**Wavelet Toolbox**

**Author Information**：

Tian Ye / [tye459@usc.edu](mailto:tye459@usc.edu) / [tian\_ye1994@hotmail.com](mailto:tian_ye1994@hotmail.com) (preferred) / 323-616-7117 (US)

**Description:**

This project implements discrete wavelet transform and apply it on image classification problem. The data set is att\_face, which is a face data set containing 400 images of 40 people face. Each person has 10 images with different expression (happy, curious...). The goal of the project is to use wavelet transform to extract feature from different levels and present those images with a feature vector. A test image will be classified to a label according to its extracted feature vector.

In the next three section, a brief theoretical description on wavelet transform will be discussed in S1, followed by the implementation detail in S2 and finished with result log file in S3.

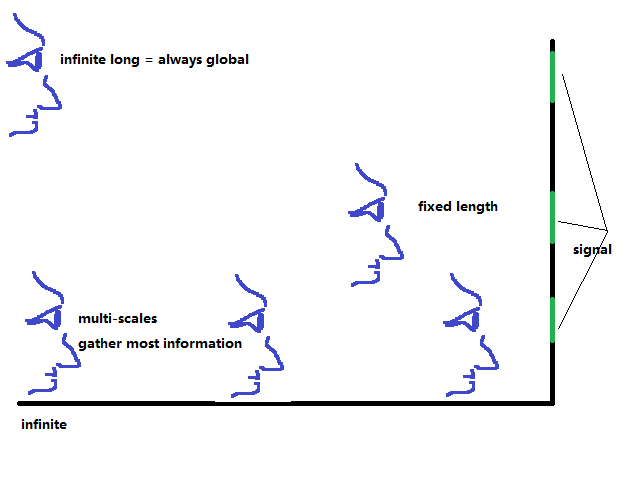
**section 1：Brief theory**

If you take linear Algeria point of view, almost all the transform in signal processing field is basically decomposing the original signal to a series of basis. It is kind of like projecting a space to multi-subspace, and how much is its projection on a specific subspace, or direction, can be seen as a feature.

In the very basic Fourier transform, the signal is decomposed to frequency slots, then we can know the proportion of each frequency’s energy, which forms a feature vector. However, this can only be applied on the whole signal. If the signal is 100 seconds duration, then the energy analysis is based on the whole 100 seconds. If someone want to know what it is like at 15 second, he needs to modify the transform by adding a window, which forms short time Fourier transform. By windowing, 100 seconds can be divided into 200 half second, for instance, and we can use the information extracted from those half seconds to present an instant state.

However, how long should the window be is a trap. Too long, then the information of the frame is too mixed and time resolution is bad. Too short, then the information is not completed enough, making the followed analysis meaningless. In addition, even you choose an optimized window size for current frame, because the real world signal always changes rapidly, the size may not works well for the rest of the frame.

Although wavelet transform is not designed to solve those problem, but it does improve those issues greatly. First, it no longer use fix length window, it do “multi-scale” transform, which is equivalent to have multi-window size setting. Second, the transform basis is no longer infinite long, but finite long, which means it won't need a window to solve time resolution issue. Most important thing is the transform basis is no longer fixed, you can choose optimized basis for different tasks. All of those lead to much powerful feature extracted by wavelet transform than conventional methods.



In this project we need to classify faces. Consider we just apply 2D discrete Fourier transform, what we extract is just level one global frequency information. And our images are so closed to each other, poor accuracy is expected. If the theory makes sense, then we should see fine test accuracy in the end.

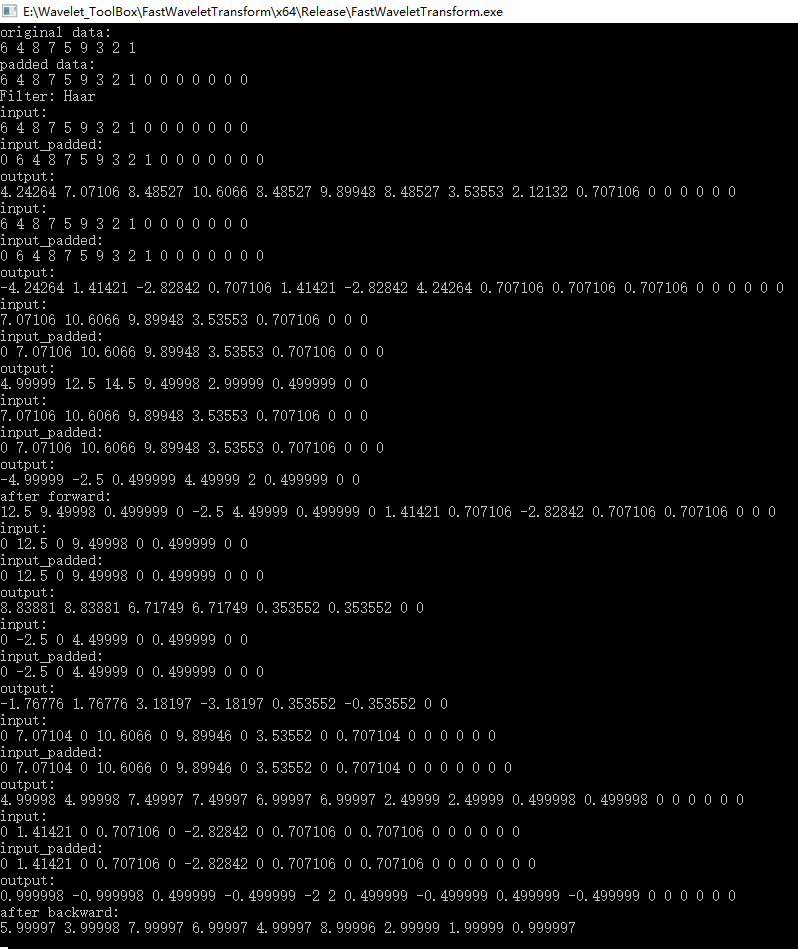
**section 2：Implementation**

Mallat algorithm is used. In each level, a highpass and a corresponding lowpass filter is applied to the signal, two response are generated. Down Sample and persevere the high frequency part, the low frequency will be reused to perform a next level transform. When recover, perform deconvolution on response with recover highpass and lowpass filter and add them together.

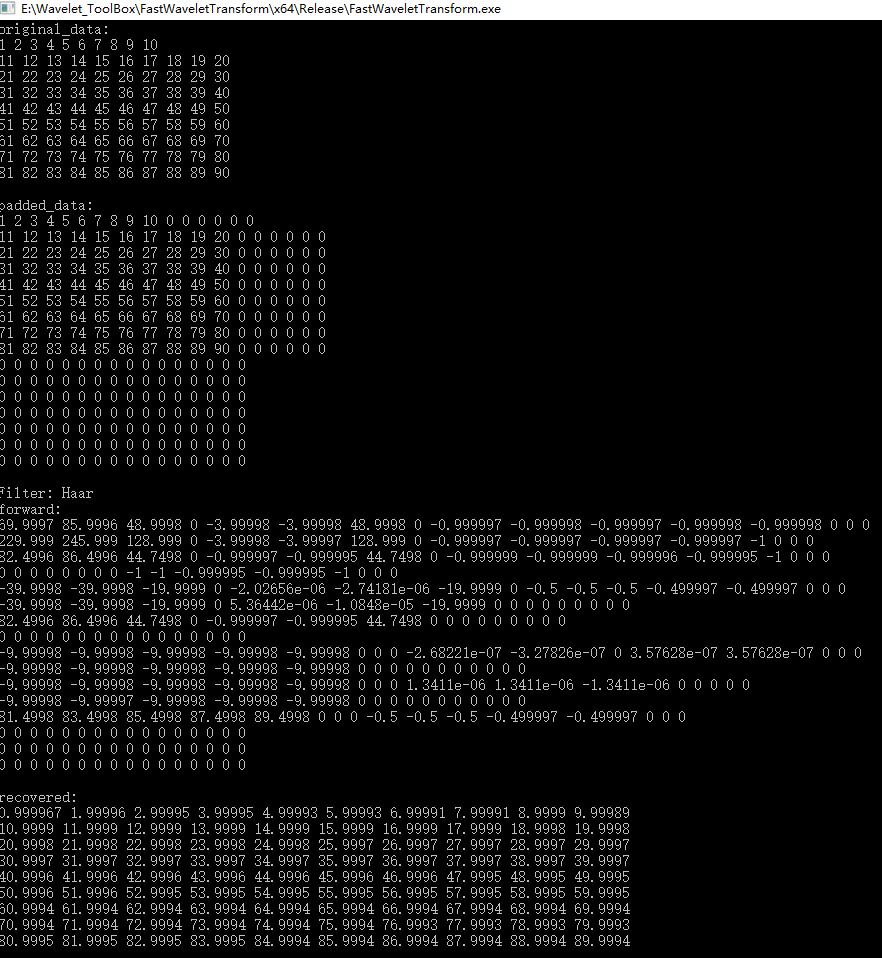
The way to perform 2D transform is almost exactly the same expect for perform 2 times for each level. First perform 1D transform / reconstruct on all rows, then perform the same operation on all cols.

To make the transform more standard, i constrain the length of the input must be 2^N. If the signal can not satisfy this, zeros will be padded at the end of the signal. I have already integrated some common filters like Haar and DpN.

In the code i pack all those function as a class named WaveletToolBox. To use the function, first create a WaveletToolBox object. 1D array comes in as std::vector while 2D array comes in std::vector<std::vector>. Function Test1D and Test2D show how to perform a multi-level forward transform and backward reconstruct. The output log of them are below:



1D\_2\_level\_forward&backward



2D\_2\_level\_forward&backward

**section 3：Classification**

The image sets are seperated to 2 part: training set (280 images) and test part (120 images). For training images, we compute their mean and subtract the mean from all 400 images in the data set. Then we perform 6 levels DWT on all of them. For response images in each level, we DO NOT gather LowLow (frequency low in both x and y axis) response. For other response, compute their energy and set all of those energy as a 1D array to form a feature vector.

Now each image is presented as a vector. For a test image, compute its distance from all of the train images’ vectors and assign the label to the one whose distance is the smallest.

Finally we gather the accuracy for both test images only. (because in this case distance for training images will always be zero so that the accuracy will always be 1) The result is shown as below:

real label: 1 predicted label: 1 distance: 3.49884

real label: 1 predicted label: 1 distance: 4.06054

real label: 1 predicted label: 1 distance: 2.68804

real label: 2 predicted label: 2 distance: 1.28339

real label: 2 predicted label: 2 distance: 2.35137

real label: 2 predicted label: 2 distance: 1.32549

real label: 3 predicted label: 3 distance: 1.03677

real label: 3 predicted label: 4 distance: 1.59453

real label: 3 predicted label: 3 distance: 1.81133

real label: 4 predicted label: 4 distance: 0.787604

real label: 4 predicted label: 4 distance: 0.873002

real label: 4 predicted label: 18 distance: 1.23302

real label: 5 predicted label: 40 distance: 1.6393

real label: 5 predicted label: 5 distance: 0.871625

real label: 5 predicted label: 5 distance: 1.66428

real label: 6 predicted label: 6 distance: 1.0593

real label: 6 predicted label: 6 distance: 1.86546

real label: 6 predicted label: 6 distance: 0.976311

real label: 7 predicted label: 7 distance: 1.59191

real label: 7 predicted label: 7 distance: 1.40003

real label: 7 predicted label: 7 distance: 2.29256

real label: 8 predicted label: 8 distance: 1.50737

real label: 8 predicted label: 8 distance: 1.88159

real label: 8 predicted label: 8 distance: 1.03722

real label: 9 predicted label: 9 distance: 1.20584

real label: 9 predicted label: 9 distance: 1.77709

real label: 9 predicted label: 9 distance: 1.89096

real label: 10 predicted label: 10 distance: 1.71781

real label: 10 predicted label: 8 distance: 2.84814

real label: 10 predicted label: 10 distance: 2.03072

real label: 11 predicted label: 11 distance: 2.56131

real label: 11 predicted label: 11 distance: 1.80361

real label: 11 predicted label: 11 distance: 1.41608

real label: 12 predicted label: 12 distance: 1.74767

real label: 12 predicted label: 3 distance: 2.33919

real label: 12 predicted label: 7 distance: 1.86613

real label: 13 predicted label: 13 distance: 0.616496

real label: 13 predicted label: 13 distance: 1.70764

real label: 13 predicted label: 13 distance: 0.538492

real label: 14 predicted label: 14 distance: 1.53222

real label: 14 predicted label: 14 distance: 2.13979

real label: 14 predicted label: 14 distance: 1.84261

real label: 15 predicted label: 2 distance: 2.53339

real label: 15 predicted label: 15 distance: 1.12106

real label: 15 predicted label: 15 distance: 0.831282

real label: 16 predicted label: 16 distance: 2.48601

real label: 16 predicted label: 16 distance: 2.24627

real label: 16 predicted label: 1 distance: 2.20635

real label: 17 predicted label: 17 distance: 1.82202

real label: 17 predicted label: 17 distance: 0.886406

real label: 17 predicted label: 17 distance: 1.45385

real label: 18 predicted label: 18 distance: 0.745387

real label: 18 predicted label: 18 distance: 1.40892

real label: 18 predicted label: 5 distance: 1.92148

real label: 19 predicted label: 19 distance: 1.67783

real label: 19 predicted label: 19 distance: 2.20401

real label: 19 predicted label: 19 distance: 2.43952

real label: 20 predicted label: 20 distance: 1.81433

real label: 20 predicted label: 20 distance: 1.82378

real label: 20 predicted label: 20 distance: 1.32713

real label: 21 predicted label: 33 distance: 1.84128

real label: 21 predicted label: 21 distance: 1.09909

real label: 21 predicted label: 21 distance: 1.35734

real label: 22 predicted label: 22 distance: 0.991982

real label: 22 predicted label: 22 distance: 1.04424

real label: 22 predicted label: 22 distance: 1.29588

real label: 23 predicted label: 23 distance: 1.00264

real label: 23 predicted label: 23 distance: 2.20825

real label: 23 predicted label: 23 distance: 0.962519

real label: 24 predicted label: 24 distance: 0.982965

real label: 24 predicted label: 24 distance: 1.51767

real label: 24 predicted label: 24 distance: 1.84383

real label: 25 predicted label: 25 distance: 1.54179

real label: 25 predicted label: 25 distance: 0.77193

real label: 25 predicted label: 25 distance: 0.723693

real label: 26 predicted label: 26 distance: 2.02047

real label: 26 predicted label: 26 distance: 1.43019

real label: 26 predicted label: 26 distance: 1.90731

real label: 27 predicted label: 27 distance: 0.79217

real label: 27 predicted label: 27 distance: 1.98859

real label: 27 predicted label: 27 distance: 2.27532

real label: 28 predicted label: 26 distance: 2.72467

real label: 28 predicted label: 28 distance: 3.15805

real label: 28 predicted label: 28 distance: 2.07538

real label: 29 predicted label: 29 distance: 1.37675

real label: 29 predicted label: 29 distance: 2.01364

real label: 29 predicted label: 29 distance: 1.09317

real label: 30 predicted label: 30 distance: 1.15239

real label: 30 predicted label: 30 distance: 1.12407

real label: 30 predicted label: 30 distance: 0.723679

real label: 31 predicted label: 31 distance: 0.785801

real label: 31 predicted label: 31 distance: 1.4594

real label: 31 predicted label: 31 distance: 1.54884

real label: 32 predicted label: 32 distance: 1.81273

real label: 32 predicted label: 32 distance: 1.86623

real label: 32 predicted label: 32 distance: 1.32198

real label: 33 predicted label: 30 distance: 1.94912

real label: 33 predicted label: 33 distance: 1.4484

real label: 33 predicted label: 33 distance: 0.645682

real label: 34 predicted label: 34 distance: 0.854715

real label: 34 predicted label: 34 distance: 1.09658

real label: 34 predicted label: 34 distance: 0.863092

real label: 35 predicted label: 35 distance: 1.17699

real label: 35 predicted label: 35 distance: 2.82917

real label: 35 predicted label: 35 distance: 1.11039

real label: 36 predicted label: 36 distance: 1.67479

real label: 36 predicted label: 36 distance: 2.07696

real label: 36 predicted label: 7 distance: 2.91865

real label: 37 predicted label: 37 distance: 1.5732

real label: 37 predicted label: 37 distance: 1.53643

real label: 37 predicted label: 37 distance: 2.14517

real label: 38 predicted label: 38 distance: 1.11205

real label: 38 predicted label: 38 distance: 1.67468

real label: 38 predicted label: 38 distance: 1.34414

real label: 39 predicted label: 39 distance: 1.46311

real label: 39 predicted label: 39 distance: 2.05778

real label: 39 predicted label: 39 distance: 1.67475

real label: 40 predicted label: 40 distance: 1.09502

real label: 40 predicted label: 40 distance: 2.2421

real label: 40 predicted label: 40 distance: 1.90141

**Total number of test cases:(test) 120**

**# correct: 107 test accuracy: 0.891667**