WAIR OVERVIEW

The WAIR software is a tool for quantitative analysis of various n-dimensional (n-D) image registration techniques. In particular, its applications include, but are not limited to, analyzing warp performance for stereotactic Human Brain anatomical and functional data. Using the Discrete Wavelet Transform (WT) we develop several image registration (warp) classification schemes. The first one, called "wavelet-space triangle analysis", is applicable for studying a family of warps on a single or multiple n-D data sets. For each data set the WAIR routine assigns a positive real number to every warp alignment in the family, and the best warp, for the given data, will be the one having the smallest value associated with it. The second classification method, called "cluster group classification", is applicable for analyzing the overall performance of a family of warps of a groups of data sets. Here, there is a single number assigned to each registration alignment, based on its group-clustering characteristics. Third goodness of warp approach, called SGC (spread group classification), is applicable for analyzing functional brain data. It gives preference to registration techniques that spread apart baseline versus activation functional signal for group data.

Both, the multi-dimensional discrete WT and the corresponding "triangle", "cluster-group" and "spread-group" classification approaches are included in the package. All routines are invoked as standard UNIX commands and (with one exception) neither expect nor allow user interaction once the command has been issued. The series of 'C' subroutines which comprise the WAIR library can be easily incorporated into the user's site specific programs adapted to their particular needs.

Currently, the WAIR package does not provide any graphics or image display capabilities. And no file format converters are included in this software. The user is supposed to convert their original data into 4 byte floating-point raw data files, then execute the WAIR routines to obtain the desired warp ranking.

This package was originally developed to work on Sun SPARC stations using the 'C' language compiler provided by Sun as part of the standard system software. If you currently use an 8 bits/pixel file format on a Sun SPARC station equipped with the standard 'C' compiler, have at least (XX Bytes + 16 MB) of RAM (where XX = 4*(Number of data sets)*(Number of warps)*(Dimension_1 * ... *Dimension_n) Bytes), and are familiar with the UNIX operating system, you should be able to install and use the WAIR package without additional assistance, even if you know nothing about 'C' programming. However, if any of these conditions is not met, it is likely that you will need the assistance of a 'C' programmer who is familiar with the UNIX operating system.

There are several 2D test images provided in the subdirectory "test_data.dir/" that can be used to inspect the installation of the software, as well as, for testing and interpreting the results of the wavelet analysis, both analytically and visually.

INSTALLATION AND COMPILING

- Extract the files in a new empty directory using the following UNIX commands:
  - uncompress WAIR2.0.tar.Z
  - tar xvf WAIR2.0.tar
- Installing WAIR2.0
  - Read the "README" file first
  - Inside the WAIR2.0.dir %> make -f Makefile_IDD
- Installing (the SGI-specific, ViewKit Vk Library used) Graphical User Interface (GUI) to WAIR 2.0
  - Read the README files within each GUI_WAIR_2_0.dir sub-directory
  - Install different parts of GUI_WAIR by typing "csh README" on the command line, within each GUI_WAIR_2_0.dir sub-directory separately
  - To start the GUI type "./GUI_WAIR_2_0.dir/GUI_WAIR" within your WAIR2.0.dir directory

SYNOPSIS OF THE WAIR 2.0 COMMANDS

The WAIR 2.0 software is a tool for automated, fast and robust quantitative analysis of various image-registration (warping) techniques applied to a single or multiple data sets. Warp performance is evaluated using the wavelet transforms of the warped-resliced data. Several different schemes for measuring goodness of image registration in "compressed" wavelet space are employed.

The main **routines** in the WAIR 2.0 package are **WT_IWT**, **wave_space_varThresh**, **CGC_WTC**, **SGC_CWT** and **triangleWTC**. The first one is used to find the (multi-dimensional) discrete wavelet and the inverse transforms of the resliced data. The second is employed to obtain the thresholded signals, in wavelet space, which are used by the following routines to numerically characterize warp performance based on different classification approaches.

Before invoking the wavelet-based warp classification programs one needs to obtain the WT's of the data (using WT_IWT) and the thresholded WT's (using wave_space_varThresh). This last routine, **wave_space_varThresh** is used interactively and allows choosing one of three wavelet shrinkage schemes: Uniform (at different levels), Donoho-Johnstone (DJ) and Dinov-Sumners (DS) thresholding. For more advanced users we recommend the use of the automated thresholding routine **wave_space_varThresh_inter**.

One way to numerically analyze warp performance on a single or multiple data is the "wavelet-space triangle" scheme. It is quite different from the CGC and SGC techniques (see the technical notes) because it uses the original data prior to warping and the target of the warp in determining warp ranking in reduced wavelet space. The "triangle" method is applied by calling the procedure **triangleWTC**. To quantitatively evaluate warp performance using functional (e.g., PET) data one employs the routine **SGC_CWT**. This technique ranks alignment algorithms based on how far apart they warp groups (different activation paradigms) of data (across subjects). To evaluate image registration using the same SGC (Spread Group

Classification) scheme without thresholding (equivalent to spatial domain analysis) one omits the wavelet-thresholding step (wave_space_varThresh) of the analysis.

There are several auxiliary routines in WAIR 2.0 that are frequently used for visual interpretation of low-dimensional (1D, 2D) wavelet shrinkage and compression. The program **MultiDim_IWT_varThresh**, can be employed for inverting the wavelet transform directly (without going through "wave_space_varThresh") using one of three different wavelet-thresholding approaches - Uniform (at various levels), Donoho-Johnstone (DJ) and Dinov-Sumners (DS).

ROUTINES

- **WT_IWT**

  Purpose:
  This WAIR command is used to obtain the n-Dimensional Discrete Wavelet Transform and its Inverse. Usually 1<= n <= 4, though this is not a real limitation, any positive integer n would work, however, the computational complexity increases exponentially with the increase of dimensionality. The user specifies the wavelet filter, the number of dimensions of the data and the size of each dimension.
  The program will generate a "*.log" output file that contains some information about the WT (or IWT) of the data (e.g., min/max values).

  Usage:
  WT_IWT *Wavelet Basis Which Transform Nomadic x_1_Dim_Size [[x_k_Dim_Size]] data_in.img WT_data_out.img*
  where the following definitions are used:

  WaveletBasis
  Wavelet basis used to determine the WT/IWT of the signal.
  Choose one of the following:
  WaveletBasis==1 , Spline_3_7;
  WaveletBasis==2 , Spline_3_3;
  WaveletBasis==3 , Spline_2_4;
  WaveletBasis==4 , Spline_2_2;
  WaveletBasis==5 , Pseudocoiflet_4_4;
  WaveletBasis==6 , Haar;
  WaveletBasis==7 , Daubechies_20;
  WaveletBasis==8 , Daubechies_12;
  WaveletBasis==9 , Daubechies_10;
  WaveletBasis==10 , Daubechies_8;
  WaveletBasis==11 , Daubechies_6;
  WaveletBasis==12 , Daubechies_4;
  WaveletBasis==13 , Coiflet_6;
  WaveletBasis==14 , Coiflet_4;
  WaveletBasis==15 , Coiflet_2;
  WaveletBasis==16 , BurtAdelson;

WaveletBasis==17 , BattleLemarie;

WhichTransform
Choose "1" for WT and "0" for IWT

Num_Dim
the number of dimensions of the data set

x_1_Dim_Size
the size of the first (fastest varying index) dimension of the data. Remember, all dimension sizes need to be **powers of 2**

x_k_Dim_Size
the size of the k-th dimension of the data

data_in.img
name of the file containing the input raw (4 byte) data set

WT_data_out.img
name of the output file that will contain the floating-point array of the WT of the input data

Examples:
./WT_IWT 7 1 2 256 256 ./data.dir/4Bmri_0fltr.img ./data.dir/WT4Bmri_0fltr.img

This will save the 2D Discrete Wavelet transform of "4Bmri_0fltr.img" in the file "WT4Bmri_0fltr.img". Both files will contain 2D floating point (4 Bytes) images of size 256*256. Daubechies 20 coefficient filter bank is employed to find the DWT of the data. Also look at the batch file "batch_DWT_IWT".

- **wave_space_varThresh**

  Purpose:
  This WAIR command is used to obtain the n-Dimensional Inverse Discrete Wavelet Transform under various thresholding schemes.

  (Interactive) Usage:
  `wave_space_varThresh`

  Examples:
  `./wave_space_varThresh`

  This will start an interactive routine. Input the correct information following the prompt and hit after each entry.

- **CGC_WTC**

Purpose:
This WAIR command is used to obtain the Cluster Group Classification (CGC) and the induced warp ranking having a family of warps applied to a group of data sets.

The CGC Classification of family of image-registration techniques is determined based on the warp performance on a group of data sets. You need to have executed "WT_IWT" and "wave_space_varThresh" first before you call "CGC_WTC", if wavelet-space analysis is to be done!

Usage:
CGC_WTC *Nomadic x_1_Dim_Size [[x_k_Dim_Size]] Num_Data_Sets Num_Warps [[Data_1_Warp_k]] [[Data_2_Warp_k]] ... [[Data_n_Warp_k]] results_Data_CGC.text*

where the following definitions are used:

Num_Dim
the number of dimensions of the data set

x_1_Dim_Size
the size of the first (fastest varying index) dimension of the data. Remember, all dimension sizes need to be **powers of 2**

x_k_Dim_Size
the size of the k-th dimension of the data

Num_Data_Sets
the number of data sets in the **group**

Num_Warps
the number of **warps** involved in the study

Data_1_Warp_k
data **"l"** resliced using warp **"k"**, 1<=l<=Num_Data, 1<=k<=Num_Warps

results_Data_CGC.text
file that will contain the output of the CGC analysis

Examples:
3 ./WTmri1_Wp1.img ./WTmri1_WP2.img ./WTmri1_Wp3.img ./WTmri2_Wp1.img ./WTmri2_Wp2.img ./WTmri2_Wp3.img ./WTmri3_Wp1.img ./WTmri3_Wp2.img ./WTmri3_Wp3.img ./WTmri4_Wp1.img ./WTmri4_Wp2.img ./WTmri4_Wp3.img ./WTmri5_Wp1.img ./WTmri5_Wp2.img ./WTmri5_Wp3.img ./resultsMRI_CGC.text

This command will quantify, using CGC, the performance of 3 warps (Wp1, Wp2, Wp3) applied to a group of 5 (thresholded) stereotactic (3D) WT's of the images of size 256*256*128 (mri1, ..., mri5). The results of the CGC analysis will be saved in the output file "resultsMRI_CGC.text", in

the current directory. Also look at the batch file "batch_CGC". **Note** that the above example represents a **single** command-line.

- **SGC_WTC**

  Purpose:
  This WAIR command is used to obtain the Spread Group Classification (SGC) and the induced warp ranking having a family of warps applied to a group of data sets. For example, if we need to quantify warp performance, based on **m** subjects, each scanned under **k** different activation paradigms, and we wish to see large spread between the groups of **m** images, for the various functional activation stimuli, then we use the SGC analysis separately for every warp.

  The SGC Classification of family of image-registration techniques is determined based on the warp performance on a group of data sets. You need to have executed "WT_IWT" and "wave_space_varThresh" first before you call "SGC_CWT", if wavelet-space analysis is to be done!

  Usage:
  SGC_CWT *Num_Data TotNumParadigms TotNumBaseline Num_Dims x_1_Dim_Size [[x_k_Dim_Size]] [[WT_Data_1_Par_k]] [[WT_Data_2_Par_k]] ... [[WT_Data_m_Par_k]] results_WTData_SGC_whichWarp.text*

  where the following definitions are used:

  Num_Data
  the number of the data sets (**m**)

  TotNumParadigms
  the total number of activation paradigms, applied to all subjects (**k**)

  TotNumBaseline
  the total number of "base-line" activations. If no paradigm is being repeated, put 1

  Num_Dims
  the total number of dimensions of your data

  x_1_Dim_Size
  the size of the first (fastest varying index) dimension of the data. Remember, all dimension sizes need to be **powers of 2**

  x_k_Dim_Size
  the size of the k-th dimension of the data

  WT_Data_i_Par_j
  The (thresholded) wavelet transform of the data **"i"** under the activation paradigm **"j"**, 1<=i<=Num_Data, 1<=j<=TotNumParadigms

results_WTData_SGC_whichWarp.text
file that will contain the output of the SGC analysis for the particular image registration

Examples:
./SGC_CWT 3 2 1 3 128 128 64 ./WTpet1_freq1_warp1_12812864.img ./WTpet1_freq2_warp1_12812864.img ./WTpet2_freq1_warp1_12812864.img ./WTpet2_freq2_warp1_12812864.img ./WTpet3_freq1_warp1_12812864.img ./WTpet3_freq2_warp1_12812864.img ./results_SGC_CWT_Warp1.text

This command will quantify, using SGC, the performance of 1 image registration (warping) based on 3 subjects each scanned twice, under two different activation paradigms. The data are 3D of matrix size 128x128y64z. The results of the SGC analysis will be saved in the output file "results_SGC_CWT_Warp1.text", in the current directory. Also look at the batch file "batch_SGC_CWT". **Note** that the above example represents a **single** command-line.

- **TriangleWTC**

Purpose:
This WAIR command is used to obtain the compressed wavelet-space triangle warp classification for single or multiple data. The induced warp ranking reflects various GS's (gold-standards) along the line connecting the data to the target in compressed wavelet space. You need to have executed **WT_IWT** and **wave_space_varThresh** first before you call **triangleWTC**.

Usage:
triangleWTC *Num_Dim x_1_Dim_Size [[x_k_Dim_Size]] Num_Data_Sets Num_Warps Target Data_1 [[Data_1_Warp_k]] Data_2 [[Data_2_Warp_k]] ... Data_n [[Data_n_Warp_k]] results_Data_Tri.text*

where the following definitions are used:

Num_Dim
the number of dimensions of the data set

x_1_Dim_Size
the size of the first (fastest varying index) dimension of the data. Remember, all dimension sizes need to be **powers of 2**

x_k_Dim_Size
the size of the k-th dimension of the data

Num_Data_Sets
the number of data sets in the **group**

Num_Warps

the number of **warps** involved in the study

Target
the name of the data-file used as a **target** of the warps

Data_l
the name of the **l**-th original data set (prior to warping) 1<=**l**<=n

Data_1_Warp_k
data **"l"** resliced using warp **"k"**, 1<=l<=Num_Data, 1<=k<=Num_Warps

results_Data_Tri.text
file that will contain the output of the wavelet space **triangle** warp ranking

Examples:
./triangleWTC 4 128 128 256 512 5 3 ./WTtarget.img ./WTpet1.img ./WTpet1_Wp1.img ./WTpet1_Wp2.img ./WTpet1_Wp3.img ./WTpet2.img ./WTpet2_Wp1.img ./WTpet2_Wp2.img ./WTpet2_Wp3.img ./WTpet3.img ./WTpet3_Wp1.img ./WTpet3_Wp2.img ./WTpet3_Wp3.img ./WTpet4.img ./WTpet4_Wp1.img ./WTpet4_Wp2.img ./WTpet4_Wp3.img ./WTpet5.img ./WTpet5_Wp1.img ./WTpet5_Wp2.img ./WTpet5_Wp3.img ./resultsPET_Tri.text

This command will quantify, using the "triangle" wavelet-space approach, the performance of 3 warps (Wp1, Wp2, Wp3) applied to a group of 5 4D images of size 128*128*256*512 (pet1, ..., pet5). The warp rankings and results of this analysis will be saved in the output file "resultsPET_Tri.text", in the current directory. Also look at the batch file "batch_triangle". **Note** that the above example represents a **single** command-line.

AUXILLIARY ROUTINES

- **MultiDim_IWT_varThresh**

Purpose:
The only interactive routine in WAIR 2.0 is "MultiDim_IWT_varThresh". This WAIR command is used to obtain the multi-dimensional inverse wavelet transform (IWT) of a signal using one of 3 possible wavelet **thresholding** schemes, Uniform (at various levels), Donoho-Johnstone (DJ) and Dinov-Sumners (DS) frequency adaptive approach (see the technical notes).

This procedure is useful for comparing and visualizing the differences between the 3 wavelet shrinkage techniques in the cases of lower dimensional (1D and 2D) images. You need to have executed "MultiDim_WT" first before you call **MultiDim_IWT_varThresh**.

Usage:
MultiDim_IWT_varThresh

This routines requires no arguments or input parameters. After calling **MultiDim_IWT_varThresh** follow the interactive instructions - you will be asked to input certain information.

Examples:

```
./MultiDim_IWT_varThresh
```

This interactive routine will determine the IWT of a signal after applying a wavelet shrinkage to the wavelet coefficients.

WAIR GRAPHICAL USER INTERFACE

1. Change the working directory to WAIR2.0.dir/WAIR_JAVA_GUI.dir
2. TYPE make to install the main WAIR_JAVA_GUI
3. TYPE WAIR_GUI_start to START the main WAIR_JAVA_GUI
   This is a JAVA-based GUI which should work on all platforms. Try to use this GUI and not the OLD Vk-based one, described below. Of course, we have configured the GUI to make UNIX executable calls, so for people working on PC/Mac's you may have to open the source and re-configure the GUI commands for your system.

Another (OLDER) GUI is implemented in a VisualKit (Vk Library, based on Motif) graphical user interface to WAIR which simplifies the usage of the software. The GUI is easy to install and very handy for running large scale brain data analyses or any number of volumes in a repetitive fashion.

1. Change the working directory to WAIR2.0.dir/GUI_WAIR_2_0.dir
2. Run csh README to install the main WAIR_GUI
3. Similarly, in the subdirectories: GUI_1_warps.dir, GUI_2_WT.dir, GUI_3_Fourier.dir, GUI_4_wave_T_thresh.dir, GUI_5_warp_rank.dir and GUI_6_viz.dir, type "csh README" to install the local parts of the GUI.

TECHNICAL NOTES

WAVELET BASES UTILIZED IN THE WAIR PACKAGE

WAIR 2.0 utilizes the following 17 types of wavelet bases:

The delimeter "whichWaveFilter" is user specifiable from within the set of:

- **whichWaveFilter**==1 ====> Spline: parameters=(3,7);
- **whichWaveFilter**==2 ====> Spline (3,3);
- **whichWaveFilter**==3 ====> Spline (2,4);
- **whichWaveFilter**==4 ====> Spline (2,2);
- **whichWaveFilter**==5 ====> Pseudo-Coiflet 4_4;
- **whichWaveFilter**==6 ====> Haar wavelet;
- **whichWaveFilter**==7 ====> Daubechies 20 coefficient basis; DEFAULT
- **whichWaveFilter**==8 ====> Daubechies 12;
- **whichWaveFilter**==9 ====> Daubechies 10;
- **whichWaveFilter**==10 ====> Daubechies 8;
- **whichWaveFilter**==11 ====> Daubechies 6;

- **whichWaveFilter**==12 ====> Daubechies 4;
- **whichWaveFilter**==13 ====> Coiflet 6 parameter;
- **whichWaveFilter**==14 ====> Coiflet 4;
- **whichWaveFilter**==15 ====> Coiflet 2;
- **whichWaveFilter**==16 ====> Burt-Adelson;
- **whichWaveFilter**==17 ====> Battle-Lemarie.

WARRANTY AND LIABILITY

ANY VERSION OF THE WAIR PACKAGE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. IN NO EVENT SHALL THE DEVELOPERS OF THIS SOFTWARE, OR THE INSTITUTIONS THEY REPRESENT, BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM THE LOSS OF USE, DATA, OR PROFITS, OR FROM PATIENT MISDIAGNOSIS OR MISTREATMENT, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE SOFTWARE IS BORNE BY YOU. SHOULD ANY PART OF THE WAIR PACKAGE PROVE DEFECTIVE YOU AND NOT THE WAIR OWNERS AND DEVELOPERS ASSUME THE ENTIRE COST OF ANY SERVICE, REPAIR OR LOSS. YOU OR THE PERSON USING WAIR ON YOUR BEHALF OR FOR YOUR BENEFIT IS RESPONSIBLE TO DETERMINE THAT THE SOFTWARE MEETS ALL OF YOUR REQUIREMENTS AND IS SAVE, RELIABLE AND HARMLESS FOR YOUR NEEDS.

ACKNOWLEDGEMENTS

The owners and developers of the WAIR software are grateful to Dr. David Rottenberg, Dr. Stephen Strother, Kirt Schaper and Jon Anderson at PET Imaging Service at the VAMC, Minneapolis, MN, for providing support, encouragement, data, constructive recommendations and feedback in regard to the development and testing of this package.

The group led by Dr. Arthur Toga, including Paul Thompson, Roger Woods, Michael Mega, Colin Holmes and others, at the Laboratory of Neuro Imaging at UCLA has been actively involved in the implementation and validation of the WAIR routines.

Funding and support for the development and documentation of the WAIR software came from NIH, BWF, FSU and other organizations and grants.