Senior Project CIS 4911-U01

Multi-Touch and Mid-Air Framework

Multi-Touch and Gyroscope Visualizer

User Manual

Team Members:

Steven Ignetti (SUMMER 2015)

Alfredo Zellek (SUMMER 2015)

Richard  A. Lopez (SPRING 2015)

Mentor:

Francisco Ortega

Professor:

Masoud Sadjadi

Overview

Input devices have advanced at an outstanding rate in the last few decades. The introduction of the mouse was one of the factors that brought the personal computer out of the technical user niche and into the regular consumer. Now touch input is in almost every cellphone and readily available to the general public. Newer input forms have been developed since such as multi touch screen, 3D mouse, midair tracking, and others. Up until now developers have had to code their respective games and applications for the individual input devices which make it a time consuming task and it limits the utilization of more than one input device in use at a time.

The Touch Air Motion Framework is trying to create a more encompassing input device library such that developers can simply choose their input devices and code their apps without having to deal with the individuality of each device. This document pertains to the visualization interface for the TAM framework.

MANUAL

Requirements

To utilize version 2.0 of the TAM framework you need the following:

* Touch screen computer
* Windows 7 or greater
* 2 GB RAM minimum
* 512 MB video card minimum
* Microsoft Visual Studio 2013
* Qt version 5.4.1
* Qt plugin version 1.2.4 for Visual Studio 2013 version 5.4

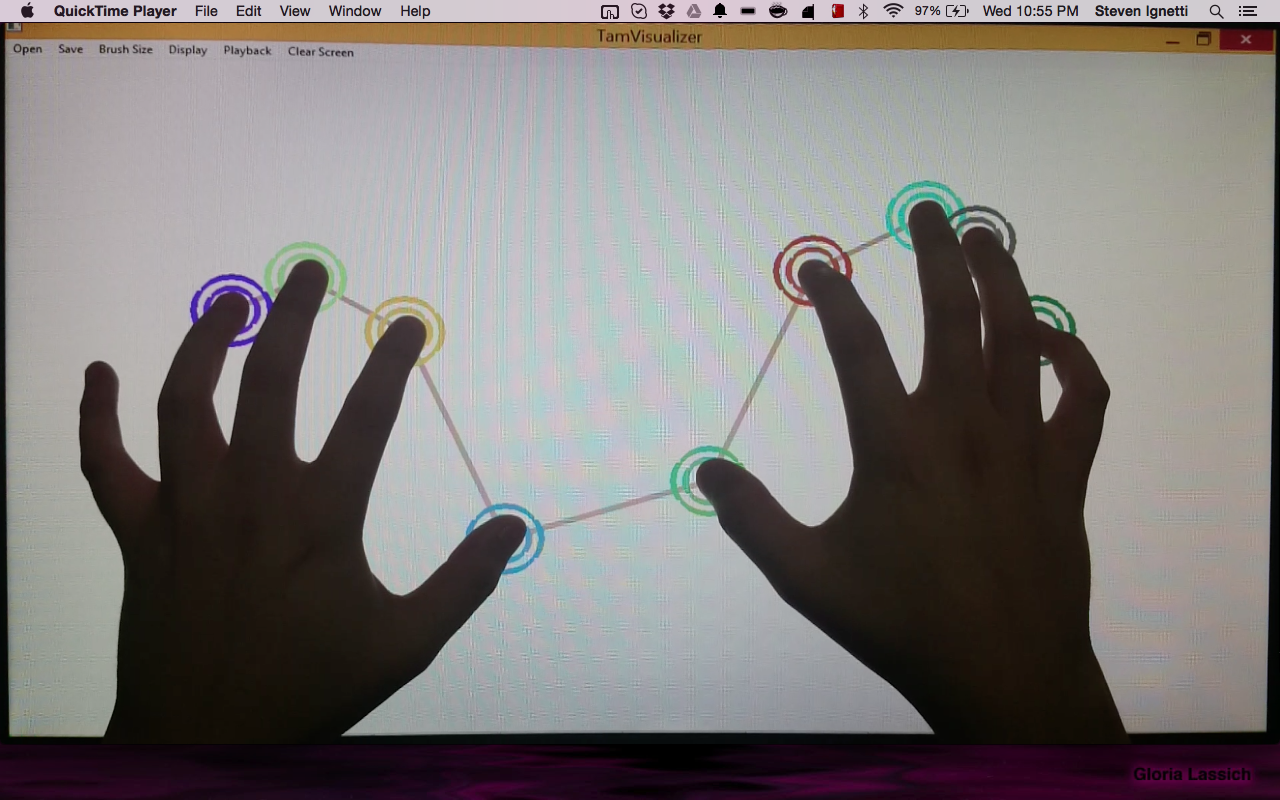
Installation

For instructions on the installation of the TAM framework please refer to the “Installation” manual located in the Code folder for this version.

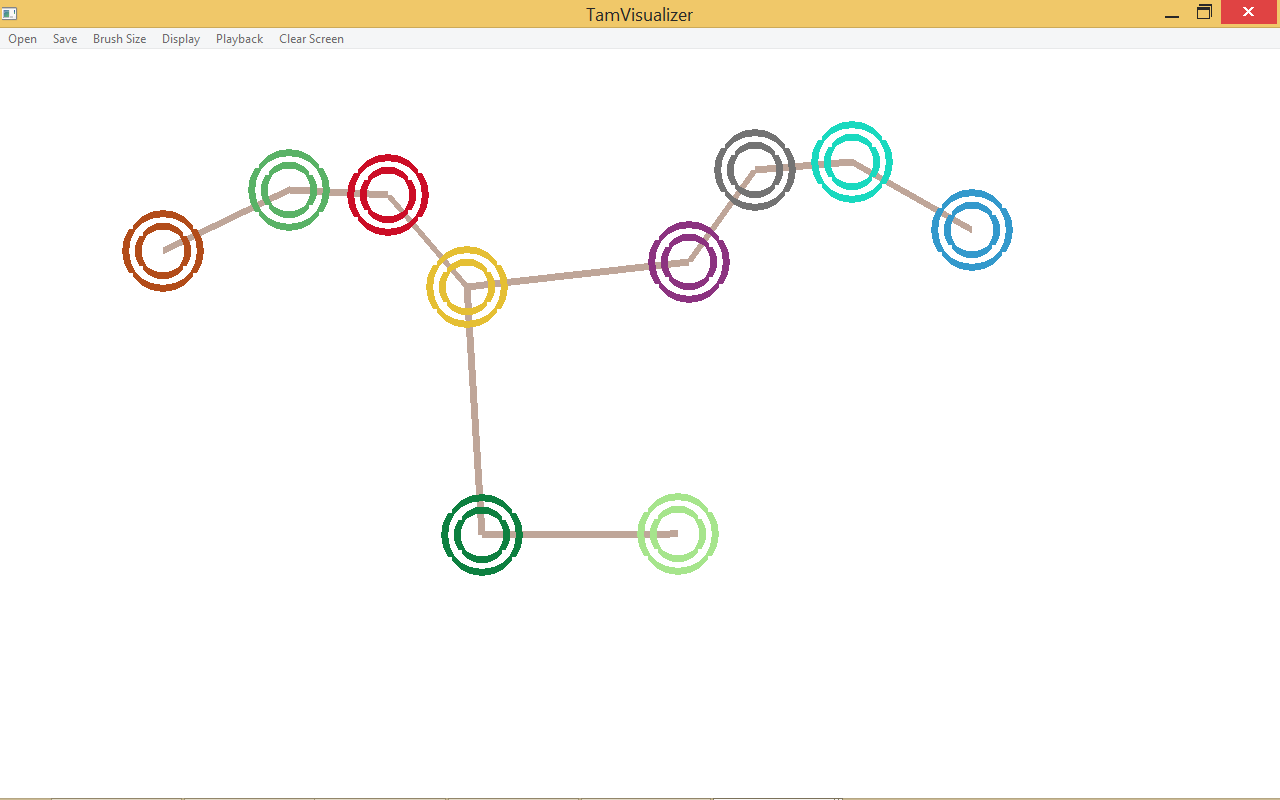
**Multi-Touch Visualizer**

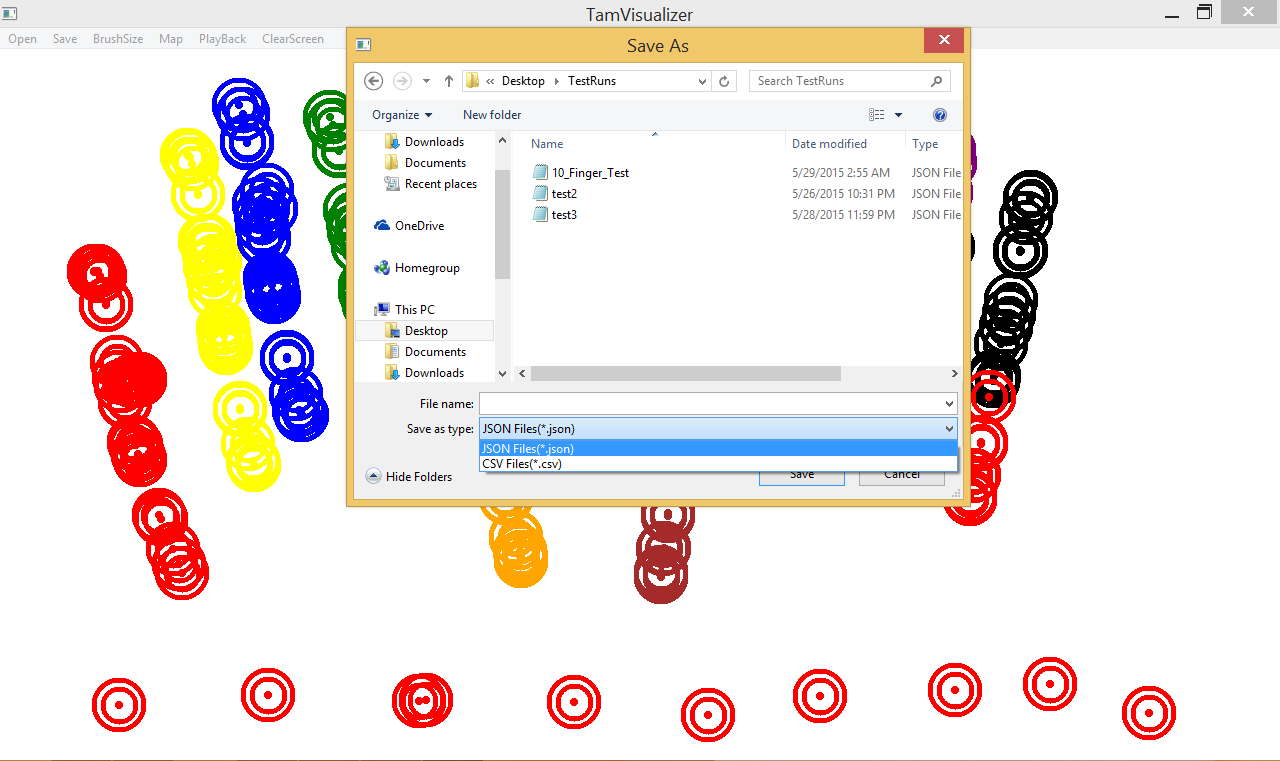
Features

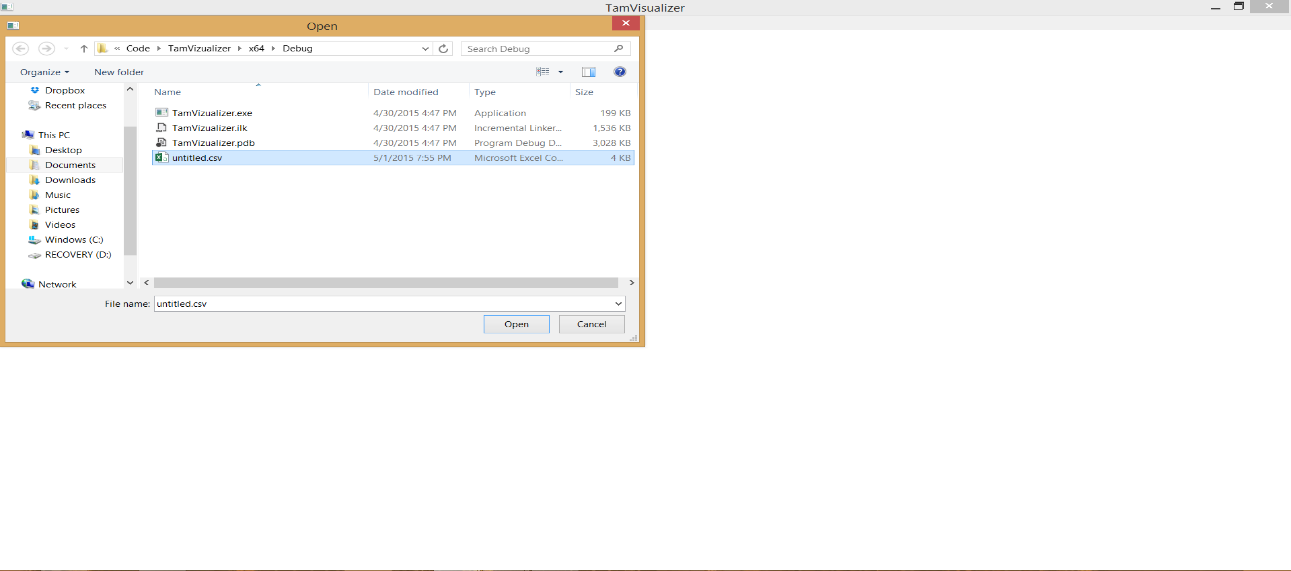
* **Tracking:** An existing feature from version 1.0, with many improved features in the latest version. Being a multi touchscreen visualizer, one should expect a visual response to touch screen events. In version 1.0, the application would draw the results of where your fingers touch the screen. In the latest version, the application now tracks the position of your fingers on the touchscreen device with visual circles highlighting the contact areas in the monitor. The application must be running Windows 7 / 8 or greater, and is tested to handle up to 10 fingers at one time.



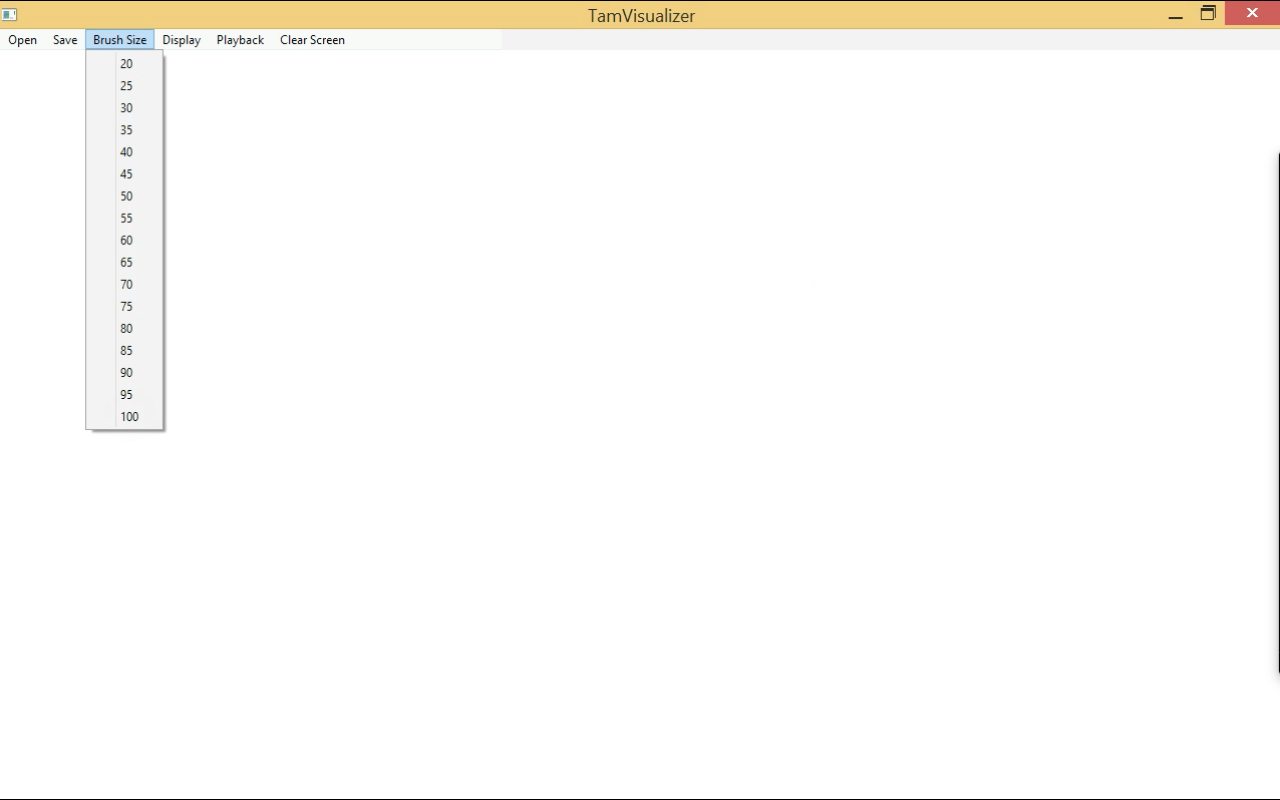
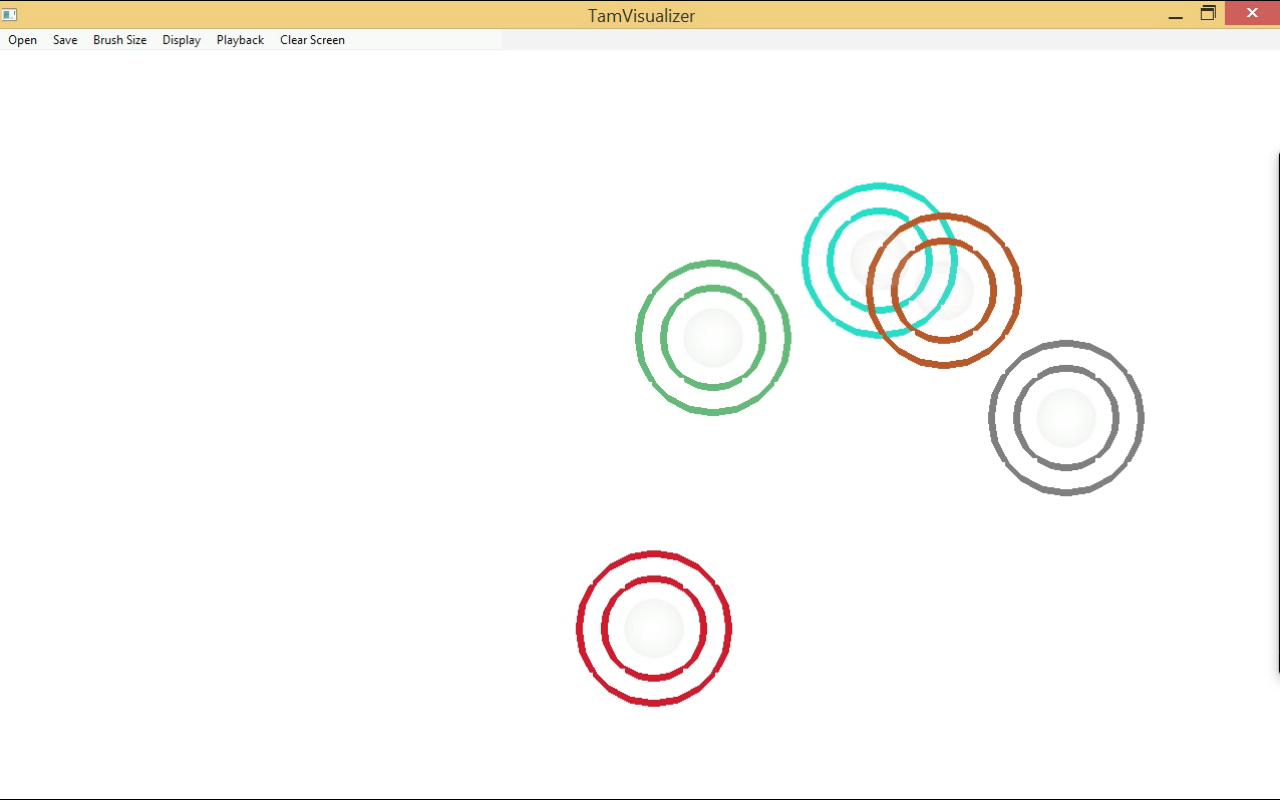
* **Input Recording:** An existing feature from version 1.0. As you touch the screen and draw through it, all of the individual touch points are being saved to a list. This is a temporary file maintained only as long as the application is running or you choose to replay your motions.
* **Playback:** An existing feature from version 1.0, with some improved features in the latest version. To replay your drawing all you need to do is press the “Playback” button, this will start a replay of all your past touch events. At the end, your past touch-event recording is cleared to allow you to create new touch points. What is improved is that the replay happens in real-time, meaning that any timed reaction on the touch point recording will reflect on the replay as well.

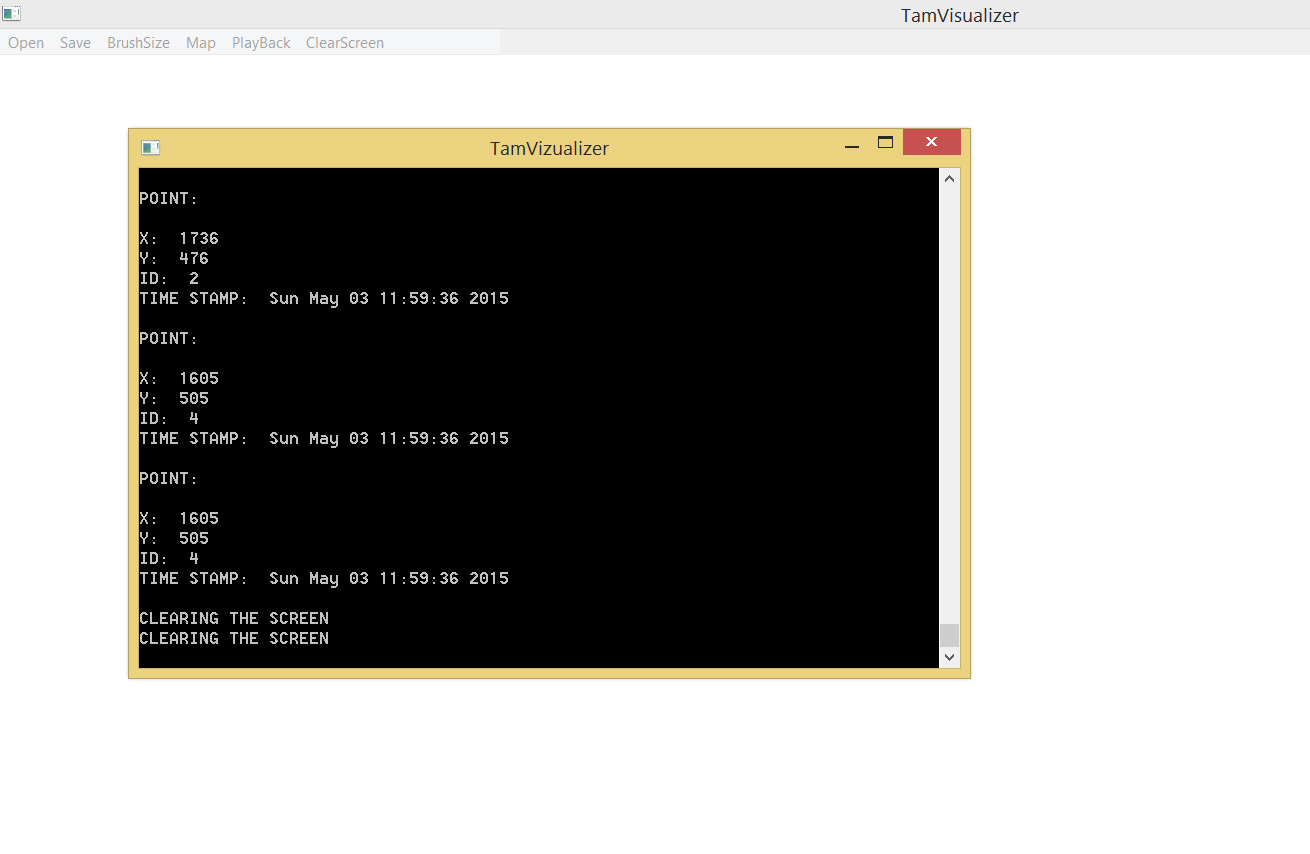
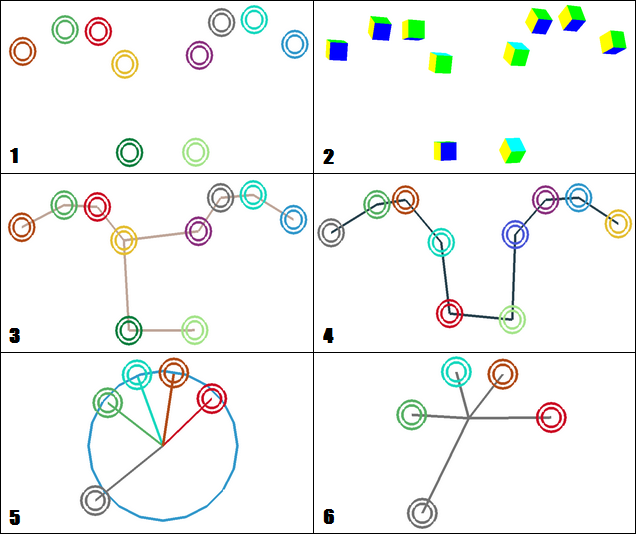
****

* **Save Touch Points:** An existing feature from version 1.0, with some improved features in the latest version. If you need a more permanent list of all your touch points, simply press the “Save” button after you finished drawing. This will prompt you with a save box, search the location you what the file to be saved to and type the name you prefer for the file, finally press “Save” and your data will be saved permanently. What is improved is that you can now save it as a JSON file as well as a CSV file used in the previous version.
* **Playback from a File:** An existing feature from version 1.0, with some improved features in the latest version. There is no use to having your data in a file if you can’t see the results. Simply press the “Open” button, and an Open pop-up will prompt you to search for the location of the file, once you find it all you must do is double click on the file or single click and press “Open”. A replay of your drawing will start immediately. As stated previously, JSON is now a file-type that can be opened so long as it was created by this application.

****

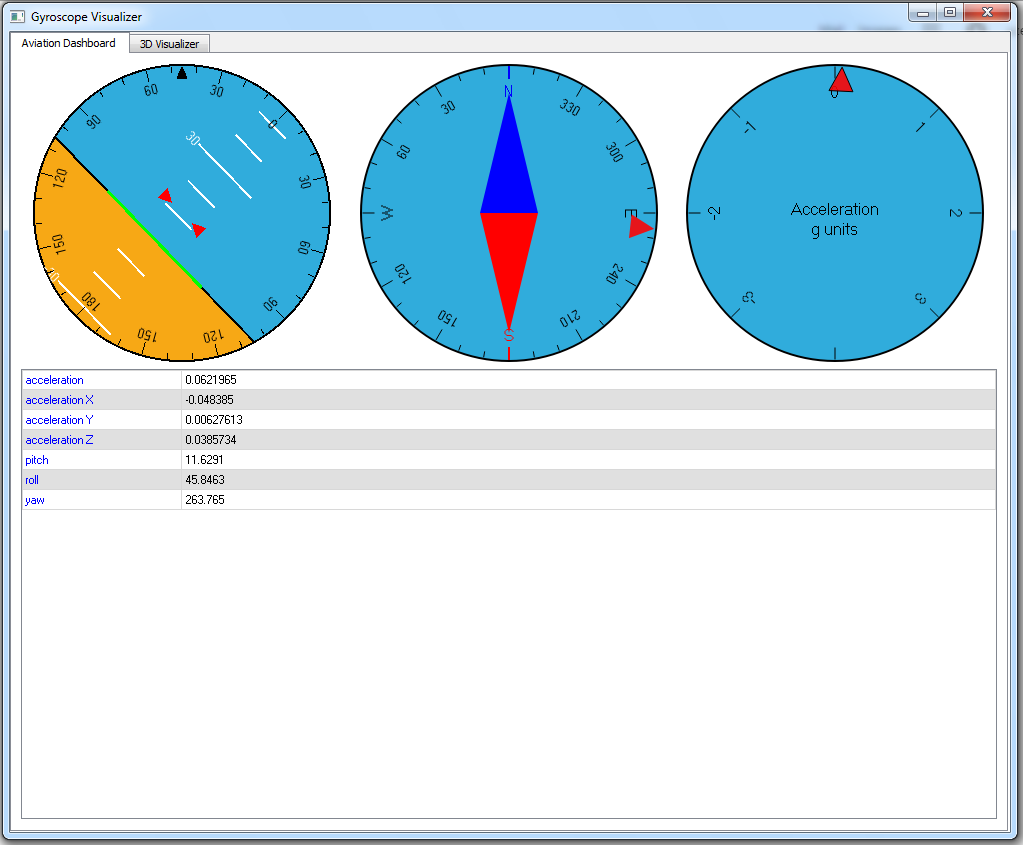
* **Resize Brush:** An existing feature from version 1.0, with modifications in the latest version. Want to change the size of the finger tracking displayed on the application? All you need to do is click the “Brush Size” button; this will open the menu of available brush sizes, select one and start drawing with your new brush size.

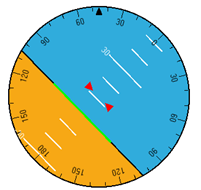


* **Clear Screen:** An existing feature from version 1.0, with modifications in the latest version. If the recording is getting to be too long, simply click on the “Clear Screen” button. This will clear your screen as well as delete the list of touch points you have created previously.
* **Debug Console:** An existing feature from version 1.0. If you need to know the specific points on the screen, the id of the fingers, the time of the action, or any other output that the application creates all you need to do is look at the Debug Monitor. This is the screen that opens along with the application. This will receive any output sent by the system to the standard output.
* **Debug Window:** A new feature in the latest version. This Debug Window behaves the same was as the Debug Window, but is purposed solely for the application and can be used across other platforms. This window will eventually be improved in later versions to include more debugging features.
* **Change Display:** A new feature in the latest version. The application can now use your finger positions for other purposes now, including algorithmic calculations and eventually gesture recognition. This is shown by the “Display” drop-down menu, listing a number of ways the fingers can be displayed. Some of the displays include Circular Connections (#5 below) and Shortest Mapping (#4 below).
* **OpenGL Rendering:**  A new feature in the latest version. The application now renders all the shapes on screen using OpenGL. This allows for not only speed in graphics, but also allows for 3D visualization. This feature will let future versions make 2D visualizations stand out, such as with the OpenGL Cubes Display setting (#2 in the above figure), or used in 3D-visualizations of three-dimensional input devices, an example of which will be shown in the gyroscope visualizer portion of the manual.

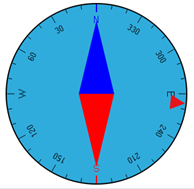
**Gyroscope Visualizer**

Features

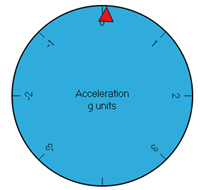
* **Aviation Dashboard** **:** This feature allows the user to visualizer the gyroscope’s rotations in 2 Dimensions. It includes an Attitude-indicator, a Compass, and an Accelerometer gauge. 
  + **Attitude-Indicator:** Visualizes the Pitch and Roll of the gyroscope. This gives you an idea of the orientation the gyroscope is in.

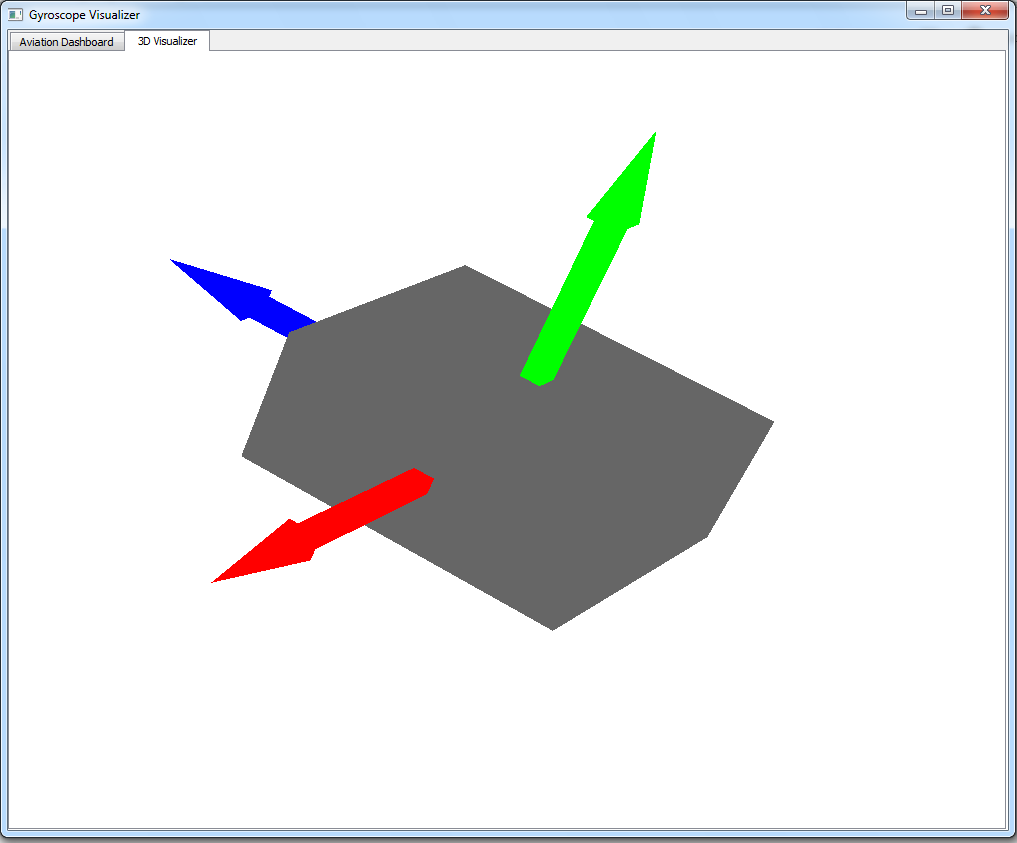
****

* + **Compass:** Acts like a regular compass, and yields information about the heading of the compass. Degrees are marked in 30 degree intervals.

****

* + **Accelerometer:** Measure acceleration in g (9.8 m/s^2) units. Allows the user to see the positive and negative acceleration of the gyroscope. Note: the gyroscope is very sensitive, so it will also detect minor differences in acceleration.

****

* + **Data display:** The data display is a text base display of the data that is visualized by the Aviation-style dashboard. It includes Pitch, Roll, Yaw, Acceleration, and Acceleration in each of the coordinate directions.****
* **3D Visualizer:** This is the 3D visualizer for the gyroscope. It maps the quaternion rotation data given by the gyroscope to an on-screen representation. Red arrow is the x-axis, green arrow is the y-axis, and blue arrow is the z-axis. For the Yei gyroscope, the green arrow denotes the top side of the device. This is where one will find the blue LED and the buttons. ****