Appendix A

Source Code

A1. Main Program

```
% ------------------------------------------------------
%
% This program is used for MSc EE UTP thesis
% Program to do DCT-based and DTCWT-based video codec
%
%
% Supervisor : Dr. Vooi Voon Yap
% vooiv@petronas.com.my
%
% Unan Y Oktiawati
% G2010342
% unan_yusmaniar@utp.edu.my
% last modified : mar 07
% ------------------------------------------------------

% initialization condition

close all;
clear all;

% call the source file

imageName = 'hanif'; % name of file

% initialization value

err = 0;

% define variable needed

mbSize = 16; % macro block size
p = 7; % search parameter

% activate timer for counting time consuming information

tic; % start timer

% start of the program

% initialization part

for i = 1:30 % i = 1:n n=number of frames

    imgINumber = i; % I frame
    imgPNumber = i+2; % P frame

% Read the sequences

    if imgINumber < 10
      imgIFile =
      sprintf('F:thesis/tried/BlockMatchingAlgoMPEG/%s/gray/%s00%d.ras',
      imageName, imageName, imgINumber);
```

```
      % state the location of the sequences file
    elseif imgINumber < 100
      imgIFile =
      sprintf('F:thesis/tried/BlockMatchingAlgoMPEG/%s/gray/%s0%d.ras',
      imageName, imageName, imgINumber);

    end

    if imgPNumber < 10
      imgPFile =
      sprintf('F:thesis/tried/BlockMatchingAlgoMPEG/%s/gray/%s00%d.ras',
      imageName, imageName, imgPNumber);

    elseif imgPNumber < 100
      imgPFile =
      sprintf('F:thesis/tried/BlockMatchingAlgoMPEG/%s/gray/%s0%d.ras',
      imageName, imageName, imgPNumber);

    end

    [a,map]=imread(imgIFile);
    imgI = double(imread(imgIFile));
    imgP = double(imread(imgPFile));

% changing the frame into suitable size

    imgI = imresize(imgI,[100 120]); % [a b]= size of frame
    imgP = imresize(imgP,[100 120]);

% Transformation part

% 2D DTCWT decomposition level 1 using filter 'near_sym_b' and 'qshift_b'

[YlI,YhI] = dtwavexfm2(imgI,1,'near_sym_b','qshift_b');
[YlP,YhP] = dtwavexfm2(imgP,1,'near_sym_b','qshift_b');

% Discrete Cosine Transform (DCT)

imgP=dct(imgP);
imgI=dct(imgI);

% Motion Estimation

% DTCWT+ARPS Motion Estimation

[motionVect1, computations1] = motionEstARPS(YlP,YlI,mbSize,p);

% ARPS Motion Estimation

[motionVect, computations] = motionEstARPS(imgP,imgI,mbSize,p);

% Motion Compensation

imgComp1 = motionComp(YlI, motionVect1, mbSize);
imgComp = motionComp(imgI, motionVect, mbSize);
```

```
% For DTCWT, size of imgComp1 should be agreed with YlI

imgComp1 = imresize(imgComp1,size(YlI));

% inverse transform

% 2D DTCWT reconstruction

imgDTCWT = dtwaveifm2(imgComp1,YhI,'near_sym_b','qshift_b');

% inverse DCT

imgComp=idct(imgComp);

% Stop the timer

time=toc; % time=time consuming

% Calculate PSNR value

DTCWTpsnr(i)=imgPSNR(imgDTCWT, imgI, 255);
Comppsnr(i)=imgPSNR(imgComp, imgI, 255);

% Display results

imshow(uint8(imgDTCWT));
figure; imshow(uint8(imgComp));

% Create an avi file

% For DCT-based

imgComp=uint8(imgComp);
A(i)=immovie(imgComp,map);
movie2avi(A,'hanifA.avi');

% For DTCWT-based

imgDTCWT=uint8(imgDTCWT);
B(i)=immovie(imgDTCWT,map);
movie2avi(B,'hanifB.avi');

end

% calculating the difference PSNR

a=DTCWTpsnr-Comppsnr;
save a

% For Visual comparison purpose

figure;
subplot(131);imgI=uint8(imgI);imshow(imgI);title('imgI');
subplot(133);imgDTCWT=uint8(imgDTCWT);imshow(imgDTCWT);
```

```
title('ARPS+DTCWT');
subplot(132);imgComp=uint8(imgComp);imshow(imgComp);title('ARPS');

% Plot figure of PSNR

figure;
i=1:i;
plot(i,DTCWTpsnr,i,Comppsnr);

% completed with title, legend and label

title('\it{Hanif}');
legend('ARPS+DTCWT','ARPS');
xlabel('frame no.');ylabel('PSNR');
```

A2. DTCWT

Kingsbury's Dual-Tree Complex Wavelet Transform available from ngk@eng.cam.ac.uk.

The function:

```
function Z = dtwaveifm2(Yl,Yh,biort,qshift,gain_mask);
```

This function is used to perform an n-level dual-tree complex wavelet (DTCWT) 2-D reconstruction by Nick Kingsbury and Cian Shaffrey Cambridge University, May 2002.

Z is the reconstructed real image matrix

Yl is the real lowpass image from the final level

Yh is a cell array containing the 6 complex highpass subimages for each level.

gain_mask is Gain to be applied to each subband.

Used variable in this thesis are

- 'near_sym_b' or Near-Symmetric 13,19 tap filters for biort, biorthogonal filters;
- 'qshift_b' or Q-Shift 14,14 tap filters for qshift, quarter sample shift;
- 'ones(6,length(Yh))' as the default value of gain_mask.

The DTCWT function

```
% initialization value

% state the No of levels.
% check the gain_mask value, if it is not defined then use the default
value of gain_mask
% check the input, should be in correct form or give error message

% do c2q function

% Do even Qshift filters on columns.
% Do even Qshift filters on rows.
% Check size of Z and crop as required

%check to see if this result needs to be cropped for the rows
%check to see if this result needs to be cropped for the cols

% do c2q function

% Do odd top-level filters on columns.
% Do odd top-level filters on rows.
```

```
% c2q function

% Scale by gain and convert from complex w(:,:,1:2) to real quad-numbers
in z.

% Arrange pixels from the real and imag parts of the 2 subbands into 4
separate subimages (A, B, C and D).

%  A----B
%  |    |
%  |    |
%  C----D

% Recover each of the 4 corners of the quads.
```

A3. Inverse DTCWT

Kingsbury's Dual-Tree Complex Wavelet Transform available from ngk@eng.cam.ac.uk.

the function :

```
function [Yl,Yh,Yscale] = dtwavexfm2(X,nlevels,biort,qshift);
```

Function to perform a n-level DTCWT-2D decompostion on a 2D matrix X by Nick Kingsbury and Cian Shaffrey, Cambridge University, Sept 2001

X is 2D real matrix/Image.
nlevels is No. of levels of wavelet decomposition.
Yl is the real lowpass image from the final level.
Yh is a cell array containing the 6 complex highpass subimages for each level.
Yscale is an OPTIONAL output argument, that is a cell array containing real lowpass coefficients for every scale.

Variables used in this thesis are

- 'near_sym_b' => Near-Symmetric 13,19 tap filters for biort, biorthogonal filters;
- 'qshift_b' => Q-Shift 14,14 tap filters for qshift, quarter sample shift.

The inverse DTCWT function

```
% Check if the inputs are in correct form or give error message

% Check to see if the image is odd in size, if so an extra row/column
will be added to the bottom/right of the image

% Initialization

% if sx(1) is not divisable by 2 then we need to extend X by adding a row
at the bottom
% Any further extension will be done in due course.
%if sx(2) is not divisable by 2 then we need to extend X by adding a
coulomb to the left
% Any further extension will be done in due course.

% Do odd top-level filters on cols.
% Do odd top-level filters on rows.
```

```
% do q2c function

% Extend by 2 rows if no. of rows of LoLo are divisable by 4;
% Extend by 2 cols if no. of cols of LoLo are divisable by 4;

% Do even Qshift filters on rows.
% Do even Qshift filters on columns.

% do q2c function

q2c function

function z = q2c(y)

% Convert from quads in y to complex numbers in z.

% Arrange pixels from the corners of the quads into 2 subimages of
alternate real and imaginary pixels.
%  a----b
%  |    |
%  |    |
%  c----d

% Combine (a,b) and (d,c) to form two complex subimages.
% Form the 2 subbands in z.
```

A4. Filter

Kingsbury's Dual-Tree Complex Wavelet Transform available from ngk@eng.cam.ac.uk.

This function is used to filter the columns of image X using the two filters ha and hb which is the reverse of ha. This function is introduced by Cian Shaffrey, Nick Kingsbury, Cambridge University, August 2000. Modified to be fast if X = 0, May 2002.

The function :

```
function Y = colifilt(X, ha, hb)
```

ha operates on the odd samples of X and hb on the even samples.
Both filters should be even length, and h should be approx linear phase with a quarter sample advance from its mid pt (ie |h(m/2)| > |h(m/2 + 1)|).

The output is interpolated by two from the input sample rate and the results from the two filters, Ya and Yb, are interleaved to give Y.
Symmetric extension with repeated end samples is used on the composite X columns before each filter is applied.

```
% if m/2 is even, so set up t to start on d samples.
% Set up vector for symmetric extension of X with repeated end samples.
% Use 'reflect' so r < m2 works OK.

% if m/2 is odd, so set up t to start on b samples.
% Set up vector for symmetric extension of X with repeated end samples.
% Use 'reflect' so r < m2 works OK.
```

A5. PSNR

Based on formula in equation 3.2 and equation 3.3

$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(C_{ij} - R_{ij}\right)^2 \tag{3.2}$$

$$PSNR = 10\log_{10}\left[\frac{(peak\ to\ peak\ value\ of\ original\ data)^2}{MSE}\right] \tag{3.3}$$

```
% Computes image's PSNR
%
% Input
%   A : The original image
%   B : The reconstructed image
%   n : the peak value possible of any pixel in the images
%
% Ouput
%   psnr : The image's PSNR
%
% last updated by [Un@N] 01 mar 07

function psnr = imgPSNR(A, B, n)

[row col] = size(A);

err = 0;

for a = 1:row
    for b = 1:col
        err = err + (A(a,b) - B(a,b))^2;
    end
end

MSE = err / (row*col);

psnr = 10*log10(n*n/MSE);
```

A6. ARPS

This function can be downloaded from

http://mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=8761&objectType=file

```
% Computes motion vectors using Adaptive Rood Pattern Search method
%
% Based on the paper by Yao Nie, and Kai-Kuang Ma
% IEEE Trans. on Image Processing
% Volume 11 Number 12, December 2002 :  Pages 1442:1448
%
% Input
%   imgP : The image for which we want to find motion vectors
%   imgI : The reference image
%   mbSize : Size of the macroblock
%   p : Search parameter  (read literature to find what this means)
%
% Ouput
%   motionVect : the motion vectors for each integral macroblock in imgP
%   ARPScomputations: The average number of points searched for a
macroblock
%
% Written by Aroh Barjatya
* with modification by unan mar 07

function [motionVect, ARPScomputations] = motionEstARPS(imgP, imgI,
mbSize, p)

% define size of image

% The index points for Small Diamond search pattern

% Storing the positions of points where the checking has been already
done in a matrix that is initialized to zero. As one point is checked,
set the corresponding element in the matrix to one.

% start off from the top left of the image
% walk in steps of mbSize
% mbCount will keep track of how many blocks have been evaluated

% the Adaptive Rood Pattern search starts
% scanning in raster order

% if in the left most column then make sure that the LDSP is done with
stepSize = 2
% if the point due to motion vector is one of the LDSP points then no
need to calculate it again
% check at the rood pattern 5 points
% check 6 points

% The index points for first and only Large Diamond search pattern
```

```
% do the LDSP
% row/Vert co-ordinate for ref block
% col/Horizontal co-ordinate

% outside image boundary
% center point already calculated

% The doneFlag is set to 1 when the minimum is at the center of the
diamond

% row/Vert co-ordinate for ref block
% col/Horizontal co-ordinate

% row co-ordinate for the vector
% col co-ordinate for the vector
```

Appendix B

Subjective Visual Quality

Value	*Rating*	*Description*
1	Excellent	An extremely high quality video.
2	Fine	A high quality video. Artifacts are not objectionable.
3	Passable	An acceptable quality video. Artifacts are not objectionable.
4	Marginal	A poor quality video. Artifacts are somewhat objectionable.
5	Inferior	A very poor video, but still acceptable. Artifacts are visible and objectionable.
6	Unusable	A bad video that it is unacceptable.

Give a value to the video displayed on the mobile phone screen using the rating scale given in the above table. Record your value on a score sheet. Give the value on appropriate column.

Thank you for your cooperation.

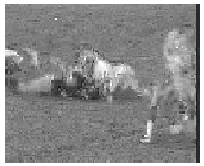
B1. Subjective Visual Quality Form

(page 1)

Caltrain A

Caltrain B

Football A



Football B

Value	*Rating*
1	Excellent
2	Fine
3	Passable
4	Marginal
5	Inferior
6	Unusable

Susie A

Susie B

Hanif A

Hanif B

Missa A

Missa B

B2. Subjective Visual Quality Form

(page 2)

Table B1. Frequency of Subjective Visual Quality

Value	Hanif A	Hanif B
1	2	11
2	7	14
3	19	13
4	15	8
5	3	2
6	4	2

Value	Susie A	Susie B
1	0	3
2	10	22
3	17	14
4	18	5
5	4	4
6	0	1

Value	Missa A	Missa B
1	0	1
2	8	18
3	22	19
4	16	7
5	3	5
6	1	0

Value	Football A	Football B
1	3	3
2	6	7
3	2	13
4	6	8
5	14	13
6	19	6

Value	Caltrain A	Caltrain B
1	3	1
2	4	15
3	18	9
4	11	10
5	10	13
6	4	2

Appendix C

Data Result

Table C1. Data Results

Sequence	Moving area (%)	File Size (kB)	Method	File Size (kB)	Compression (%)	Computation
Caltrain	91.64	97	DCT	80	17.525	9.763
			DTCWT	77	20.619	10.071
Football	94.99	197	DCT	111	43.655	10.724
			DTCWT	86	56.345	10.870
Hanif	84.13	60	DCT	48	20.000	6.811
			DTCWT	42	30.000	6.8963
Missa	84.7	29	DCT	24	17.241	7.328
			DTCWT	22	24.138	8.649
Susie	85.87	54	DCT	41	24.074	5.903
			DTCWT	35	35.185	5.812

Sequence	PSNR (dB)	Time (s)	delta time (s)	Memory Usage (kB)	delta Memory Usage (kB)
Caltrain	21.163	8.352	5.448	2387.4	151.4
	23.149	8.800		2236.0	
Football	20.577	8.304	0.817	3780.0	897.6
	21.810	7.487		2882.4	
Hanif	31.610	7.195	1.381	828.0	60.0
	31.977	7.576		768.0	
Missa	34.187	7.274	3.750	1428.0	216.0
	36.164	8.024		1212.0	
Susie	34.069	6.250	0.166	2965.0	37.0
	35.674	6.084		2928.0	

Appendix D

Publications

"Video Compression using Dual Tree Complex Wavelet Transform" in Proceeding of IEEE International Conference on Intelligent and Advance System (ICIAS), Kuala Lumpur, Malaysia, 2007.

"Evaluating the Effects of the Dual Tree Complex Wavelet Transform and the Adaptive Rood Pattern Search on a Video Codec" in Proceeding of IEEE International Conference on Industrial Electronics and Applications (ICIEA), Singapore, 2008.