Wavelet Transforms in theTMS320C55x

Cesar Iovescu *C5000 Applications*

ABSTRACT

Wavelets have been developed to analyze the frequency components of a signal according to a scale. They provide more information than the Fourier transform for signals which have discontinuities or sharp spikes.

This application report briefly presents the history of the wavelet, starting with Fourier, and describes the implementation of the wavelet transform using filter banks in the image processing field. At the end of this report are some of the most common applications, such as edge detection, noise removal, decomposition and reconstruction. They are illustrated using the TMS320C55x™ Imaging Library (C55x™ IMGLIB), provided in the TI web site at www.ti.com. The code is implemented on the TMS320C55x.

Contents

1 Wavelets, an Introduction . . . 3
1.1 Fourier Transform . . . 3
1.2 Short Time Fourier Transform . . . 3
1.3 Wavelet Transform . . . 5
2 Discrete Wavelet Transform . . . 5
2.1 Wavelets and Perfect Reconstruction Filter Banks . . . 7
2.2 Wavelet Filter Bank Implementation . . . 8
3 Wavelets Image Processing . . . 11
3.1 Wavelet Decomposition of Images . . . 11
4 Wavelets Applications . . . 13
4.1 One Dimension Wavelet Applications . . . 13
4.1.1 Discontinuity Detection Example . . . 13
4.1.2 Noise Removal Example . . . 14
4.1.3 1-D Perfect Decomposition and Reconstruction Example . . . 16
4.2 Two Dimension Wavelet Applications . . . 17
4.2.1 2-D Perfect Decomposition and Reconstruction Example . . . 17
4.2.2 Edge Detection Example . . . 18
5 References . . . 19
Appendix A Fourier Transform . . . 20
A.1 Fourier Transforms . . . 20
A.2 Window Fourier Transforms . . . 20

Appendix B Wavelet Transform 21
B.1 Continuous Wavelet Transforms 21
B.2 Discrete Wavelets 22
B.2.1 Multiresolution Analysis and Scaling Function 23
B.2.2 Orthogonal Wavelets Bases 24
B.3 Wavelet and Perfect Reconstruction Filter Bank 25
Appendix C Wavelet Functions API 27

List of Figures

Figure 1. Sinusoids With Two Deltas 4
Figure 2. Fourier Transform of Signal in Figure 1 With Different Window Size 4
Figure 3. Continuous Wavelet Transform of Signal Shown in Figure 1 5
Figure 4. Discrete Wavelet Transform 5
Figure 5. A Two-Level Wavelet Decomposition 7
Figure 6. A Two-Level Wavelet Reconstruction 7
Figure 7. Pyramid Packet 7
Figure 8. Wavelet Packet Decomposition 8
Figure 9. Filter Bank Algorithm in Matrix Format 9
Figure 10. Reconstruction From Figure 9 10
Figure 11. Altered Reconstruction From Figure 9 10
Figure 12. Original Image One-Level 2-D Decomposition 11
Figure 13. Three Popular Wavelet Decomposition Structures on Image: (a) Pyramid, (b) Spacl, (c) Wavelet Packet 11
Figure 14. (a) Original Image (b) Three-Level Pyramid Structure Decomposition 12
Figure 15. Input Signal 13
Figure 16. Zoom of the Discontinuity of the Sinusoidal Input Signal 13
Figure 17. One-Level Decomposition of the Discontinue Signal 14
Figure 18. Noisy Signal 14
Figure 19. One-Level Decompostion of the Noisy Signal 15
Figure 20. Threshold Applied to the High-Pass Channel 15
Figure 21. Reconstructed Signal 15
Figure 22. Level 3 Pyramid Decompostion of the Original Sine Wave Signal 16
Figure 23. LLL3 (left) and HLL3 (right) Components of Three-Level Decomposition 16
Figure 24. Reconstructed Error 17
Figure 25. Image Used in the 2D Wavelet Application 17
Figure 26. One-Level Decomposed and Reconstructed Image 18
Figure 27. Picture Used in Edge Detection Application 18
Figure 28. 2-D One-Level Decomposition Detects the Edges 18
Figure B–1. Mexican Hat Wavelet and Its Fourier Transform 21
Figure B–2. Localization of the Discrete Wavelets in the Time-Scale Space on a Dyadic Grid 23
Figure B–3. A Two-Level Wavelet Decomposition and Reconstruction 26

List of Tables

Table 1. Daubechies (p =2,3) Wavelet Coefficients 6
Table C–1. Wavelet Functions Written in C Code 27

1 Wavelets, an Introduction

Whether we like it or not we are living in a world of signals. Nature is talking to us with signals: light, sounds… Men are talking to each other with signals: music, TV, phones…

The human body is equipped to survive in this world of signals with sensors such as eyes and ears, which are able to receive and process these signals. Consider, for instance, our ears: they can discriminate the volume and tone of a voice. Most of the information our ears process from a signal is in the frequency content of the signal.

Scientists have developed mathematical methods to imitate the processing performed by our body and extract the frequency information contained in a signal. These mathematical algorithms are called *transforms* and the most popular among them is the *Fourier Transform*.

1.1 Fourier Transform

Jean Baptiste Joseph Fourier was a French scientist who lived in the early 1800's. He studied the principles of Heat Transfer and developed the Fourier Transform in order to solve the partial differential equations involved in his research. The Fourier Transform can decompose any periodic function into a linear combination of sines and cosines. The coefficients of the sines and cosines are the frequency components of the signal. This concept provided the foundation for Frequency Domain Analysis.

The Fourier transform gives us information about the behavior of a function in the frequency domain. However, it does not provide us any information about how the function behaves in the time domain. This means that although we might be able to determine all the frequencies present in a signal, we do not know when they are present. This is an important problem for signals whose behavior changes with time. These signals are called non-stationary signals.

Different algorithms have been developed to analyze non-stationary signals, and represent them in the time and frequency domain at the same time. These algorithms use either the short time Fourier transform or the wavelet transform method. The following sections discuss these methods.

1.2 Short Time Fourier Transform

The first method is to cut the signal into slices in time and then examine the frequency content of each of these slices. The short time Fourier transform (STFT) uses this concept. It is clear that analyzing a signal this way gives more information about the when and where of different frequency components, but it leads to a fundamental problem as well: how to cut the signal?

Heisenberg's uncertainty principle was discovered in the quantum physics area. The principle describes that you can not measure both the position and velocity of a particle exactly. The same kind of phenomena was found in our signal processing area: it is impossible to know the exact frequency and the exact time of occurrence of this frequency in a signal. In other words, a signal can simply not be represented as a point in the time-frequency space. The uncertainty principle shows that it is very important how one cuts the signal.

This example shows the uncertainty principle. The signal

$$f(t) = \sin 2\pi v_1 t + \sin 2\pi v_2 t + K\left[\delta\left(t - t_1\right) + \delta\left(t - t_2\right)\right]$$

consists of two sinusoids at frequencies of V_1 = 500 Hz and V_2 = 1000 Hz and two delta functions occurring at t_1 = 192 ms and t_2 = 196 ms. Figure 1shows 2048 samples of the signal containing the two deltas.

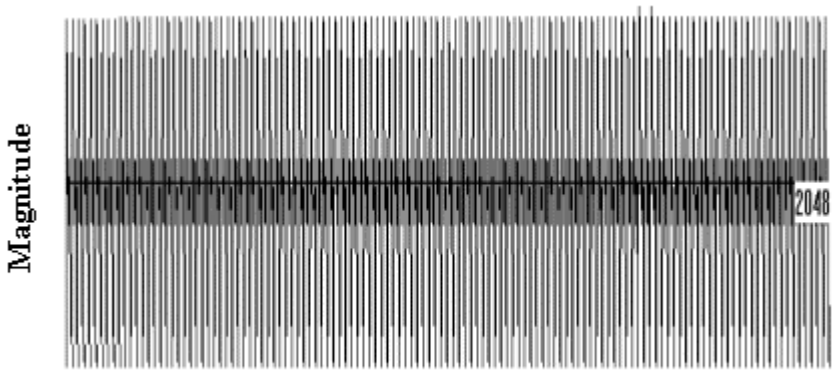


Figure 1. Sinusoids With Two Deltas

We choose different sizes of cuts to do the Fourier transform. Figure 2 shows the result of the series of transforms. Since the delta functions are separated by 32 samples, window sizes equal to or greater than 32 samples are not narrow enough to resolve the delta functions. On the other hand, a large window is effective to separate the two sinusoids with different frequencies. The shortest window whose size is 16 has very good time localization and separates the two deltas while losing resolution of the two sinusoids. The longest window with 128 points has high frequency resolution and separates the two sinusoids very clearly, but causes blur of the two deltas.

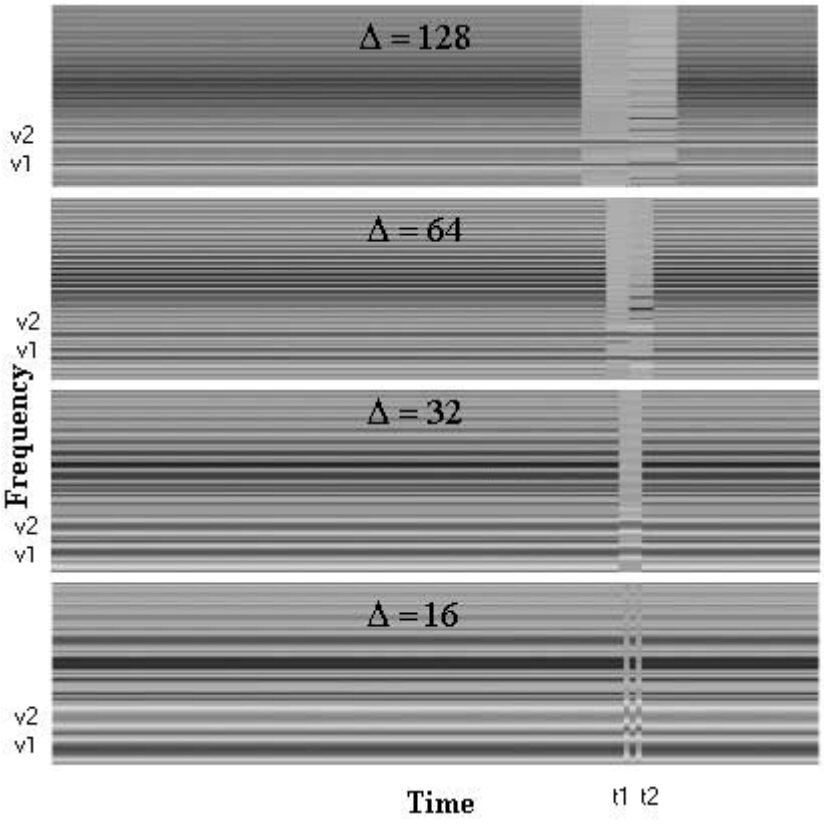


Figure 2. Fourier Transform of Signal in Figure 1 With Different Window Size

1.3 Wavelet Transform

The second method to analyze non-stationary signals is to first filter different frequency bands, cut these bands into slices in time, and then analyze them.

The *wavelet transform* uses this approach. The *wavelet transform* or *wavelet analysis* is probably the most recent solution to overcome the shortcomings of the Fourier transform. In wavelet analysis the use of a fully scalable modulated window solves the signal-cutting problem. The window is shifted along the signal and for every position the spectrum is calculated. Then this process is repeated many times with a slightly shorter (or longer) window for every new cycle. In the end the result is a collection of time-frequency representations of the signal, all with different resolutions. Because of this collection of representations, we can speak of a multiresolution analysis. In the case of wavelets, we normally do not speak about time-frequency representations but about time-scale representations.

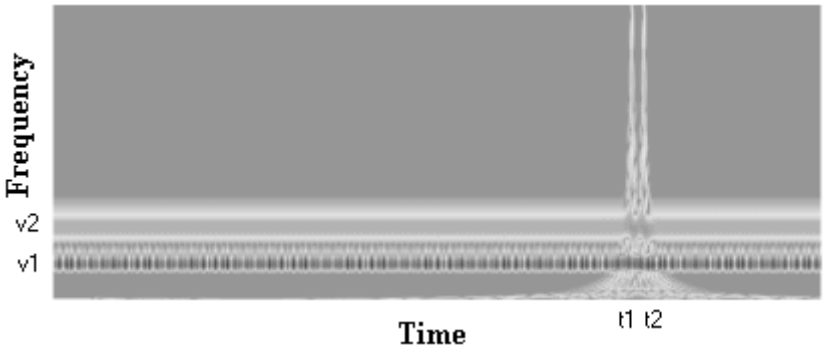


Figure 3. Continuous Wavelet Transform of Signal Shown in Figure 1

2 Discrete Wavelet Transform

The discrete wavelet transform (DWT) was developed to apply the wavelet transform to the digital world. Filter banks are used to approximate the behavior of the continuous wavelet transform. The signal is decomposed with a high-pass filter and a low-pass filter. The coefficients of these filters are computed using mathematical analysis and made available to you. See Appendix B for more information about these computations.

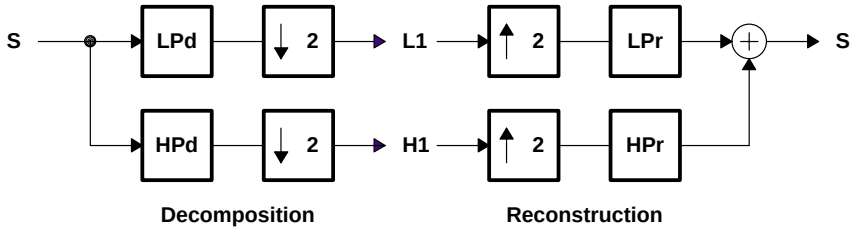


Figure 4. Discrete Wavelet Transform

Where

LPd: Low Pass Decomposition Filter

HPd: High Pass Decomposition Filter

LPr: Low Pass Reconstruction Filter

HPr: High Pass Reconstruction Filter

The wavelet literature presents the filter coefficients to you in tables. An example is the Daubechies filters for wavelets. These filters depend on a parameter *p* called the vanishing moment.

Table 1. Daubechies (p =2,3) Wavelet Coefficients

Vanishing Moment	n	$\_h_p[n]$
p = 2	0	0.48296291311445341
	1	0.8365163037378079
	2	0.2241438680420134
	3	–0.1294095225512604
p = 3	0	0.332670552950
	1	0.80691509311
	2	0.459877502118
	3	–0.135011020010
	4	–0.085441273882
	5	0.03522629291882

The $h_p[n]$ coefficients are used as the low-pass reconstruction filter (LPr).

The coefficients for the filters HPd, LPd and HPr are computed from the h[n] coefficients as follows:

- High-pass decomposition filter (HPd) coefficients

  $\overline{g}[n] = (-1)^n h[L-n]$ (L: length of the filter)

- Low-pass reconstruction filter (LPr) coefficients

  $\overline{h}[n] = h[L-n]$ (L: length of the filter)

- High-pass reconstruction filter (HPr) coefficients

  $\overline{g}[n] = g[L-n]$ (L: length of the filter)

The Daubechies filters for Wavelets are provided in the C55x IMGLIB for $2 \leq p \leq 10$.

Since there are several sets of filters, we may ask ourselves what are the advantages and disadvantages to using one set or another.

First we need to understand that we will have perfect reconstruction no matter what the filter length is. However, longer filters provide smoother, smaller intermediate results. Thus, if intermediate processing is required, we are less likely to lose information due to necessary threshold or saturation. However, longer filters obviously involve more processing.

2.1 Wavelets and Perfect Reconstruction Filter Banks

Filter banks decompose the signal into high- and low-frequency components. The low-frequency component usually contains most of the frequency of the signal. This is called the approximation. The high-frequency component contains the *details* of the signal.

Wavelet decomposition can be implemented using a two-channel filter bank. Two-channel filter banks are discussed in this section briefly. The main idea is that perfect reconstruction filter banks implement series expansions of discrete-time signals.

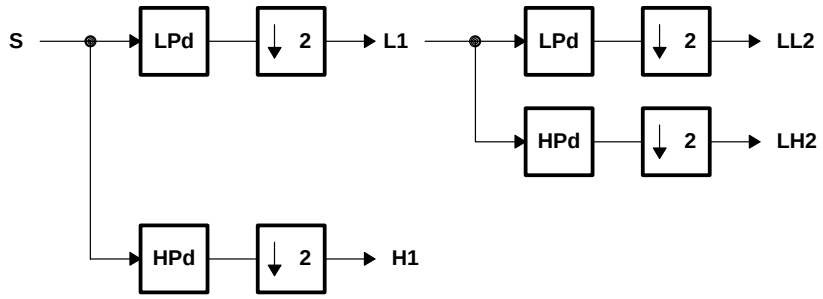


Figure 5. A Two-Level Wavelet Decomposition

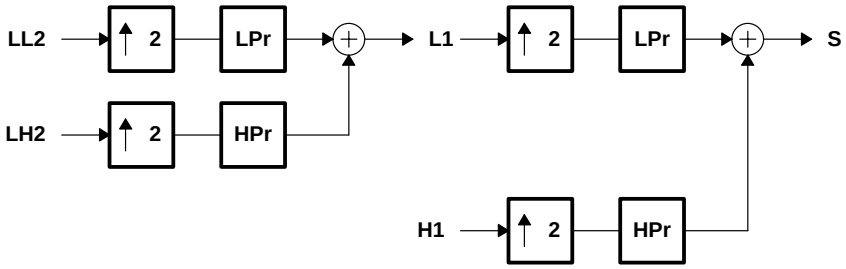


Figure 6. A Two-Level Wavelet Reconstruction

The input and the reconstruction are identical; this is called perfect reconstruction. Two popular decomposition structures are *pyramid* and *wavelet packet*. The first one decomposes only the approximation (low-frequency component) part while the second one decomposes both the approximation and the detail (high-frequency component).

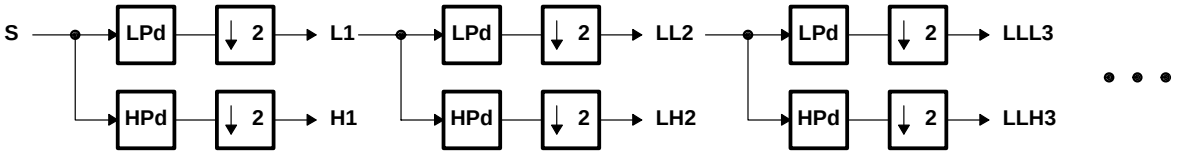


Figure 7. Pyramid Packet

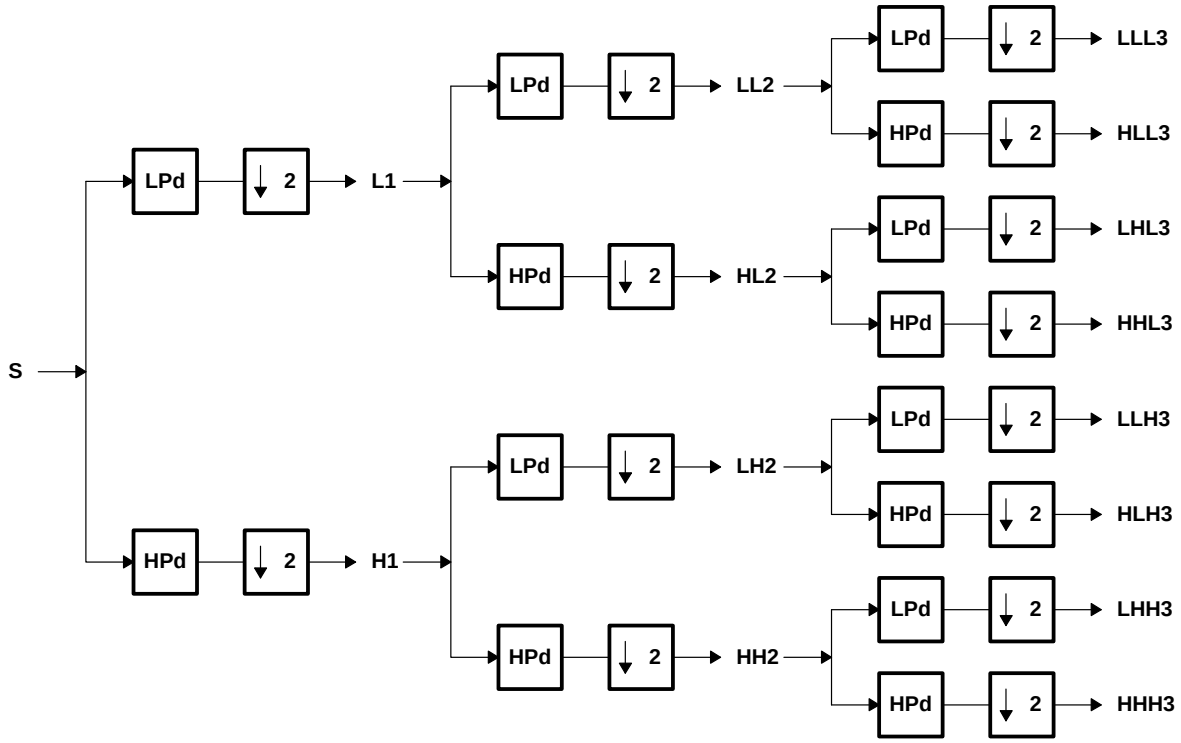


Figure 8. Wavelet Packet Decomposition

The C55x IMGLIB provides the following functions for one dimension pyramid and packet decomposition and reconstruction. Complete information about these functions can be found in the C55x IMGLIB.

- 1-D discrete wavelet transform

  void IMG_wave_decom_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);

- 1-D inverse discrete wavelet transform

  void IMG_wave_recon_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);

- 1-D discrete wavelet package transform

  void IMG_wavep_decom_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);

- 1-D inverse discrete wavelet package transform

  void IMG_wavep_recon_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);

2.2 Wavelet Filter Bank Implementation

When using the filter bank algorithm to do wavelet transform, the decomposed signal's length is:

$$\left(Length_{signal} + Length_{filter} - 1\right)/2$$

After reconstruction, the reconstructed signal length is:

$$Length_{signal} + 2 * Length_{filter} - 2$$

So we need to cut off the head and tail, or use a circular buffer to make the reconstructed signal exactly the same as the original signal. For thsi The following items are required

A signal with ten samples:

$$\{S_1, S_2, S_3, \cdots S_9, S_{10}\}$$

A pair of filters with six elements each:

$$\{\bar{g}_0, \bar{g}_1, \cdots \bar{g}_5\} \text{ and } \{\bar{h}_0, \bar{h}_1, \cdots \bar{h}_5\}$$

The reconstruction filters:

$$\{g_0, g_1, \cdots g_5\} \text{ and } \{h_0, h_1, \cdots h_5\}$$

The decomposed signal should be:

$$\{\bar{a}_1, \bar{a}_2, \cdots \bar{a}_5\} \text{ and } \{\bar{d}_1, \bar{d}_2, \cdots \bar{d}_5\}$$

To do the decomposition, we do the convolution:

$$a_1 = S_1\bar{g}_5 + S_2\bar{g}_4 + S_3\bar{g}_3 + S_4\bar{g}_2 + S_5\bar{g}_1 + S_6\bar{g}_0$$

$$a_2 = S_3\bar{g}_5 + S_4\bar{g}_4 + S_5\bar{g}_3 + S_6\bar{g}_2 + S_7\bar{g}_1 + S_8\bar{g}_0$$

Down-sampling is required after the convolution. This is avoided by picking every other output as our output. So the convolution is shifted by 2 instead by 1, and the down-sampling is unnecessary.

Figure 9 shows how the algorithm can be written in matrix format:

$$\begin{bmatrix}
\bar{g}_5 & \bar{g}_4 & \bar{g}_3 & \bar{g}_2 & \bar{g}_1 & \bar{g}_0 & & & & & & \\
\bar{h}_5 & \bar{h}_4 & \bar{h}_3 & \bar{h}_2 & \bar{h}_1 & \bar{h}_0 & & & & & & \\
 & & \bar{g}_5 & \bar{g}_4 & \bar{g}_3 & \bar{g}_2 & \bar{g}_1 & \bar{g}_0 & & & & \\
 & & \bar{h}_5 & \bar{h}_4 & \bar{h}_3 & \bar{h}_2 & \bar{h}_1 & \bar{h}_0 & & & & \\
 & & & & \bar{g}_5 & \bar{g}_4 & \bar{g}_3 & \bar{g}_2 & \bar{g}_1 & \bar{g}_0 & & \\
 & & & & \bar{h}_5 & \bar{h}_4 & \bar{h}_3 & \bar{h}_2 & \bar{h}_1 & \bar{h}_0 & & \\
\bar{g}_1 & \bar{g}_0 & & & & & \bar{g}_5 & \bar{g}_4 & \bar{g}_3 & \bar{g}_2 & & \\
\bar{h}_1 & \bar{h}_0 & & & & & \bar{h}_5 & \bar{h}_4 & \bar{h}_3 & \bar{h}_2 & & \\
\bar{g}_3 & \bar{g}_2 & \bar{g}_1 & \bar{g}_0 & & & & & \bar{g}_5 & \bar{g}_4 & & \\
\bar{h}_3 & \bar{h}_2 & \bar{h}_1 & \bar{h}_0 & & & & & \bar{h}_5 & \bar{h}_4 & &
\end{bmatrix}
\begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \end{bmatrix}
=
\begin{bmatrix} a_1 \\ d_1 \\ a_2 \\ d_2 \\ a_3 \\ d_3 \\ a_4 \\ d_4 \\ a_5 \\ d_5 \end{bmatrix}$$

Figure 9. Filter Bank Algorithm in Matrix Format

The reconstruction of this example is shown in Figure 10.

$$\begin{bmatrix} h_5 & & & & & h_0 & h_1 & h_2 & h_3 & h_4 \\ h_4 & h_5 & & & & h_0 & h_1 & h_2 & h_3 & \\ h_3 & h_4 & h_5 & & & h_0 & h_1 & h_2 & & \\ h_2 & h_3 & h_4 & h_5 & & h_0 & h_1 & & & \\ h_1 & h_2 & h_3 & h_4 & h_5 & h_0 & & & & \\ h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & & & & \\ & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & & & \\ & & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & & \\ & & & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & \\ & & & & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 \end{bmatrix} \begin{bmatrix} a_1 \\ 0 \\ a_2 \\ 0 \\ a_3 \\ 0 \\ a_4 \\ 0 \\ a_5 \\ 0 \end{bmatrix} + \begin{bmatrix} g_5 & & & & & g_0 & g_1 & g_2 & g_3 & g_4 \\ g_4 & g_5 & & & & g_0 & g_1 & g_2 & g_3 & \\ g_3 & g_4 & g_5 & & & g_0 & g_1 & g_2 & & \\ g_2 & g_3 & g_4 & g_5 & & g_0 & g_1 & & & \\ g_1 & g_2 & g_3 & g_4 & g_5 & g_0 & & & & \\ g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & & & & \\ & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & & & \\ & & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & & \\ & & & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & \\ & & & & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 \end{bmatrix} \begin{bmatrix} d_1 \\ 0 \\ d_2 \\ 0 \\ d_3 \\ 0 \\ d_4 \\ 0 \\ d_5 \\ 0 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \end{bmatrix}$$

Figure 10. Reconstruction From Figure 9

Figure 11 shows the reconstruction written it in an elegant format:

$$\begin{bmatrix} h_5 & g_5 & & & h_1 & g_1 & h_3 & g_3 \\ h_4 & g_4 & & & h_0 & g_0 & h_2 & g_2 \\ h_3 & g_3 & h_5 & g_5 & & & h_1 & g_1 \\ h_2 & g_2 & h_4 & g_4 & & & h_0 & g_0 \\ h_1 & g_1 & h_3 & g_3 & h_5 & g_5 & & \\ h_0 & g_0 & h_2 & g_2 & h_4 & g_4 & & \\ & & h_1 & g_1 & h_3 & g_3 & h_5 & g_5 \\ & & h_0 & g_0 & h_2 & g_2 & h_4 & g_4 \\ & & & & h_1 & g_1 & h_3 & g_3 & h_5 & g_5 \\ & & & & h_0 & g_0 & h_2 & g_2 & h_4 & g_4 \end{bmatrix} \begin{bmatrix} a_1 \\ d_1 \\ a_2 \\ d_2 \\ a_3 \\ d_3 \\ a_4 \\ d_4 \\ a_5 \\ d_5 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \end{bmatrix}$$

Figure 11. Altered Reconstruction From Figure 9

These calculations can be easily implement on a DSP with circular buffer management.

The reconstruction filters from the matrix in Figure 10 changed from:

$\{h_0\, h_1\, h_2\, h_3\, h_4\, h_5\}$ and $\{g_0\, g_1\, g_2\, g_3\, g_4\, g_5\}$

to

$\{h_1\, g_1\, h_3\, g_3\, h_5\, h_5\}$ and $\{h_0\, g_0\, h_2\, g_2\, h_4\, g_4\}$

This is done for the convenience of implementing wavelets on TMS320C55x.

3 Wavelets Image Processing

Wavelets have found a large variety of applications in the image processing field. The JPEG 2000 standard uses wavelets for image compression. Other image processing applications such as noise reduction, edge detection, and finger print analysis have also been investigated in the literature.

3.1 Wavelet Decomposition of Images

In wavelet decomposing of an image, the decomposition is done row by row and then column by column. For instance, here is the procedure for an N x M image. You filter each row and then down-sample to obtain two N x (M/2) images. Then filter each column and subsample the filter output to obtain four (N/2) x (M/2) images.

Of the four subimages obtained as seen in Figure 12, the one obtained by low-pass filtering the rows and columns is referred to as the LL image. The one obtained by low-pass filtering the rows and high-pass filtering the columns is referred to as the LH images. The one obtained by high-pass filtering the rows and low-pass filtering the columns is called the HL image. The subimage obtained by high-pass filtering the rows and columns is referred to as the HH image. Each of the subimages obtained in this fashion can then be filtered and subsampled to obtain four more subimages. This process can be continued until the desired subband structure is obtained.

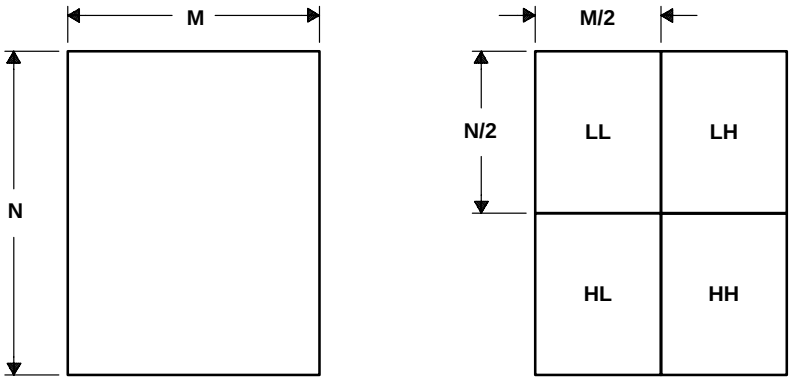


Figure 12. Original Image One-Level 2-D Decomposition

Three of the most popular ways to decompose an image are: pyramid, spacl, and wavelet packet, as shown in Figure 13.

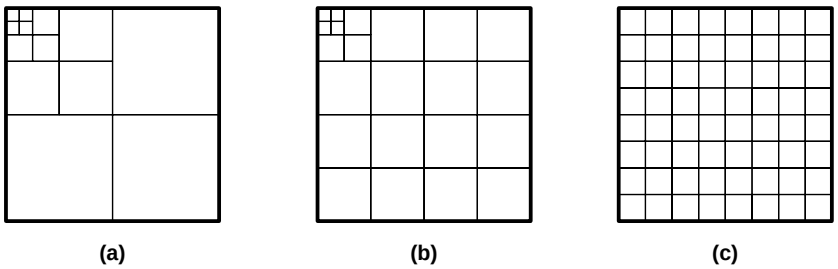


Figure 13. Three Popular Wavelet Decomposition Structures on Image: (a) Pyramid, (b) Spacl, (c) Wavelet Packet

- In the structure of pyramid decomposition, only the LL subimage is decomposed after each decomposition into four more subimages.
- In the structure of wavelet packet decomposition, each subimage(LL, LH, HL, HH) is decomposed after each decomposition.
- In the structure of spacl, after the first level of decomposition, each subimage is decomposed into smaller subimages, and then only the LL subimage is decomposed.

Figure14 shows a three-level decomposition image of pyramid structure.

(a) (b)

Figure 14. (a) Original Image (b) Three-Level Pyramid Structure Decomposition

In the part I development stage, the JPEG 2000 standard supports the pyramid decomposition structure. In the future all three structures will be supported.

For two dimensions, the C55x IMGLIB provides functions for pyramid and packet decomposition and reconstruction. Complete information about these functions can be found in the C55x IMGLIB.

- 2-D discrete wavelet transform

  void IMG_wave_decom_two_dim(short **image, short * wksp, int width, int height,
  int *wavename, int level);

- 2-D inverse discrete wavelet transform

  void IMG_wave_recon_two_dim(short **image, short * wksp, int width, int height,
  int *wavename, int level);

- 2-D discrete wavelet package transform

  void IMG_wavep_decom_two_dim(short **image, short * wksp, int width, int height,
  int *wavename, int level);

- 2-D inverse discrete wavelet package transform

  void IMG_wavep_recon_two_dim(short **image, short * wksp, int width, int height,
  int *wavename, int level);

4 Wavelets Applications

TI provides several one dimension and two dimension wavelets applications, which illustrate how to use the wavelets functions provided in the C55x IMGLIB.

4.1 One Dimension Wavelet Applications

The 1D_Demo.c file presents applications of the one-dimension wavelet. A 128-point sine wave is used as input for all these applications as shown in Figure 15:

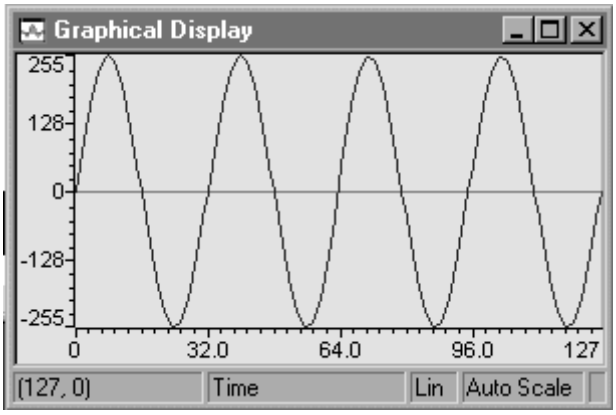


Figure 15. Input Signal

4.1.1 Discontinuity Detection Example

The first application shows how the wavelet transform can be used to detect a discontinuity. We remove the point 64 from the sine wave to produce a discontinuity.

The discontinuity is shown in Figure 16. This code is used to produce the discontinuity:

```
// Initialization
            //================================================
            freq1 = 2;
            rate  = 64;
            for( i = 0; i < (LENGTH>>1); i++ )
                  signal[i] = 255 * sin(2.0*3.1415*freq1*i/rate);
            for( i = (LENGTH>>1); i < LENGTH; i++ )
                  signal[i] = 255 *sin(2.0*3.1415*freq1*(i+1)/rate);
```

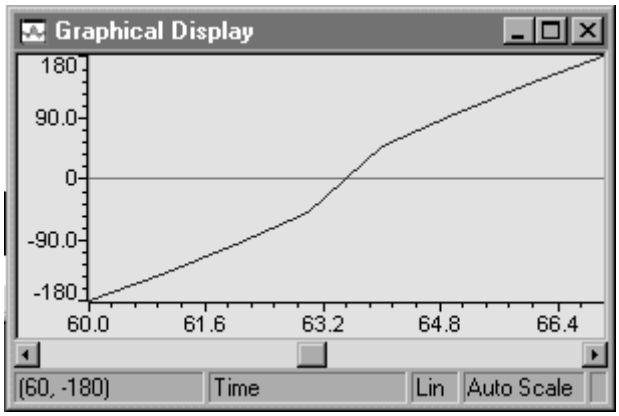


Figure 16. Zoom of the Discontinuity of the Sinusoidal Input Signal

We notice that the discontinuity is at point 64.

We are performing a one-level decomposition:

```
// Using Wavelet to find a discontinuity
            //================================================
            IMG_wave_decom_one_dim( signal, temp_wksp, db4, LENGTH, 1 );
            //------------------------------------------------
```

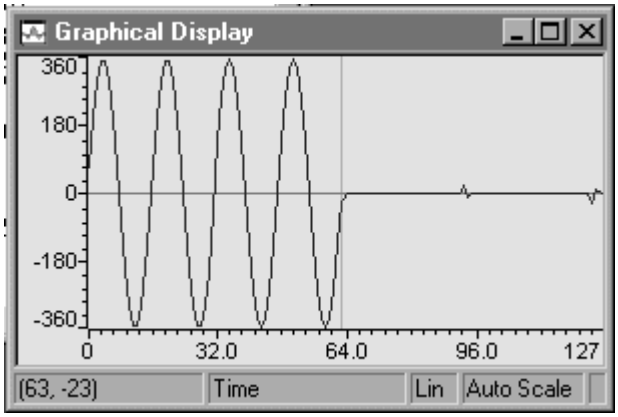


Figure 17. One-Level Decomposition of the Discontinue Signal

We notice that the high part of the signal (the part that contains the details) shows the discontinuity of the original input signal. The discontinuity appears at point 32 of the high part which is point 96 = 64 + 32 on the graph in Figure 17, which represents the high and low parts.

4.1.2 Noise Removal Example

The second application shows how the wavelets can be used to reduce the noise. The input signal is the sine wave plus high frequency noise as seen in Figure 18:

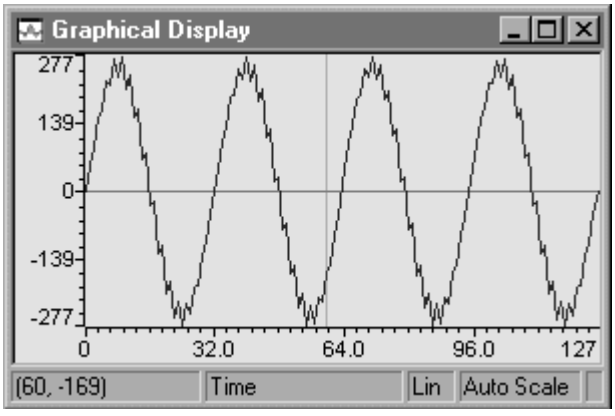


Figure 18. Noisy Signal

We are performing a one-level decomposition:

```
// Using Wavelet for Denosing
            //================================================
            for( i = 0; i < LENGTH; i++ )
                  signal[i] = backup[i] + noise[i];
            IMG_wave_decom_one_dim( signal, temp_wksp, db4, LENGTH, 1);
            for( i = (LENGTH>>1); i < LENGTH; i++ )
            {
                  if(signal[i] > 3 ) signal[i] = 3;
                  if(signal[i] < –3 ) signal[i] = –3;
            }
            IMG_wave_recon_one_dim( signal, temp_wksp, db4, LENGTH, 1 );
```

We notice that the low part has the shape of the signal (Figure 19) whereas the high part has the spikes from the noise (Figure 20). A threshold [–3, +3] is applied to the high part and the signal is reconstructed as in Figure 21.

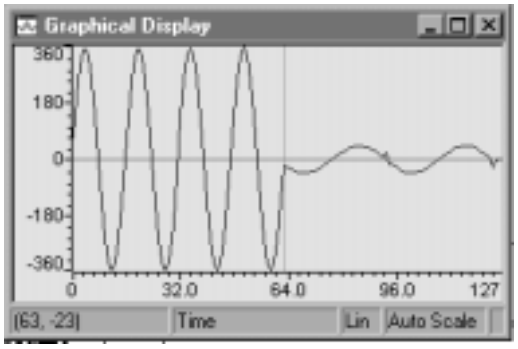


Figure 19. One-Level Decompostion of the Noisy Signal

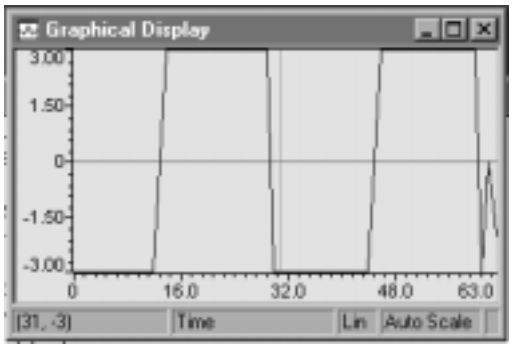


Figure 20. Threshold Applied to the High-Pass Channel

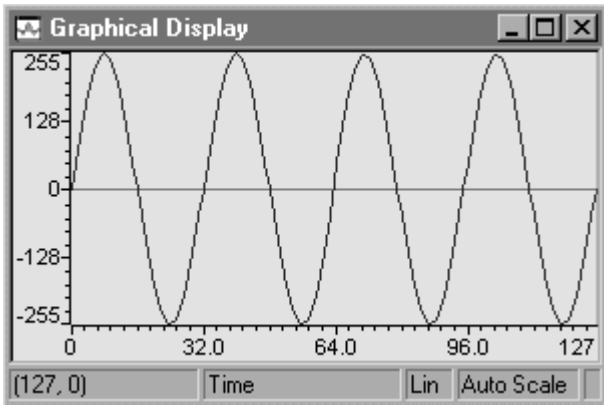


Figure 21. Reconstructed Signal

4.1.3 1-D Perfect Decomposition and Reconstruction Example

The third application shows a three-level pyramid decomposition and reconstruction of the input signal:

```
// Perfect Reconstruction of Pyramid, Level 3
            //================================================
            for( i = 0; i < LENGTH; i++ )
                  signal[i] = backup[i];
            IMG_wave_decom_one_dim( signal, temp_wksp, db4, LENGTH, 3 );
            IMG_wave_recon_one_dim( signal, temp_wksp, db4, LENGTH, 3 );
                  for( i = 0; i < LENGTH; i++ )
                  noise[i] = signal[i] – backup[i];
            //------------------------------------------------
```

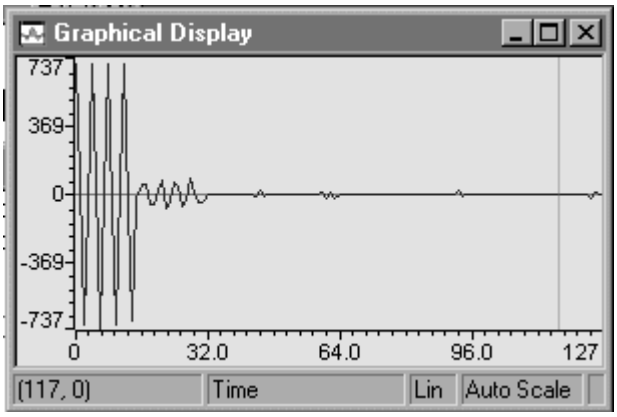


Figure 22. Level 3 Pyramid Decompostion of the Original Sine Wave Signal

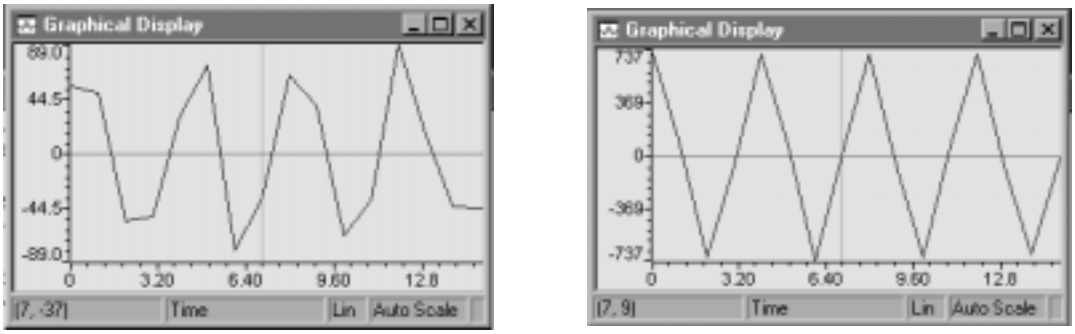


Figure 23. LLL3 (left) and HLL3 (right) Components of Three-Level Decomposition

The error signal shown in Figure 24 represents the difference between the original signal and the reconstructed signal. This error signal is not zero because of the dynamic range of the 16-bit fixed-point data.

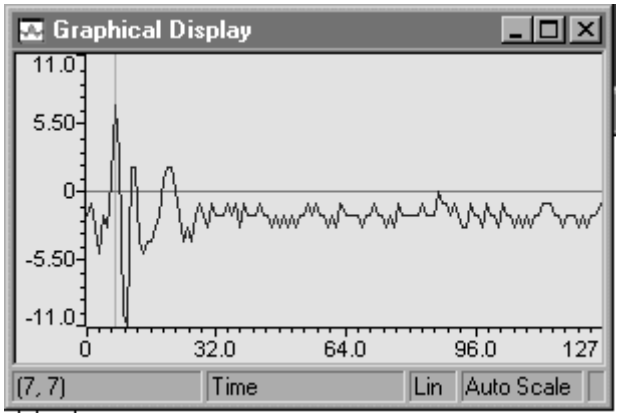


Figure 24. Reconstructed Error

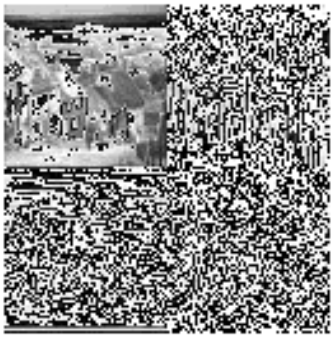
4.2 Two Dimension Wavelet Applications

The 2D_Demo.c file, provided in the C55x IMGLIB, presents applications of the two-dimension wavelet. A 128x128 image is used as input for all these applications. In the first application we are using the picture in Figure 25:

Figure 25. Image Used in the 2D Wavelet Application

4.2.1 2-D Perfect Decomposition and Reconstruction Example

In this application, the image is one-level decomposed and reconstructed. You notice no difference between the original picture and the reconstructed picture as shown in Figure 26.



One-Level Decomposed Image

Reconstructed Image

Figure 26. One-Level Decomposed and Reconstructed Image

4.2.2 Edge Detection Example

In the second application, a 2-D edge detection is performed for the picture in Figure 27:

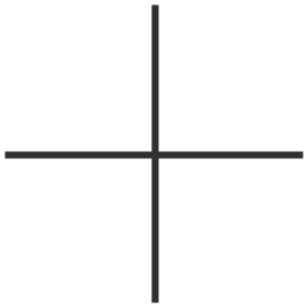


Figure 27. Picture Used in Edge Detection Application

The result of the one-level decomposition shows that the edges are detected as in Figure 28:

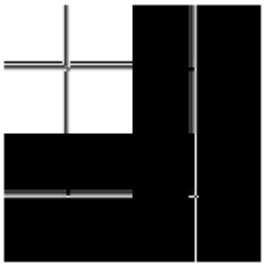


Figure 28. 2-D One-Level Decomposition Detects the Edges

The HH part of the picture has a vertical line. This happens because a row by row processing was performed first and then a column by column.

5 References

1. S. Mallat. A Wavelet tour of Signal Processing. Academic Press, 2000.
2. J. C. Goswami and Andrew K. Chan. Fundamentals of Wavelets: Theory, algorithm, and applications. John Wiley & Sons, 1999
3. Martin Vetterli and Jelena Kovacevic. Wavelets and Subband Coding. Prentice Hall, 1995
4. Clemens Valen. A really friendly guide to wavelets. http://perso.wanadoo.fr/polyvalens/clemens/wavelets/wavelets.html
5. Robi Polikar. The engineer's ultimate guide to wavelet analysis. http://www.public.iastate.edu/~rpolikar/WAVELETS/WTtutorial.html
6. Khalid Sayood. Introduction to Data Compression. Morgan Kaufmann, 2000

Appendix A Fourier Transform

A.1 Fourier Transforms

The Fourier transform's utility lies in its ability to analyze a signal in the time domain for its frequency content. The transform works by first translating a function in the time domain into a function in the frequency domain. The signal can then be analyzed for its frequency content because the Fourier coefficients of the transformed function represent the contribution of each sine and cosine function at each frequency. An inverse Fourier transform translating the data from the frequency domain into the time domain.

If f has finite energy, the theory of Fourier integrals proves that the amplitude $\hat{f}(w)$ of each e^{jwt} is the Fourier transform of f:

$$\hat{f}(\omega) = \int_{-\infty}^{+\infty} f(t)e^{-jwt}\,dt \quad (1)$$

The inverse *Fourier Transform* is:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \hat{f}(\omega)e^{jwt}\,d\omega \quad (2)$$

A.2 Window Fourier Transforms

In 1946, Gabor introduced window Fourier atoms to measure the *frequency variation* of sounds. A real and symmetric window *g(t)* is translated by μ and modulated by the frequency ξ:

$$g_{\xi\mu}(t) = e^{i\xi t}\,g(t-\mu) \quad (3)$$

It is normalized ||*g*|| = 1 so that ||$g_{\mu\xi}$|| =1 for any (μ,ξ) $\in \Re^2$. The resulting windowed Fourier transform of $f \in L^2(\Re)$ is

$$Sf(\mu,\xi) = < f, g_{\mu\xi} > = \int_{-\infty}^{+\infty} f(t)g(t-\mu)e^{-i\xi t}\,dt \quad (4)$$

This transform is also called the *STFT* (*Short-Time-Fourier-Transform*) or *Gabor Transform* because the multiplication by g(t–μ) localizes the Fourier integral in the neighborhood of t = μ

The *Heisenberg Uncertainty Principle* shows that the temporal variance and the frequency variance of $f \in L^2(\Re)$ satisfy

$$\sigma_g^2\,\sigma_t^2 \geq \frac{1}{4} \quad (5)$$

Thus, to have finer resolution in time, that is, reduce σ_t^2, you end up with an increase in σ_5^2, or a lower resolution in the frequency domain.

Appendix B Wavelet Transform

B.1 Continuous Wavelet Transforms

To analyze signal structures of very different sizes, it is necessary to use the time-frequency atom method with different time supports. The wavelet transform decomposes signals over dilated and translated wavelets. A wavelet is a function $\psi \in L^2(\Re)$ with a zero average:

$$\int_{-\infty}^{+\infty} \psi(t)dt = 0 \tag{6}$$

It is normalized $||\psi|| = 1$, and centered in the neighborhood of $t = 0$. A family of time-frequency atoms is obtained by scaling

Dilating Ψ by S and translating it by μ:

$$\psi_{\mu,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\mu}{s}\right) \tag{7}$$

These atoms remain normalized: $||\psi_{\mu\xi}|| = 1$. The wavelet transform of $f \in L^2(\Re)$ at time μ and scale S is determined with this equation:

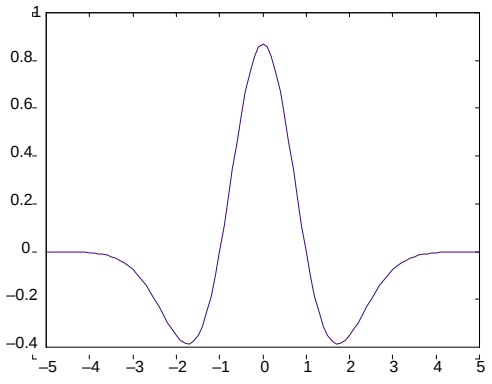
$$WF(\mu, s) = \prec f, \psi_{\mu,s} \succeq \int f(t)\frac{1}{\sqrt{s}}\psi \cdot\left(\frac{t-\mu}{s}\right)dt \tag{8}$$

Wavelets equal to the second derivative of a Gaussian function are called Mexican hats. They were first used in computer vision to detect multiscale edges. The normalized Mexican hat wavelet is determined with this equation:

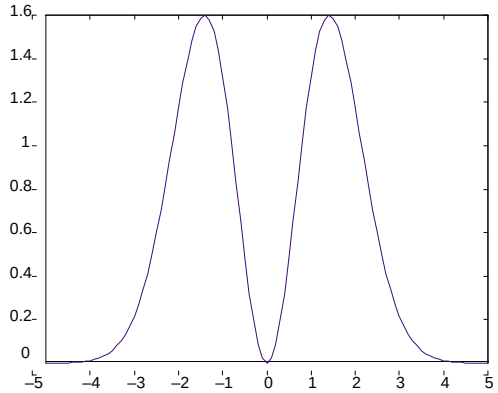
$$\psi(t) = \frac{2}{\pi^{1/4}\sqrt{3}\sigma}\left(\frac{t^2}{\sigma^2} - 1\right)\exp\left(-\frac{t^2}{2\sigma^2}\right) \tag{9}$$

For $\sigma = 1$, Figure 30 plots $-\Psi$ and its Fourier transform

$$\hat{\psi}(\omega) = \frac{-\sqrt{8}\sigma^{5/2}\pi^{1/4}}{\sqrt{3}}\omega^2\exp\left(\frac{-\sigma^2\omega^2}{2}\right) \tag{10}$$



$-\psi(t)$



$-\hat{\psi}(\omega)$

Figure B–1. Mexican Hat Wavelet and Its Fourier Transform

If a real function $\psi \in L^2(\Re)$ satisfies the admissibility condition as determined by this equation:

$$C_\psi = \int_0^{+\infty} \frac{\left|\hat{\psi}(\omega)\right|^2}{\omega} d\omega < \infty \quad (11)$$

Then any $f \in L^2(\Re)$ satisfies the following equations:

$$f(t) = \frac{1}{C_\psi} \int_0^{+\infty} \int_{-\infty}^{+\infty} Wf(\mu, s) \frac{1}{\sqrt{s}} \psi\left(\frac{t-\mu}{s}\right) d\mu \frac{ds}{s^2} \quad (12)$$

$$\int_{-\infty}^{\infty} \left|f(t)\right|^2 dt = \frac{1}{C_\psi} \int_0^{+\infty} \int_{-\infty}^{+\infty} |Wf(\mu, s)|^2 d\mu \frac{ds}{s^2} \quad (13)$$

The admissibility condition shows that:

$$\left|\hat{\psi}(\omega)\right|^2 = 0 \quad (14)$$

which means the wavelets must have a band-pass like spectrum. This is a very important observation, which is used later on to build an efficient wavelet transform.

B.2 Discrete Wavelets

The CWT is calculated by continuously shifting a continuously scalable function over a signal and calculating the correlation between the two. It will be clear that these scaled functions will be nowhere near an orthogonal basis and the obtained wavelet coefficients will therefore be highly redundant. For most practical applications we would like to remove this redundancy.

To overcome this problem, *discrete wavelets* have been introduced. Discrete wavelets are not continuously scalable and translatable but can only be scaled and translated in discrete steps. This is achieved by modifying the wavelet representation to create this equation:

$$\psi_{j,k}(t) = \frac{1}{\sqrt{S_0^j}} \psi\left[\frac{t - k\tau_0 S_0^j}{S_0^j}\right] \quad (15)$$

Although it is called a discrete wavelet, it is normally a continuous function. The *j* and *k* are integers and $s_0 > 1$ is a fixed dilation step. The translation factor τ_0 depends on the dilation step. The effect of discretizing the wavelet is that the time-scale space is now sampled at discrete intervals. So that the sampling of the frequency axis corresponds to *dyadic sampling*, $s_0 = 2$ is usually chosen. This is a very natural choice for computers, the human ear and music, for instance. For the translation factor, $\tau_0 = 1$ is usually chosen so that we also have dyadic sampling of the time axis.

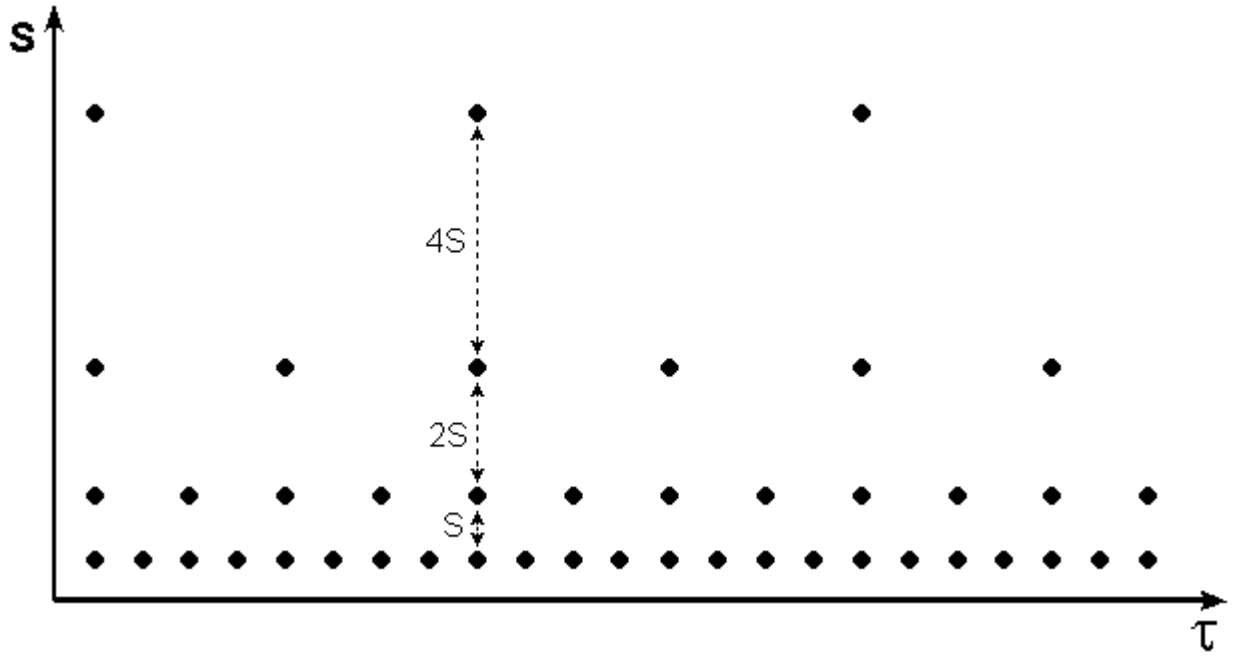


Figure B–2. Localization of the Discrete Wavelets in the Time-Scale Space on a Dyadic Grid

When discrete wavelets are used to transform a continuous signal, the result is a series of wavelet coefficients, and it is referred to as the *wavelet series decomposition*. An important issue in such a decomposition scheme is of course the question of reconstruction. It is all very well to sample the time-scale joint representation on a dyadic grid, but if it will not be possible to reconstruct the signal, it will not be of great use. As it turns out, it is indeed possible to reconstruct a signal from its wavelet series decomposition.

B.2.1 Multiresolution Analysis and Scaling Function

After solving the redundant problem, a way needs to be found to design wavelets that will be easy to use. Here are some concepts about multiresolution analysis and scaling function.

Let $L^2(\Re)$ denote the vector space of measurable, square-integrable one-dimensional functions $f(x)$, then a multiresolution analysis of $L^2(\Re)$ consists of a ladder of spaces,

$$\cdots \subset V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots$$

with $\lim V_j = \bigcup_{j=-\infty}^{j=+\infty} V_j = L^2(\Re)$, $\lim_{j=-\infty} V_j = \bigcap_{j\in Z} V_j = \{0\}$,

which satisfy the following two conditions:

1. $f(t) \in V_j \Leftrightarrow f\left(\frac{t}{2}\right) \in V_{j+1}$

2. There exists $\phi \in V_0$ such that $\{\phi(t-n)\}_{n\in Z}$ constitutes an orthogonal basis of V_0.

Since ϕ generates a multiresolution analysis, it is called a *scaling function*. The space V_j can be considered as different approximation spaces: for a given f, the successive projections $ProjV_j$ describe an approximations of f with resolution 2^j.

If we define $\phi_{j,n}(t) = \frac{1}{\sqrt{2^j}}\phi\left(\frac{t-n}{2^j}\right)$, the family $\{\phi_{j,n}\}_{n\in Z}$ is an orthonormal basis of V_j for all $j \in Z$.

The multiresolution causality property imposes that $V_j \subset V_{j-1}$. In particular, $2^{-1/2}\,\varphi(t/2) \in V_j < V_0$. Since $\{\phi(t-n)\}_{n\in Z}$ is an orthonormal basis in V_0, we can decompose this equation:

$$\frac{1}{\sqrt{2}}\phi\left(\frac{t}{2}\right) = \sum_{n=-\infty}^{+\infty} h[n]\phi(t-n) \tag{16}$$

With this equation:

$$h[n] = \left(2^{-1/2}\phi(t/2), \phi(t-n)\right) \tag{17}$$

This scaling equation relates a dilation of ϕ by 2 to its integer translations. The sequence *h[n]* will be interpreted as a discrete filter.

The Fourier transform of both sides of (16) yields

$$\hat{\phi}(2\omega) = 2^{-1/2}\,\hat{h}(\omega)\hat{\phi}(\omega) \tag{18}$$

By substitution, we obtain

$$\hat{\phi}(\omega) = \left[\prod_{p=1}^{P}\frac{\hat{h}\left(2^{-p/2}\omega\right)}{\sqrt{2}}\right]\hat{\phi}\left(2^{-P/2}\omega\right) \tag{19}$$

If $\hat{\phi}(\omega)$ is continuous at $\omega = 0$ then

$$\hat{\phi}(\omega) = \prod_{p=1}^{\infty}\frac{\hat{h}\left(2^{-p}\omega\right)}{\sqrt{2}}\hat{\phi}(0) \tag{20}$$

B.2.2 Orthogonal Wavelets Bases

Here are some conditions to construct orthogonal wavelet bases. For biorthogonal, please read the texts in the reference list.

Condition 1

Let $\phi \in L^2(\Re)$ be an integrable scaling function. The Fourier series of $h[n] = \langle 2^{-1/2}\,\phi(t/2), \phi(t-n)\rangle$ satisfies the following equations:

$$\text{for } \forall\omega \in \Re, \left|\hat{h}(\omega)\right|^2 + \left|\hat{h}(\omega+\pi)\right|^2 = 2 \tag{21}$$

$$\hat{h}(0) = \sqrt{2} \text{ or } \sum_n h(n) = \sqrt{2} \tag{22}$$

Orthonormal wavelets carry the details necessary to increase the resolution of a signal approximation. The approximations of f at the scales 2^j and 2^{j-1} are respectively equal to their orthogonal projections on V_j and V_{j-1}. Let W_j be the orthogonal complement of V_j in V_{j-1}:

$$V_{j-1} = V_j \oplus W_j \tag{23}$$

The orthogonal projection of f on V_{j-1} can be decomposed as the sum of orthogonal projections on V_j and W_j:

Orthonormal wavelets carry the details necessary to increase the resolution of a signal approximation. The approximations of f at the scales 2^j and 2^{j-1} are respectively equal to their orthogonal projections on V_j and V_{j-1}. Let W_j be the orthogonal complement of V_j in V_{j-1}:

$$\text{Pr oj}V_{j-1}f = \text{Pr oj}V_j f + \text{Pr oj}W_j f \quad (24)$$

The complement $PRojW_j f$ provides the details of the f that appear at the scale 2^{j-1} but which disappear at the coarser scale 2^j. The following condition proves that one can construct an orthonormal basis of W_j by scaling and translating a wavelet ψ.

Condition 2

Let ϕ be a scaling function and the corresponding conjugate mirror filter. Let ψ be the function whose Fourier transform is

$$\hat{\psi}(\omega) = 2^{-1/2}\hat{g}(\omega/2)\hat{\phi}(\omega/2) \quad (25)$$

with

$$\hat{g}(\omega) = e^{-j\omega}\hat{h}^*(\omega + \pi) \quad (26)$$

Let us denote

$$\psi_{j,n}(t) = \frac{1}{\sqrt{2^j}}\psi\left(2^{-j}t - n\right) \quad (27)$$

For any scale 2^j, $\{\Psi_{j,n}\}\ n \in z$ is an orthonormal basis of W_j. For all scales, $\{\Psi_{j,n}\}\ n \in Z$ is an orthonormal basis of $L^2(\Re)$.

The family $\{\psi_{j,n}\}_{n\in Z}$ is an orthonormal basis of W_j if and only if these equations are true:

$$\text{for}\,\omega \in \Re, |\hat{g}(\omega)|^2 + |\hat{g}(\omega + \pi|^2 = 2 \quad (28)$$

$$\hat{g}(\omega)\hat{h}^*(\omega) + \hat{g}(\omega + \pi)\hat{h}^*(\omega + \pi) = 0 \quad (29)$$

Then this results:

$$g[n] = \left\langle 2^{-\frac{1}{2}} \prec \psi(t/2), \phi(t - n)\right\rangle = (-1)^{l-n}h[l - n] \quad (30)$$

$$\sum_n g[n] = 0 \quad (31)$$

B.3 Wavelet and Perfect Reconstruction Filter Bank

Decomposition coefficients in a wavelet orthogonal basis are computed with a fast algorithm that cascades discrete convolutions with h and g, and subsamples the output. We describe a fast filter bank algorithm that computes the orthogonal wavelet coefficients of a signal measured at a finite resolution. A fast wavelet transform decomposes successively each approximation $ProV_j f$ into a coarser approximation $ProbW_{j+1} f$ plus the wavelet coefficients carried by $ProjW_{j+1} f$. In the other direction, the reconstruction from wavelet coefficients recovers each $PRojV_{j+1} f$ from $ProbV_{j+1} f$ and $ProjW_{j+1} f$. Let's define $\overline{h}[n] = h[-n]$ and $\overline{g}[n] = g[-n]$.

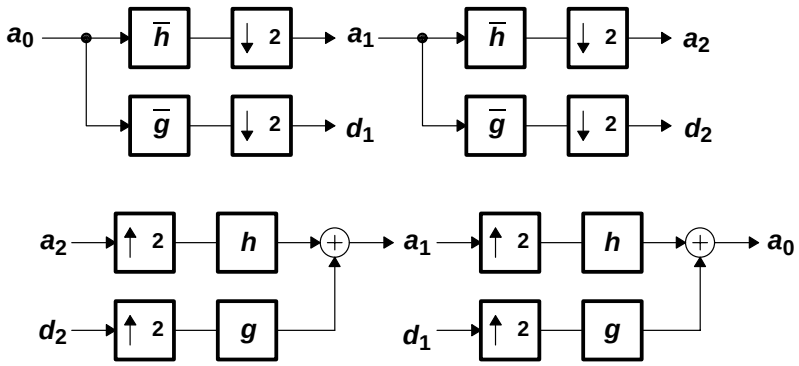


Figure B–3. A Two-Level Wavelet Decomposition and Reconstruction

Figure B–3 shows a fast wavelet transform is computed with a cascade of filters with $\overline{h}$ and $\overline{g}$ followed by a factor 2 subsampling. A fast inverse wavelet transform reconstructs progressively each a_j by inserting zeros between samplings of a_{j+1} and d_{j+1}, filtering and adding the output.

The input and the output are identical; this is called perfect reconstruction.

Appendix C Wavelet Functions API

The following functions belong to the C55x image library (see Table C–1).

Table C–1. Wavelet Functions Written in C Code

Description	Syntax
1-D discrete wavelet transform	void IMG_wave_decom_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);
1-D inverse discrete wavelet transform	void IMG_wave_recon_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);
1-D discrete wavelet package transform	void IMG_wavep_decom_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);
1-D inverse discrete wavelet package transform	void IMG_wavep_recon_one_dim(short *in_data, short *wksp, int *wavename, int length, int level);
2-D discrete wavelet transform	void IMG_wave_decom_two_dim(short **image, short * wksp, int width, int height, int *wavename, int level);
2-D inverse discrete wavelet transform	void IMG_wave_recon_two_dim(short **image, short * wksp, int width, int height, int *wavename, int level);
2-D discrete wavelet package transform	void IMG_wavep_decom_two_dim(short **image, short * wksp, int width, int height, int *wavename, int level);
2-D inverse discrete wavelet package transform	void IMG_wavep_recon_two_dim(short **image, short * wksp, int width, int height, int *wavename, int level);

NOTE: Overflow Prevention in IMGLIB

There is no overflow prevention in these functions. However, if the input is in the range of [0, 255], there should be no overflow up to at least five levels of decomposition for all wavelets filters. The library has the following five families of wavlets: bior, coif, daub, rbio and sym.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third–party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265