**Chelsea Lewis**

**Indexing and Retrieval of XBOX Kinect Motion Data**

**Overview:**

The purpose of this project is to index and retrieve the motion data retrieved from the Kinect. The final edition of this project uses a rule-based algorithm to determine an appropriate point value for scoring.

**Computer Specifications**

This project was built and run on a Windows 7 64-bit operating system with 4.00 GB of memory. Further specs: Intel® Core™ i5-2430M CPU @ 2.40 GHz.

**Installation Instructions:**

The files for this project were written, built, and compiled using NetBeans IDE 6.9.1. This NetBeans installation is using Cygwin compiler in order to compile C++ code. In order to run the .exe included in this project folder, the user must have a Cygwin compiler. If using a terminal, the user can compile all source code using g++, with main.cpp being the executable class.

**Operating Instructions:**

When operating the .exe file, the application will begin to run, first prompting the user to locate the folder in which all patient data is located. Within this folder, each patient has their own individual folder which contains all exercises performed and recorded from the Kinect.

Once the user selects this folder, the application will compute the point value each patient receives for each exercise performed. This point value is based on the NIH Stroke Scale and will be displayed to the user during the execution of the program.

If the code needs to be updated or revised, the best place to start is with CandidateSearch.cpp. This file is where all of the important classes are called to compute the scoring values. Matching.cpp is where the scoring is actually computed.

**Copyright and Licensing Information:**

Some open source code was used to assist with the development of this project. While it is not used in the final project edition, code for a Dynamic Time Warping algorithm was used in a previous edition of this project. This DTW algorithm borrowed from [**http://www.koders.com/java/fid647E612AEB89260F5C952B17879C93F00A21CD80.aspx?s=AddOpts**](http://www.koders.com/java/fid647E612AEB89260F5C952B17879C93F00A21CD80.aspx?s=AddOpts)

ignores length discrepancies between two motions and computes the smallest distance between the two motions.