frames

function[FrameSet] = FrameExtractionFull_Wcar (maxFrames);

for countmax = 1:maxFrames %Importing Frames from the database
%%%for countmax = 1:FrameSkip:maxFrames
if countmax < 10
        mystr = strcat('C:\Matlab7\work\WCar\Wcar0', int2str(countmax), '.bmp')
    else
        mystr = strcat('C:\Matlab7\work\WCar\Wcar', int2str(countmax), '.bmp')
    end
     FrameSet(:,:,countmax) = double(rgb2gray(imread(mystr,'bmp')));
          % Resolution of the frame is 720x576 (704x576)
end
```

**5. ScalableFramework.m**

```
%Scalable Framework
%Input
%------
%FrameSet – Sample image set
%FrameSkip - using temporal level 1 with one frame skip
%Output
%--------
%Finish_SF – Indicator of successful completion scalable operation

function [Finish_SF] = ScalableFramework(FrameSet, FrameSkip);

[m n max_frames] = size(FrameSet);

for frame_no = 1: FrameSkip: max_frames-FrameSkip %Temporal Skip of 1 frame

    %Reference Frame
    %--------------------
    if frame_no < max_frames
        O1 = FrameSet(:,:,frame_no);
   end


    %Prediction Frame
    %-------------------
    if frame_no < max_frames
        O2 = FrameSet(:,:,frame_no+FrameSkip);
    end
```

```
%Re-sizing the frames to appropriate resolution (in powers of two)
%--------------------------------------------------------------------------
m2 = 576;
n2 = 704;
O1 = O1(1:m2, 1:n2,:);
O2 = O2(1:m2, 1:n2,:);

% 2. Wavelet subband extraction and Multiresolution representation
% --------------------------------------------------------------------------
N = 2; %decomposition level
[A1_ref A2_ref A3_ref A1_pred A2_pred A3_pred] = SpatialScaling(N, O1, O2);

%Multiresolution Motion Estimation
%-----------------------------------------
%Simulation Configuration
%------------------------------
%1. Top-down approach
%2. Variable block size approach

%Resolution R1 (144x176)
%------------------------------
FrameI1 = A1_ref; %Reference Frame
FrameP1 = A1_pred; %Prediction / Current Frame
blockSize1 = 16;
p=16; %Search parameter

%Motion estimation using threshold and selective data transmission
[MotionVector1, MismatchBlocks1] = ME_ErrIdentification(FrameI1, FrameP1,
                                                 blockSize1, p, frame_no);
%MotionCompensation
%(1). Block replacement - the erroneous blocks are replaced by their original blocks
 PredictedFrame_R1_Blk = motionComp_BlkReplace(FrameI1, MotionVector1,
                                      blockSize1, ReplaceBlocks1, FrameP1);

%(2) Motion vector replacement – for mismatch blocks
 [mv_reducedArea1] = ME_SearchReduced(FrameI1, FrameP1, blockSize1, p,
                                      MismatchBlocks1, frame_no);
PredictedFrame_R1_MV = motionComp_MVReplace(FrameI1, MotionVector1,
                                      blockSize1, MismatchBlocks1, mv_reducedArea1);

%Estimated-Image Storage
%------------------------------
E_1R1 = PredictedFrame_R1_Blk/max(max(PredictedFrame_R1_Blk));
imwrite(E_1R1,strcat('C:\Matlab7\work\Result\R1\',int2str(frame_no),'_R1Blk.bmp'),'b
mp');
E_2R1 = PredictedFrame_R1_MV/max(max(PredictedFrame_R1_MV));
```

```
imwrite(E_2R1,strcat('C:\Matlab7\work\Result\R1\',int2str(frame_no),'_R1MV.bmp'),'
bmp');

%Resolution R2 (288x352)
%-----------------------------
  FrameI2 = A2_ref;
  FrameP2 = A2_pred;
  blockSize2 = 32;
  p2 = 34;

%MotionCompensation
%(1). Block replacement - the erroneous blocks are replaced by their original blocks
 PredictedFrame_R2_Blk = motionComp_BlkReplace(FrameI2, MotionVector1,
                                  blockSize2, MismatchBlocks1, FrameP2);
 %(2) Motion vector replacement – for mismatch blocks
 PredictedFrame_R2_MV = motionComp_MVReplace(FrameI2, MotionVector1,
                                  blockSize2,MismatchBlocks1, mv_reducedArea1);
 %Estimated-Image Storage
 %------------------------------
 E_1R2 = PredictedFrame_R2_Blk/max(max(PredictedFrame_R2_Blk));
   imwrite(E_1R2,strcat('C:\Matlab7\work\Result\R2\',int2str(frame_no),'_R2Blk.bmp'),
   'bmp');
 E_2R2 = PredictedFrame_R2_MV/max(max(PredictedFrame_R2_MV));
   imwrite(E_2R2,strcat('C:\Matlab7\work\Result\R2\',int2str(frame_no),'_R2MV.bmp')
   ,'bmp');

 %Resolution R3 (576x704)
 %------------------------------
    FrameI3 = A3_ref;
    FrameP3 = A3_pred;
    blockSize3 = 64;
    p3 = 66;

 %MotionCompensation
 %(1). Block replacement - the erroneous blocks are replaced by their original blocks
 PredictedFrame_R3_Blk = motionComp_BlkReplace(FrameI3, MotionVector1,
           blockSize3, MismatchBlocks1, FrameP3);
 %(2) Motion vector replacement – for mismatch blocks
 PredictedFrame_R3_MV = motionComp_MVReplace(FrameI3, MotionVector1,
           blockSize3,MismatchBlocks1, mv_reducedArea1);
  %Estimated-Image Storage
  %------------------------------
  E_1R3 = PredictedFrame_R3_Blk/max(max(PredictedFrame_R3_Blk));
        imwrite(E_1R3,strcat('C:\Matlab7\work\Result\R3\',int2str(frame_no),'_R3Blk.b
        mp'),'bmp');
  E_2R3 = PredictedFrame_R3_MV/max(max(PredictedFrame_R3_MV));
```

```
        imwrite(E_2R3,strcat('C:\Matlab7\work\Result\R3\',int2str(frame_no),'_R3MV.b
        mp'),'bmp');

   %PSNR Computation
   %------------------------
   %R1
   %---
   psnrR1Blk(frame_no) =  PSNR(FrameP1, PredictedFrame_R1_Blk);
   psnrR1MV(frame_no) = PSNR(FrameP1, PredictedFrame_R1_MV);
   %R2
   %---
   psnrR2Blk(frame_no) =  PSNR(FrameP2, PredictedFrame_R2_Blk);
   psnrR2MV(frame_no) =  PSNR(FrameP2, PredictedFrame_R2_MV);
   %R3
   %---
   psnrR3Blk(frame_no) =  PSNR (FrameP3, PredictedFrame_R3_Blk);
   psnrR3MV(frame_no) =  PSNR (FrameP3, PredictedFrame_R3_MV);

   disp('Frame No =')
   disp(frame_no)

end

 n1 = {'PSNR R1MV'};
 n2 = {'PSNR R1Blk'};
 n3 = {'PSNR R2MV'};
 n4 = {'PSNR R2Blk'};
 n5 = {'PSNR R3MV'};
 n6 = {'PSNR R3Blk'};

 Name1 = xlswrite('C:\MATLAB7\work\Result\Result',n1,'PSNR', 'C1');
 Name2 = xlswrite('C:\MATLAB7\work\Result\Result',n2,'PSNR', 'D1');
 Name3 = xlswrite('C:\MATLAB7\work\Result\Result',n3,'PSNR', 'E1');
 Name4 = xlswrite('C:\MATLAB7\work\Result\Result',n4,'PSNR', 'F1');
 Name5 = xlswrite('C:\MATLAB7\work\Result\Result',n5,'PSNR', 'G1');
 Name6 = xlswrite('C:\MATLAB7\work\Result\Result',n6,'PSNR', 'H1');
 Name10 = xlswrite('C:\MATLAB7\work\Result\Result',psnrR1MV','PSNR', 'C2:C48');
 Name20 = xlswrite('C:\MATLAB7\work\Result\Result',psnrR1Blk','PSNR', 'D2:D48');
 Name30 = xlswrite('C:\MATLAB7\work\Result\Result',psnrR2MV','PSNR', 'E2:E48');
 Name40 = xlswrite('C:\MATLAB7\work\Result\Result',psnrR2Blk','PSNR', 'F2:F48');
 Name50 = xlswrite('C:\MATLAB7\work\Result\Result',psnrR3MV','PSNR', 'G2:G48');
 Name60 = xlswrite('C:\MATLAB7\work\Result\Result',psnrR3Blk','PSNR', 'H2:H48');

Finish_SF = 'Finished SF';
```

**6. SpatialScaling.m**

```
%Wavelet decomposition and Multidimensional representation
%Simulation Configuration
%------------------------------
%1. Haar wavelet
%2. Total decomposition – Two
%Input
%------
%1. N - decomposition level
%2. Reference – Reference Frame
%3. Prediction – Prediction Frame
%Output
%--------
%1. A1_ref, A2_ref, A3_ref – Reference Frame of decomposed resolution - R1, R2, and
%R3 respectively
%2. A1_pred, A2_pred, A3_pred – Prediction Frame of decomposed resolution - R1, R2,
%and R3 respectively

function[A1_ref, A2_ref, A3_ref, A1_pred, A2_pred, A3_pred] = SpatialScaling(N,
Reference, Prediction);

wname = 'haar'; % wavelet used
NL = N; %Decomposition level for lower resolution
NM = N-1; %Decomposition level for middle resolution

%Reference Frame - Lower resolution (R1)
[C_R,S] = wavedec2(Reference,NL,wname); %2D wavelet transform

A1_ref = appcoef2(C_R, S, wname, NL); %Approximation extraction of Level2
[H1_ref V1_ref D1_ref] = detcoef2('all', C_R, S, NL);
%Horizontal, vertical and Diagonal coefficients – Level2
%Reference Frame - Middle resolution (R2)
A2_ref = appcoef2(C_R, S, wname, NM); %Approximation extraction of Level1
[H2_ref V2_ref D2_ref] = detcoef2('all', C_R, S, NM);
%Horizontal, vertical and Diagonal coefficients – Level1

%Reference Frame - Higher resolution (R3)
A3_ref = appcoef2(C_R, S, wname, 0); %Approximation extraction of Level0

%Prediction Frame - Lower resolution (R1)
[C_P,S] = wavedec2(Prediction,NL,wname); %2D wavelet transform
A1_pred = appcoef2(C_P, S, wname, NL); %Approximation extraction of Level2
[H1_pred V1_pred D1_pred] = detcoef2('all', C_P, S, NL);
%Horizontal, vertical and Diagonal coefficients – Level2

%Prediction Frame - Middle resolution (R2)
```

```
A2_pred = appcoef2(C_P, S, wname, NM); %Approximation extraction of Level1
[H2_pred V2_pred D2_pred] = detcoef2('all', C_P, S, NM);
%Horizontal, vertical and Diagonal coefficients – Level1

%Prediction Frame - Higher resolution (R3)
A3_pred = appcoef2(C_P, S, wname, 0); %Approximation extraction of Level0
```

**7. ME_ErrIdentification.m**

```
%Motion Estimation
%----------------------
%Computes motion vectors using exhaustive search method
%
% Input
%-------
%1. FrameP - The image for which we want to find motion vectors
%2. FrameI - The reference image
%3. blockSize - Size of the macroblock
%4. p - Search parameter
%5. frameNo – pointer for frame number
%Ouput
%---------
%1. MotionVector - the displacement motion associated with each macroblock
%2. MismatchBlks – the identified error blocks or mismatches using selective data
%transmission thresholds

function [MotionVector MismatchBlks] = ME_ErrIdentification(FrameI, FrameP,
blockSize, p, frameNo)

[row col] = size(FrameI);
%For Error Identification function
sizeImg = [row col];

%Initialisations
minCost = [];
displacements = zeros(2,row*col/blockSize^2);
maxSAD = (256*blockSize*blockSize)+1; %maximum error magnitude of a block
count = 1;

for i = 1 : blockSize : row-blockSize+1
   for j = 1 : blockSize : col-blockSize+1

     BlkP = FrameP(i:i+blockSize-1,j:j+blockSize-1);
                  %Current block in PredictionFrame

      %Search window
```

```
    for m = -p : p     %vertical coordinates
        for n = -p : p  %horizontal coordinates
            SearchAreaI = i + m;
            SearchAreaJ = j + n;

            if (SearchAreaI  < 1 || SearchAreaI +blockSize-1 > row ...
                    || SearchAreaJ < 1 || SearchAreaJ +blocksSize-1 > col)
                continue;
            end

            SearchBlk = FrameI(SearchAreaI: SearchAreaI +blockSize-1, SearchAreaJ:
                                            SearchAreaJ +blockSize-1);
            %Reference block within in the search area

            sad(i,j) = sum(sum(abs(BlkP - SearchBlk)));
            if sad(i,j) < maxSAD
                maxSAD = sad(i,j);
                newI = m;
                newJ = n;
            end


        end
    end

    minCost(count) = maxSAD;

   %For error identification
    Blk(1,count) = i;
    Blk(2,count) = j;
    displacements(1,count) = newI;    % vector representing row coordinate
    displacements(2,count) = newJ;    % vector representing column coordinate
    count = count + 1;
    maxSAD = (256*blockSize*blockSize)+1;
  end
end

Block = Blk; %For error identifcation
MotionVector = displacements;

%Error Identification using selective data transmission Thresholds
[MismatchBlks]= ErrorIdentification(minCost, Block, sizeImg, frameNo);
```

**8. ErrorIdentification.m**

```
%Error Identification using two parameters
```

```
%1. Threshold  2. WorstBlock factor
%Selective Data Transmission
% - Error Blocks are identified for updates
%
%Input
%------
%1. minCost – error magnitude of the macroblock
%2. Blk – marcoblock
%3. sizeImg – size of the frame
%4. frameNo – frame number identity
%
%Output
%---------
%1. MismatchBlks – error blocks or mismatches

function [MismatchBlks]= ErrorCorrection(minCost, Blk, sizeImg, frameNo)

row = sizeImg(1,1);
col = sizeImg(1,2);

%ThFactor = 40; %Percentage calculation for threshold
%WorstBlockfactor = 100; %Percentage of worst mismatch blocks within the
                                    %threshold

[SortErrValue ind] = sort(minCost, 'descend');
range = size(SortErrValue,2);
MismatchBlk = zeros(1,range);
MismatchBlks = zeros(1,range);

MaxSAD = SortErrValue(1);
%Fixation of the Threshold Th
%-----------------------------------
%Threshold Th is a value above which macroblocks are to be replaced.
% which is the  of the highest SAD value

%%%%%%Th = round((MaxSAD * ThFactor)/100);
Th = 5000; %1; %5000;%15500; %for Carpan sample
%Th = 3000;%7300; %7359.645161; %For Bird Sample
%Th = 3000; %7000; %12500%For WhiteCar Sample


%Fixation of K parameter
%------------------------------
%K determines the number of macroblocks to be replaced (above the threshold)

for rr = 1:range
```

```
    if (SortErrValue(rr) > Th)
        BlkReplaceInd(rr) = SortErrValue(rr);
    end
end

BlkInd = size(BlkReplaceInd,2); %Number of blocks AVAILABLE above the threshold

%K factor
%----------
BlockCount = round((BlkInd * Kfactor)/100); %Number of blocks to be replaced above
%the threshold based on Kfactor

% %Making it constant
% BlockCount = 8; %25; %8;
  %BlockCount = 80;%8; %Bird
%   BlockCount = 7; %WhiteCar
% if BlkInd < BlockCount
%     BlockCount = BlkInd;
% end

%Replacement Block List
%----------------------
%The blocks to be replaced are listed in order of the block count in a Frame
%i.e. mbCount.

    RPList = sort(ind(1:BlockCount),'ascend');
    for s = 1:BlockCount
        ReplaceBlk(RPList(s)) = RPList(s);
    end
    ReplaceBlks = ReplaceBlk;
```

**9. motionComp_BlkReplace.m**

```
%MotionCompensation by resending intra blocks for the mismatch blocks.
%
%Input
%-----
%1. FrameI - Reference Frame
%2. MotionVector - motion estimates from lower resolution R1 (144x176)
%5. blockSize - size of the macroblock
%4. MismatchBlocks – the error blocks
%5. FrameP – Predicted Frame
%
%Output
%------
%1. BlkReplacedImg - Motion Compensated image with mismatches are intracoded.
```

```
function BlkReplacedImg = motionComp_BlkReplace(FrameI, MotionVector, blockSize,
MismatchBlocks, FrameP)

[row col] = size(imgI);

count = 1;
for i = 1 : blockSize : row-blockSize+1
   for j = 1 : blockSize : col-blockSize+1


      di = MotionVector(1,count);
      dj = MotionVector(2,count);

      newI = i + di;
      newJ = j + dj;

      if count == MismatchBlocks(count)
      EstImage(i:i+blockSize-1,j:j+blockSize-1) = FrameP(i:i+blockSize-
                                                   1,j:j+blockSize-1);
      else
      EstImage(i:i+blockSize-1,j:j+blockSize-1) = FrameI(newI:newI+blockSize-1,
                                                   newJ:newJ+blockSize-1);
      end

      count = count + 1;
   end
end

BlkReplacedImg = EstImage;
```

**10. ME_SearchReduced.m**

```
%Refined motion estimation with reduced search area
%The new motion vector is used as replacement for the mismatch blocks
%
% Input
%-------
%1. FrameP - The image for which we want to find motion vectors
%2. FrameI - The reference image
%3. blockSize - Size of the macroblock
%4. p - Search parameter
%5. frameNo – pointer for frame number
%
%Ouput
%---------
%1. ReplaceMVs - the new displacement motion of the mismatch blocks
```

```
function [ReplaceMVs] = ME_SearchReduced(FrameI, FrameP, blockSize, p,
MismatchBlocks, frameNo)

%Initialisations
[row col] = size(imgI);
minCost = [];
displacements = zeros(2,row*col/blockSize^2);
MotionVector = zeros(2,row*col/blockSize^2);
maxSAD = (256*blockSize*blockSize)+1;

%Reducing search area
%--------------------------
p = p - 2; %parameter in order of 2^(m)

count = 1;
for i = 1 : blockSize : row-blockSize+1
    for j = 1 : blockSize : col-blockSize+1

     if count == MismatchBlocks(count)

        BlkP = FrameP(i:i+blockSize-1,j:j+blockSize-1);

        for m = -p : p
            for n = -p : p
                SearchAreaI = i + m;
                SearchAreaJ = j + n;
                if ( SearchAreaI < 1 || SearchAreaI+blockSize-1 > row ...
                        || SearchAreaJ < 1 || SearchAreaJ+blockSize-1 > col)
                    continue;
                end
                SearchBlk = FrameI(SearchAreaI:SearchAreaI+blockSize-1,
                                            SearchAreaJ:SearchAreaJ+blockSize-1);

                sad(i,j) = sum(sum(abs(BlkP - SearchBlk)));

                if sad(i,j) < maxSAD
                    maxSAD = sad(i,j);
                    newI = m;
                    newJ = n;
                end
            end
        end

       minCost(count) = maxSAD;
```

```
      Blk(1,count) = i;
      Blk(2,count) = j;
      displacements(1,count) = newJ;    % vector representing row coordinate
      displacements(2,count) = newI;    % vector representing column coordinate

    end
      count = count + 1;
        maxSAD = (256*blockSize*blockSize)+1;
   end
  end
ReplaceMVs = displacements;
```

**11. motionComp_MVReplace.m**

```
%
%Motion Compensation by replacing motion vectors for the mismatch blocks
%The motion vector for the mismatches are replaced with new motion vector found
%using reduced search area
%
%Input
%-----
%1. FrameI - Reference Image
%2. MotionVector - motion prediction from lower resolution R1
%3. blockSize - size of the macroblock
%4. MismatchBlocks – the identified error blocks
%5. RefinedMV – the new displacements
%
%Output
%------
%1. imgMismatchComp - Motion Compensated image with mismatches replaced
%with new motion vectors.

function imgMismatchComp = motionComp_MVReplace(FrameI, MotionVector,
blockSize, MismatchBlocks, RefinedMV)

[row col] = size(imgI);

count = 1;
for i = 1 : blockSize : row-blockSize+1
   for j = 1 : blockSize : col-blockSize+1

      if count == MismatchBlocks(count)
         di = RefinedMV(1,count);
         dj = RefinedMV(2,count);
      else
         di = MotionVector(1,count);
```

```
        dj = MotionVector(2,count);
      end

      newI = i + di;
      newJ = j + dj;

      EstImg(i:i+blockSize-1,j:j+blockSize-1) = FrameI(newI:newI+blockSize-1,
                                                       newJ:newJ+blockSize-1);
      count = count + 1;
   end
end
imgMismatchComp = EstImg;
```

**12. PSNR.m**

```
%PSNR Calculation
% -for performance evaluation of the estimated images
%Input
%--------
%1. FrameP - original prediction image
%2. PredictedFrame - the estimated image
% Ouput
%---------
%2. psnrValue - the estimated PSNR of the predicted image

function psnrValue = PSNR(FrameP, PredictedFrame)

%Initialisation
[row col] = size(FrameP);
summation = 0;
n = 255; %maximum colour value of a pixel

for i = 1:row
   for j = 1:col
      summation = summation + (FrameP(i,j) - PredictedFrame(i,j))^2;
   end
end

%numerator = sum(sum((FrameP - PredictedFrame)^2));

mse = summation / (row*col);

psnrValue = 10*log10(n*n/mse);
```