Hardware Solutions for the Optimization of AFNI Processing

Trokon Johnson, Mentor: Dr. Peter Hawrylak

Department of Electrical Engineering, The University of Tulsa , Rayzor Hall 800 S. Tucker Dr., Tulsa, OK 74104

Introduction

Recently, Functional Magnetic Resonance Imaging (FMRI) technology has become a very prevalent procedure for neuroinformatics research. Whereas traditional Magnetic Resonance Imaging (MRI) technology simply allowed for production of a structural, anatomical data of the brain, an FMRI scan can read the level of blood oxygenation in different areas of the brain. Researchers can use this data to better understand how the human mind works; by providing a stimuli to a test subject and observing which areas of the brain respond with an increased level of blood flow, one may discover:

- which neural regions are associated with differing mental tasks
- which neural regions coordinate on tasks
- the differences between two subjects neural processes

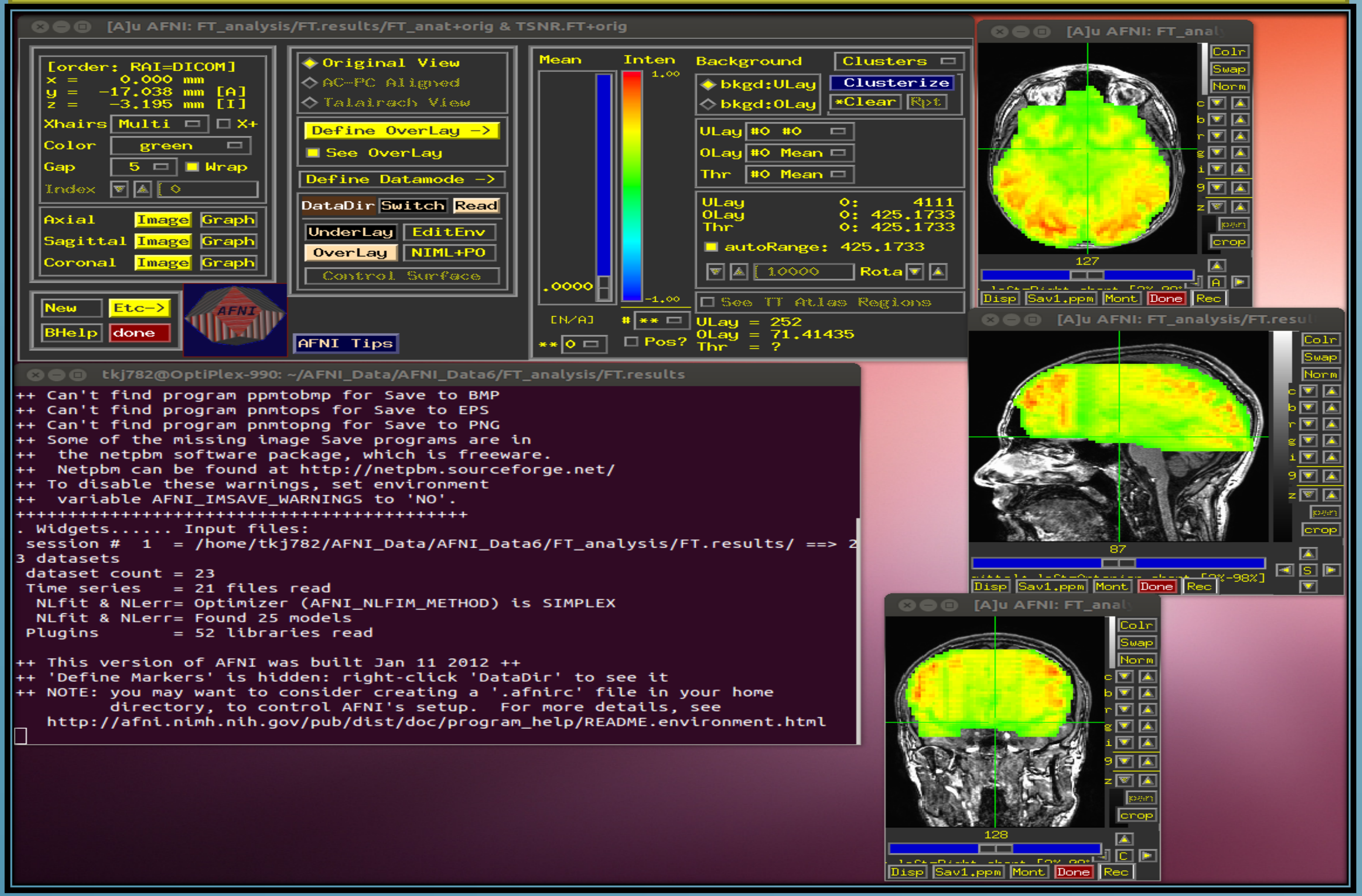
Processing this data, however, can be very time and resource intensive, depending on the technique used. This research focused on custom hardware solutions in order to In order to apply the principles of hardware solutions on fMRI technology, we selected several platforms to integrate together, with the hopes of optimizing the process

- **A**nalysis of **F**unctional **N**euro**I**mages **(AFNI)**
  - Visualization and Processing Environment for MRI data
- **F**ield **P**rogrammable **G**ate **A**rray **(FPGA)**
  - Platform programmable at the hardware level
- **V**ery high speed integrated circuit **H**ardware **D**escription **L**anguage **(VHDL)**
  - Standard method for describing and synthesizing logic circuits

This summer, the project focused on:

- Studying the AFNI Codebase
  - Open source environment developed by NIMH researchers
  - Neuroimaging viewing and processing software
  - Seeking time or memory intensive routines
- Analyzing the structure of the Field Programmable Gate Array
- Transferring programs from conventional to FPGA architecture
  - Translating code to the hardware description language VHDL

AFNI Workspace



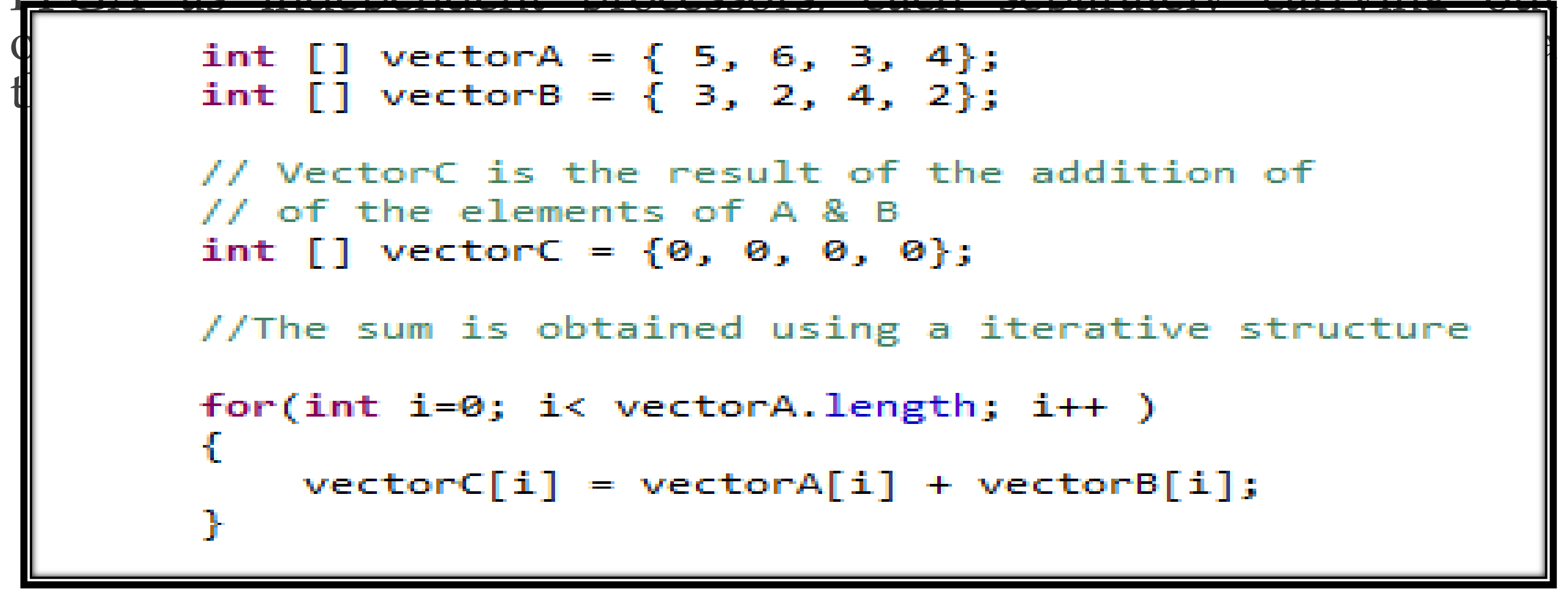
```
tkj782@OptiPlex-990: ~/AFNI_Data/AFNI_Data6/FT_analysis/FT.results
++ Can't find program ppmtobmp for Save to BMP
++ Can't find program pnmtops for Save to EPS
++ Can't find program pnmtopng for Save to PNG
++ Some of the missing image Save programs are in
++  the netpbm software package, which is freeware.
++  Netpbm can be found at http://netpbm.sourceforge.net/
++ To disable these warnings, set environment
++  variable AFNI_IMSAVE_WARNINGS to 'NO'.
++++++++++++++++++++++++++++++++++++++++++++
. Widgets...... Input files:
 session #  1  = /home/tkj782/AFNI_Data/AFNI_Data6/FT_analysis/FT.results/ ==> 2
3 datasets
 dataset count = 23
 Time series   = 21 files read
  NLfit & NLerr= Optimizer (AFNI_NLFIM_METHOD) is SIMPLEX
  NLfit & NLerr= Found 25 models
 Plugins       = 52 libraries read

++ This version of AFNI was built Jan 11 2012 ++
++ 'Define Markers' is hidden: right-click 'DataDir' to see it
++ NOTE: you may want to consider creating a '.afnirc' file in your home
         directory, to control AFNI's setup.  For more details, see
   http://afni.nimh.nih.gov/pub/dist/doc/program_help/README.environment.html
```

Premise

Within AFNI, there were a few commands in particular that were looked at. Information gained from correspondences with researchers at Tulsa's Laureate Institute for Brain Research (LIBR) informed us that two of the most resource heavy commands were 3dAutoTcorrelate, and 3dTCorrmap. These two commands carried out correlation calculations on all of the voxels in an fMRI dataset, a voxel, or volumetric pixel, being the smallest unit of data captured by the MRI device. Researchers could use this data in order to observe correlation between certain regions of the brain due to certain stimuli or tasks, and draw conclusions about the interdependence of these areas. Unfortunately, as there may be thousands of voxels in a set, these operations can be, as previously mentioned, extremely taxing on time and resources, especially if run on a study containing many subjects. As these processes rely on a number of the same calculations on a given set, however, these loops could be unrolled using the FPGA architecture.

Methods

A major principle that the research was based upon was parallel computer architecture. The goal was to use the many cores of the FPGA as independent processors, each separately carrying out



```
int [] vectorA = { 5, 6, 3, 4};
int [] vectorB = { 3, 2, 4, 2};

// VectorC is the result of the addition of
// of the elements of A & B
int [] vectorC = {0, 0, 0, 0};

//The sum is obtained using a iterative structure

for(int i=0; i< vectorA.length; i++ )
{
    vectorC[i] = vectorA[i] + vectorB[i];
}
```

The above code computes the vector sum of two vectors; the top section of the code initializes the vectors, and the bottom is a for loop that computes the vector sum. Though some assumptions in the code have been made, the basics of the for loop is that it iterates through a certain set of commands a number of times. Here, the loop sums the first index of two distinct arrays, and places the result in the same index of a new array.

Parallelism can be used to unroll this process, drastically decreasing the time required to compute similar but large scale operations. Traditional programming is carried out sequentially, one command at a time. Therefore, with the 4 element vectors seen above, command within the for loop would be executed 4 times. If this code were optimized for multiple cores, however, each core could run one instance of the command, theoretically reducing run time by a factor comparable to 4, though not exact, due to overhead resources. This principle could be applied to AFNI processing in order to accelerate time and memory intensive commands.

Pearson Correlation Coefficient

$$\rho_{X,Y} = \mathrm{corr}(X,Y) = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y},$$

Voxels	V1	V2	V3	V4
V5	C(V1, V5)	C(V2,V5)	C(V3,V5)	C(V4,V5)
V6	C(V1, V6)	C(V2,V6)	C(V3,V6)	C(V4,V6)
V7	C(V1,V7)	C(V2,V7)	C(V3,V7)	C(V4, V7)
V8	C(V1,V8)	C(V2,V8)	C(V3,V8)	C(V4,V8)

Conclusion & Future Directions

Through background research, the ideas of parallelization and hardware acceleration were validated A major problem for this work was the memory requirements of the correlation commands; the machines provided were not able to complete them. Coordination with the LIBR institute has alleviated this problem to some degree, but a more permanent solution is necessary.

The next steps in the program are to:

- Convert programs to FPGA Architecture
  - Investigate how Nexys 3 and Spartan 6e will implement and optimize functions
  - Translate code into VHDL
- Investigate protocols for data transfer to & from the FPGA
  - Ethernet
  - USB
  - On board Storage

In addition, acquisition of real data sets is hoped for, as opposed to sample data sets from the AFNI website. Alternatively, confirmation of the validity of the sample data sets could be used, which would allow for a common baseline to test the functions on the new hardware.

Benchmark tests of certain functions would be in order. The AFNI website contains some of this data already, though only for some functions.

Acknowledgements

1. Tulsa Undergraduate Research Challenge & The University of Tulsa
2. OK Louis Stokes Alliance for Minority Participation
3. Laureate Institute for Brain Research
4. TU Neuroinformatics Working Group
5. University of Tulsa Information Security