Engineering Education: Computer-Aided Engineering with MATLAB; Discrete Wavelet Transform as a Case Study

Abdul Rasak Zubair
Electrical/Electronic Engineering Department
University of Ibadan, Ibadan, Nigeria

Yusuf Kola Ahmed
Biomedical Engineering Department
University of Ilorin, Ilorin, Nigeria
M.Sc. Program: Electrical/Electronic Engineering
Dept., University of Ibadan, Ibadan, Nigeria

ABSTRACT
Engineering education is to move engineering students along the progressive path from being novices toward becoming experts in design, problem-solving and application of knowledge. Engineering problem may require more computations than is possible by hand. Computer-aided engineering is the process of solving engineering problems with the aid of computer software. Engineering Lecturers need to help engineering students to develop expertise in Computer-aided engineering with examples. The development of Discrete Wavelet Transformation is used as a case study. A wavelet is a small wave whose energy is concentrated in time. Wavelets applications include signal processing, noise filtering, image processing, and document analysis. Among wavelet families, Haar wavelet is selected. Scientific representations of the problem and a logical plan of attacking the problem are presented. Necessary equations are derived. A modular approach to programming is demonstrated. A complex problem is broken down into simple tasks and steps which are coded into simple short MATLAB programs. A program calls another program to execute some specific tasks. Programs are checked against possible errors using a situation where the answers are known. Discrete wavelet transformation and inverse discrete wavelet transformation for 1D, 2D, and 3D discrete-time signals have been implemented. 2D gray level images and 3D color images are also considered. The use of similar examples is recommended for Engineering Lecturers.

General Terms

Keywords

1. INTRODUCTION
The objective of engineering education is to move engineering students along the progressive path from being novices toward becoming experts in design, problem-solving and application of knowledge [1]. The Engineering Lecturers have the duty to lead the engineering students from what they know to what they do not know. Studies have revealed that students have misunderstandings with regards to a range of concepts [2], [3], [4]. Improving the understanding and technical know-how of students is tasks that should be accomplished by Lecturers.

Problem-solving is required whenever there is a goal to reach and attainment of that goal is not possible either by direct action or by retrieving a sequence of previously learned steps from memory [4], [5]. During problem-solving, the path to the intended goal is uncertain.

Some of the problems to be solved may be well-defined or ill-defined [7]. Most problems in class are well defined; initial conditions, goal, means for generating and evaluating the solution, and the constraints on the solution are specified. For the ill-defined problems, students need to define some of the problems’ components on their own [7]. Regular practice is important [8]. Practice makes perfect.

Formulation of scientific representations of the problem to be solved in the form of pictures, diagrams, graphs, maps, models, and simulations facilitates problem-solving. A problem representation is the model of the problem constructed by the solver to summarize his/her understanding of the problem. This model may include the elements in the problem, the inter-relationship of the elements, the goals and the types of operations that can be performed on the elements, and any constraints on the solution process.

Yildirim, Shuman and Besterfield-Sacre in [9] recommend that the expert should guide the novice toward success. The expert needs to point out the strengths and weaknesses of the novice’s product or performance. Teamwork among students to solve open-ended, real-world problems can also improve students’ problem-solving skills.

Solution to an engineering problem may require more computations than is possible by hand. A digital computer is, therefore, an essential tool in solving engineering problems [10], [11], [12]. Engineering students need to be familiar with Computer-aided engineering which is the process of solving engineering problems with the aid of computer software.

MATLAB computer programming system is a problem-solving tool for scientists and engineers [13], [14]. It has a teach-yourself feature. MATLAB is based on the mathematical concept of a matrix. Every variable X is a matrix of order m-by-n; m rows and n columns. A matrix is a rectangular array. A vector is a list or a matrix with 1 column or with 1 row. A constant k is a scalar and it’s a 1-by-1 matrix. Regarding all variables as matrices is an official MATLAB mindset [13].

Students need to develop a logical plan of attack to solve a problem and learn exact rules for writing MATLAB statements. MATLAB can handle basic arithmetic operations such as addition, subtraction, division, multiplication,
The development of Discrete Wavelet Transformation is taken as a case study.

A wavelet is defined as a small wave whose energy is concentrated in time [15], [16], [17], [18], [19]. Wavelets applications include signal processing, noise filtering, image processing, image compression, image recognition, document analysis, and feature extraction [18], [20], [21], [22]. Haar wavelet is selected. There are other wavelet families such as Daubechies (Db), Sinc, Gaussian and Mexican Hat as illustrated in Fig. 1 [15].

2. DISCRETE WAVELET TRANSFORM

2.1 Review of One-Dimensional DWT

Given a discrete-time signal X which is an N-by-1 matrix. It is required to determine Y, the one-dimensional discrete wavelet transform (1D DWT) of X at Lth decomposition level. For example, suppose X is given as in Eqn. (1).

\[
X = \begin{bmatrix}
2 \\
3 \\
1 \\
0 \\
4 \\
7 \\
6 \\
8
\end{bmatrix}
\]  

N is 8 in this example. Y is first set equal to X, \(Y = X\). Y is broken into \(y_1\) and \(y_2\) as shown in Fig. 2. \(y_1\) is set equal to Y as \(y_2\) is empty. \(y_1\) and \(y_2\) are used to multiply the wavelet transformation matrix \(W\) and an Identity Matrix I respectively for the first decomposition level as illustrated in Fig. 2 and as described by Eqns. (2). The wavelet transformation matrix \(W\) is given by Eqn. (3). \(W_1\) and \(W_2\) are the approximate and detail wavelet transformation matrices respectively; they are regarded as low pass (lp) filter and high pass (hp) filter respectively. Y is also decomposed into two parts as described by Eqns. (4), (5), (6); the approximate part \(Y_a\) which contains the low-frequency information of the signal and the details part \(Y_d\) which contains the high-frequency information of the signal.

The output \(Y\) at the end of the first decomposition level is feedback and packaged as \(y_1\) and \(y_2\) which are used to multiply the wavelet transformation matrix \(W\) and an Identity Matrix I respectively for the second decomposition level as shown in Fig. 3. The output \(Y\) at the end of the second decomposition level is feedback and packaged as \(y_1\) and \(y_2\) which are used to multiply the wavelet transformation matrix \(W\) and an Identity Matrix I respectively for the third decomposition level as shown in Fig. 4. At the end of every decomposition level, \(Y\) is decomposed into the approximate part \(Y_a\) and the details part \(Y_d\) as described by Eqns. (4), (5), (6).

\[
Y = \begin{bmatrix}
W_1 y_1 \\
I y_2
\end{bmatrix} = \begin{bmatrix}
W_1 y_1 \\
y_2
\end{bmatrix}  \tag{2}
\]
Fig 1: Some wavelet families

Fig 2: First decomposition level ($L=1$)

$$W = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}$$

(3)

$$Y_a = W_1 y_1$$

(4)

$$Y_d = \begin{bmatrix} W_2 y_1 \\ y_2 \end{bmatrix}$$

(5)

$$Y = [Y_a ; Y_d]$$

(6)

Observations on the relationship between the dimensions of $X$, $Y$, $W_1$, $W_2$, $y_1$, $y_2$, $Y_a$ and $Y_d$ on one hand and the decomposition level ($L$) and the number of elements ($N$) in $X$ on the other hand are summarized in Table 1. Both $X$ and $Y$ are $N$-by-1. Both $W_1$ and $W_2$ are $r$-by-($c$). $r$ is usually half of $c$. $y_1$ is $c$-by-1, $y_2$ is $(N-c)$-by-1, $Y_a$ is $r$-by-1, and $Y_d$ is $(N-r)$-by-1. $r$ and $c$ are related to $N$ and $L$ as indicated in Table 1 and Eqns. (7) and (8). These equations are deduced from the trends in Table 1. $W_1$ and $W_2$ are concatenated to give $W$ as in Eqn. (3) which is a MATLAB statement. Similarly, $W_1 y_1$ and $I y_2$ are concatenated to give $Y$ as in Eqn. (2).

$$r = \frac{N}{2^L}$$

(7)

$$c = \frac{N}{2^{L-1}}$$

(8)
### Fig 3: Second decomposition level ($L = 2$)

<table>
<thead>
<tr>
<th>$W_1$ (2-by-4)</th>
<th>$Y_1$ (4-by-1)</th>
<th>$Y$ (8-by-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{1}{\sqrt{2}}$</td>
<td>3.5355</td>
<td>3.0000</td>
</tr>
<tr>
<td>0</td>
<td>0.7071</td>
<td>12.5000</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{2}}$</td>
<td>7.7782</td>
<td>2.0000</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{2}}$</td>
<td>9.8995</td>
<td>-1.5000</td>
</tr>
<tr>
<td>1</td>
<td>-0.7071</td>
<td>-0.7071</td>
</tr>
<tr>
<td>0</td>
<td>0.7071</td>
<td>0.7071</td>
</tr>
<tr>
<td>0</td>
<td>-2.1213</td>
<td>-2.1213</td>
</tr>
<tr>
<td>0</td>
<td>-1.4142</td>
<td>-1.4142</td>
</tr>
</tbody>
</table>

### Fig 4: Third decomposition level ($L = 3$)

<table>
<thead>
<tr>
<th>$W_1$ (1-by-2)</th>
<th>$Y_1$ (2-by-1)</th>
<th>$Y$ (8-by-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{1}{\sqrt{2}}$</td>
<td>3.0000</td>
<td>10.9602</td>
</tr>
<tr>
<td>$\frac{1}{\sqrt{2}}$</td>
<td>12.5000</td>
<td>-6.7175</td>
</tr>
<tr>
<td>1</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>0</td>
<td>-1.5000</td>
<td>-1.5000</td>
</tr>
<tr>
<td>0</td>
<td>-0.7071</td>
<td>-0.7071</td>
</tr>
<tr>
<td>0</td>
<td>0.7071</td>
<td>0.7071</td>
</tr>
<tr>
<td>0</td>
<td>-2.1213</td>
<td>-2.1213</td>
</tr>
<tr>
<td>0</td>
<td>-1.4142</td>
<td>-1.4142</td>
</tr>
<tr>
<td>Decomposition Level</td>
<td>$W_1$</td>
<td>$W_2$</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>L=1</td>
<td>4-by-8</td>
<td>4-by-8</td>
</tr>
<tr>
<td>L=2</td>
<td>2-by-4</td>
<td>2-by-4</td>
</tr>
<tr>
<td>L=3</td>
<td>1-by-2</td>
<td>1-by-2</td>
</tr>
</tbody>
</table>

Generally for $L$ and $N$

For $L=1$, $r$-by-c

$$r = \frac{N}{2^L}$$

$$c = \frac{N}{2^{L-1}}$$

For $L=2$, $r$-by-c

$$r = \frac{N}{2^L}$$

$$c = \frac{N}{2^{L-1}}$$

$X$ is $N$-by-1; ($N = 8$ in this example.)

$$W = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}; \quad Y = \begin{bmatrix} W y_1 \\ I y_2 \end{bmatrix}; \quad Y = \begin{bmatrix} Y_a \\ Y_d \end{bmatrix}; \quad Y$ is also $N$-by-1

$W_1$ and $W_2$ are the approximate and detail wavelet transformation matrices respectively. Most of the elements of $W_1$ and $W_2$ are zeroes. Few of the elements of $W_1$ and $W_2$ are $\frac{1}{\sqrt{2}}$ or $-\frac{1}{\sqrt{2}}$. Eqns. (9) and (10) set all elements of $W_1$ and $W_2$ to zero. A careful review of $W_1$ in Figs. 4, 5 and 6 shows that on the $k^{th}$ row, the (2$k$)th column and the (2$k$-1)th column are both $\frac{1}{\sqrt{2}}$. A careful review of $W_2$ in Figs. 4, 5 and 6 shows that on the $k^{th}$ row, the (2$k$)th column and the (2$k$-1)th column are $-\frac{1}{\sqrt{2}}$ and $\frac{1}{\sqrt{2}}$ respectively. Therefore, Eqns. (11), (12), (13) and (14) set specific elements of $W_1$ and $W_2$ to $\frac{1}{\sqrt{2}}$ or $-\frac{1}{\sqrt{2}}$ as appropriate. The solution is repetitive for each decomposition level $L=1, 2, 3$, up to the specified level $L=\text{max}$.

$$W_1 = \text{zeros}(r, c)$$  

$$W_2 = \text{zeros}(r, c)$$  

$$W_1(k, (2k-1)) = \frac{1}{\sqrt{2}}$$  

$$W_1(k, 2k) = \frac{1}{\sqrt{2}}$$  

$$W_2(k, (2k-1)) = \frac{1}{\sqrt{2}}$$  

$$W_2(k, 2k) = \frac{1}{\sqrt{2}}$$

### 2.2 Basic Haar Wavelet Program

Given the $Y$ at the end of last decomposition level, it is required to determine $r$, $c$, $y_1$, $y_2$, $W_1$, and $W_2$ using Eqns. (7) to (14) and update $Y$ using Eqns. (2) and (3) for the next decomposition level ($L$). A program is developed to carry out this task. This program is named haarmat. Its flow chart and MATLAB codes are presented in Figs. 5 and 6 respectively.

The inverse transform program INVhaarbasic is presented in Fig. 7. INVhaarbasic is similar to haarbasic except for the fact that Eqn. (2) is replaced by Eqn. (15).

$$Y = \begin{bmatrix} W^{-1} y_1 \\ I y_2 \end{bmatrix} = \begin{bmatrix} W^{-1} y_1 \\ y_2 \end{bmatrix}$$  

$$r = \frac{N}{2^L}; \quad c = \frac{N}{2^{L-1}}$$

$$y_i = [Y(1,1), Y(2,1),..., Y(c,1)]$$

$$y_i = \begin{cases} [Y(c+1,1), Y(c+2,1),..., Y(N,1)] & \text{if } L > 1 \\
[j](\text{null}) & \text{if } L = 1 \end{cases}$$

$$W_1 = \text{zeros}(r, c); \quad W_2 = \text{zeros}(r, c); \quad k = 0$$

**Fig 5:** Flow chart of the haarbasic program for decomposition level $L$. 

---

**Table 1. Observations on dimensions of variables**

<table>
<thead>
<tr>
<th>Decomposition Level</th>
<th>$W_1$</th>
<th>$W_2$</th>
<th>$W$</th>
<th>$I$</th>
<th>$y_1$</th>
<th>$y_2$</th>
<th>$Y_a$</th>
<th>$Y_d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>L=1</td>
<td>4-by-8</td>
<td>4-by-8</td>
<td>8-by-8</td>
<td>-</td>
<td>8-by-1</td>
<td>0-by-1</td>
<td>4-by-1</td>
<td>4-by-1</td>
</tr>
<tr>
<td>L=2</td>
<td>2-by-4</td>
<td>2-by-4</td>
<td>4-by-4</td>
<td>4-by-4</td>
<td>4-by-4</td>
<td>4-by-1</td>
<td>2-by-1</td>
<td>6-by-1</td>
</tr>
<tr>
<td>L=3</td>
<td>1-by-2</td>
<td>1-by-2</td>
<td>2-by-2</td>
<td>6-by-6</td>
<td>2-by-1</td>
<td>6-by-1</td>
<td>1-by-1</td>
<td>7-by-1</td>
</tr>
</tbody>
</table>
Fig 6: haarbasic program for decomposition level L.

```
x=N/2^L; c=N/2^L (L-l);  
y=1 = (1+c,l);  
if L=1  
y2=[];  
else  
y2=Y(1+c:N,l);  
end  
W1=zeros(r,c);  
W2=zeros(r,c);  
for k=l:x  
W1(k,(2*k-1))=1/2^-.5;  
W2(k,(2*k-1))=1/2^-.5;  
W1(k,(2*k))=1/2^-.5;  
W2(k,(2*k))=-1/2^-.5;  
end  
W=[W1,W2];  
y=W*y1;  
Y=[y1;y2];  
```

Fig 7: INVhaarbasic program for decomposition level L.

```
x=N/2^L; c=N/2^L (L-l);  
y=1 = (1+c,l);  
if L=1  
y2=[];  
else  
y2=Y(1+c:N,l);  
end  
W1=zeros(r,c);  
W2=zeros(r,c);  
for k=l:x  
W1(k,(2*k-1))=1/2^-.5;  
W2(k,(2*k-1))=1/2^-.5;  
W1(k,(2*k))=1/2^-.5;  
W2(k,(2*k))=-1/2^-.5;  
end  
W=[W1,W2];  
y=W*y1;  
Y=[y1;y2];  
```

2.3 One-Dimensional DWT Program

2.3.1 Column Signal

Given an input column signal X (N-by-M) where \( M = 1 \), it is required to determine Y, the one-dimensional discrete wavelet transform (1D DWT) of X at \( L_m \)th decomposition level. Initialize Y as \( Y = X \). Starting with \( L = 1 \), call the haarbasic program, increase L and repeat the call of the haarbasic program up to \( L = L_m \) as illustrated in the flowchart of Fig. 8. Y obtained at the end of \( L = L_m \) is the required 1D DWT of X.

The flow chart of Fig. 8 is coded into a MATLAB program named dwt1Dc and presented in Fig. 9. The inverse transform program INVdwt1Dc presented in Fig. 10 is similar to dwt1Dc except that L decreases from \( L_m \) to 1 in INVdwt1Dc whereas L increases from 1 to \( L_m \) in dwt1Dc.

2.3.2 Row Signal

Given an input row signal X (M-by-N) where \( M = 1 \), it is required to determine Y, the one-dimensional discrete wavelet transform (1D DWT) of X at \( L_m \)th decomposition level.

Change X from being a row signal into a column signal. Call dwt1Dc program. Change the final output Y from being a column signal to a row signal. The new program is named dwt1Dr and is presented in Fig 11. The inverse transform program INVdwt1Dr presented in Fig. 12 is similar to dwt1Dr. Change Y from being a row signal into a column signal. Call INVdwt1Dc program. Change the final output X from being a column signal to a row signal.
X=X';
ctr1Dc
Y=Y';
INVdwt1Dc

Fig 11: dwt1Dr program for a row 1D signal

Fig 12: INVdwt1Dr program for a row 1D signal

2.4 Two-Dimensional DWT Program
Given a two-dimensional signal or gray level image XX (NN-by-MM), it is required to determine YY, the two-dimensional discrete wavelet transform (2D DWT) of XX at LLmth decomposition level. Initialize YY as YY = XX. If XX is a gray level image, XX should be converted from uint8 format to double format before applying it as input. The output YY should be converted from double format to uint8 format before displaying it as an image.

The 2D DWT of a signal or image can be partitioned into four regions or components namely: Approximation (A), Horizontal details (HD), Vertical details (VD), and Diagonal details (DD) as illustrated in Fig. 13. A, HD, VD, and DD are the Low/Low (LL), Low/High (LH), High/Low (HL) and the High/High (HH) frequency components of the signal respectively. At the next decomposition level, the Approximation part is further divided into another set of A, HD, VD, and DD as shown in Fig. 13.

1D DWT and inverse 1D DWT programs are required here as subprograms. The outputs of these subprograms constitute intermediate results in 2D DWT program and such outputs may be disabled from being published with ‘;’.

Starting with the decomposition level LL = 1, work on the entire YY (NN-by-MM). Take the first row in YY as X and call dwt1Dr with p_m = 1. Update YY by replacing the first row of YY with the Y obtained from dwt1Dr. Repeat for all the rows in YY. This is illustrated in Fig. 14 (a) tagged as the first step. Take the first column in the updated YY as X and call dwt1Dc with p_m = 1. Update YY by replacing the first column of YY with the Y obtained from dwt1Dc. Repeat for all the columns in YY as illustrated in Fig. 14 (b) tagged as the second step. The YY at the end of the second step is the two-dimensional discrete wavelet transform (2D DWT) for LL = 1.

For LL = LL, select the Approximation component which is the NN/LL-by-MM/LL top-left-portion of the updated YY and repeat the procedure above. The remaining portions of YY remain unchanged. For LL = 2, 2-by-2 top-left-portion of YY is taken and the procedure is repeated as illustrated in Fig. 14 (c) and (d) tagged third and fourth step respectively. Increase LL from 1 up to LL_m. In Fig. 14, the operations stopped at LL = 2 because LL_m is given as 2. A program named as dwt2D is developed and presented in Fig. 15. MATLAB code ‘clear’ is introduced in this program during test running when an error was observed. This code deletes a variable in the workspace or memory. For example, when LL changed from 1 to 2 in Fig. 14, the dimension of both X and Y changed from 1-by-4 to 1-by-2. If X and Y are not cleared before the change in LL, X and Y will remain 1-by-4 after the change in LL instead of 1-by-2 which actually led to an error during test running.

The steps for the inverse two-dimensional discrete wavelet transform program INVdwt2D of Fig. 16 are similar. In dwt2D, LL increases from 1 up to LL_m and rows are operated before columns. In INVdwt2D, LL decreases from LL_m down to 1 and columns are operated before rows.

2.5 Three-Dimensional DWT Program
Given a three-dimensional signal or color image x (NN-by-MM-by-3), it is required to determine y, the three-dimensional discrete wavelet transform (3D DWT) of y at LL_mth decomposition level. This is accomplished by simply finding the 2D DWT of each of the red, green and blue components of x. dwt2D is executed three times as indicated in the dwt3D program of Fig. 17.

The first line in Fig. 17 converts the color image from the uint8 format to the double format. The output 3D DWT (y) should not be converted from the double format to the uint8 format otherwise original x will not be recoverable from y via the INVdwt3D program of Fig. 18 which is the inverse 3D DWT program. The last line in Fig. 17 shows the image of uint8 version of y on a fresh figure. The second to the last line in Fig. 18 converts the inverse 3D DWT from the double format back to the uint8 format.
RESULTS AND DISCUSSIONS
Validation tests for the 1D and 2D DWT and inverse DWT programs were carried out for situations when the results are known. As shown in Figs. 19, 20, 21 and 22, the outputs of the 1D and 2D DWT and inverse DWT programs are found to be valid. When the output of the DWT program is supplied as input to the corresponding inverse DWT program, the output of the inverse DWT program is found in all cases to be equal to the initial input to the DWT program as illustrated in Figs. 19, 20, 21 and 22.

2D DWT and inverse 2D DWT programs are also tested with a gray level image. The results are presented in Figs. 23 (a) and (b). YY in Fig. 23(b) is in double format which gives Fig. 23(c) when converted to uint8 format. Application of image enhancement on the Approximation component gives Fig. 23(d). The image enhancement is accomplished by replacing each element e of the approximation component with 

\[ p_e = 7.0 \]

The approximation component is thus a version of the original input image but with reduced size (dimension). The VD, HD, and DD give the vertical, horizontal and diagonal edges (details) of the input image.

3D DWT and inverse 3D DWT programs are tested with a color image. The results are presented in Figs. 24 (a) and (b). YY in Fig. 24(b) is in the double format which gives Fig. 24(c)
when converted to the uint8 format. Application of image enhancement on the Approximation component gives Fig. 24(d). The image enhancement is accomplished by replacing each element \( e \) of the approximation component with \( e' \) where \( p = 0.8 \).

```matlab
x = double(x);
for t = 1:3
    XX = double(x(:, :, t));
dwt2D
    y(:, :, t) = YY;
end
figure; imshow(uint8(y));
```

**Fig 17: dwt3D program for 3D signal**

```matlab
for t = 1:3
    YY = y(:, :, t);
    INVdwt2D
    x(:, :, t) = XX;
end
x = uint8(x);
figure; imshow(x);
```

**Fig 18: INVdwt3D program for 3D signal**

<table>
<thead>
<tr>
<th>X</th>
<th>Y(( L_0 = 3 ))</th>
<th>Ya</th>
<th>Yd</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10.9602</td>
<td>-6.7175</td>
<td>-6.7175</td>
</tr>
<tr>
<td>3</td>
<td>-6.7175</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>1</td>
<td>-1.5000</td>
<td>-1.5000</td>
<td>-1.5000</td>
</tr>
<tr>
<td>0</td>
<td>0.7071</td>
<td>0.7071</td>
<td>0.7071</td>
</tr>
<tr>
<td>4</td>
<td>-0.7071</td>
<td>-2.1213</td>
<td>-2.1213</td>
</tr>
<tr>
<td>7</td>
<td>0.7071</td>
<td>-1.4142</td>
<td>-1.4142</td>
</tr>
<tr>
<td>6</td>
<td>-2.1213</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-1.4142</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Fig 19: dwt1Dr and INVdwt1Dr results**

<table>
<thead>
<tr>
<th>X</th>
<th>Y(( L_0 = 3 ))</th>
<th>Ya</th>
<th>Yd</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10.9602</td>
<td>-6.7175</td>
<td>-6.7175</td>
</tr>
<tr>
<td>3</td>
<td>-6.7175</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>1</td>
<td>-1.5000</td>
<td>-1.5000</td>
<td>-1.5000</td>
</tr>
<tr>
<td>0</td>
<td>0.7071</td>
<td>0.7071</td>
<td>0.7071</td>
</tr>
<tr>
<td>4</td>
<td>-0.7071</td>
<td>-2.1213</td>
<td>-2.1213</td>
</tr>
<tr>
<td>7</td>
<td>0.7071</td>
<td>-1.4142</td>
<td>-1.4142</td>
</tr>
<tr>
<td>6</td>
<td>-2.1213</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>-1.4142</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Fig 20: dwt1Dr and INVdwt1Dr results**

<table>
<thead>
<tr>
<th>XX</th>
<th>INVdwt2D</th>
</tr>
</thead>
<tbody>
<tr>
<td>125</td>
<td>165</td>
</tr>
<tr>
<td>159</td>
<td>58</td>
</tr>
<tr>
<td>112</td>
<td>97</td>
</tr>
<tr>
<td>150</td>
<td>44</td>
</tr>
<tr>
<td>114</td>
<td>207</td>
</tr>
<tr>
<td>53</td>
<td>59</td>
</tr>
<tr>
<td>79</td>
<td>136</td>
</tr>
<tr>
<td>77</td>
<td>112</td>
</tr>
</tbody>
</table>

**Fig 21: dwt2D and INVdwt2D results for \( LL_0 = 2 \)**
4. CONCLUSIONS

Development of a logical plan of attack in solving a problem has been demonstrated. Scientific representations of the problem in the form of diagrams were presented. Arithmetic and logical relationships between the input, output and intermediate variables in the form of mathematical equations were derived. Simple relevant tools of MATLAB were used in developing the programs for solving the problem. Programs were developed in modules. A program may call another program for some specific tasks. A complex problem is solved by simple programs instead of one long complex program. Simple short programs are easier to develop and debug.
Programs are validated with situations where the answers are known. Discrete wavelet transformation and inverse discrete wavelet transformation have been demystified. Discrete wavelet transformation and inverse discrete wavelet transformation for 1D, 2D, and 3D discrete-time signals in general and gray level images and color images in particular have been implemented and can be applied in Discrete Signal Processing and Digital Image Processing. The use of similar examples is recommended for Engineering Lecturers for transforming Engineering students from novices to experts vis-a-vis computer-aided engineering and problem-solving.

5. REFERENCES


