AnthroTronix

acceleglove

Control in Hand

User's Guide

www.AcceleGlove.com

Version 1.1.1 January 2011

TABLE OF CONTENTS

1. INTRODUCTION

The AcceleGlove™ is a complete development solution for adding hand-motion input or control to a range of applications in the fields of Robotics, Medical Rehabilitation and Telemedicine, Training, Gaming, and Virtual Reality/Simulation environments.

The AcceleGlove product kit includes:

- An instrumented glove that captures and outputs hand and finger position and motion data.
- A Java software library that allows developers to "hand-enable" their own applications.
- An application with a graphical user interface – Visualizer -- for training, runtime, and diagnostics.

The AcceleGlove integrates sensors, called accelerometers, within a lightweight, flexible glove. The glove transmits hand motion and orientation data over a USB cable, or using the optional AcceleGlove Wireless Module™, over wireless Bluetooth®.

The AcceleGlove SDK (Software Development Kit) allows a developer to use direct glove output or to recognize static gestures from either a pre-trained library or a user-trained library.

2. CONTENTS

The contents of your AcceleGlove package include the following:

1. An AcceleGlove with USB cable.
2. A CD containing SDK files, AcceleGlove drivers, and a User's Guide.
3. Optional AcceleGlove Wireless Module, Wireless Module cable, and Wireless Module wall charging unit.

3. QUICK START

3.1 *Software Installation*

When you open the AcceleGlove Software Development Kit CD, you will see four folders:

1. Drivers
2. SDK
3. User's Guide
4. Visualizer

3.1.1 Installing AcceleGlove Drivers

A software driver is needed for the AcceleGlove to communicate with the computer.

Windows - A low-level hardware driver is included in the Driver folder on the SDK CD. Open the Windows subfolder and double-click on the CDM 2.04.06.exe file (\Drivers\Windows\CDM 2.04.06.exe). The glove data is then accessible through the USB interface using the AcceleGlove protocol described in Section 3.2.

Mac OS X - A low-level hardware driver is included in the Driver folder on the SDK CD. FTDIUSBSerialDriver_vx.dmg installs the driver in the Mac OS kernel. The glove data is then accessible through the USB interface using the AcceleGlove protocol described in Section 3.2.

Linux Driver - Most Linux distribution kernels include the driver for the AcceleGlove hardware. Plug the AcceleGlove in the USB port and then use lsmod to determine if the ftdi_sio driver loaded. If your distribution did not load the driver, it can be found in the Driver folder on the SDK CD. The AcceleGlove data is then accessible through the USB interface using the AcceleGlove protocol described in Section 3.2.

3.1.2 Installing the AcceleGlove Visualizer

Windows – Open the AcceleGlove Visualizer folder and then the Windows folder. Double-click on the Install AcceleGlove Visualizer executable file 'AcceleGlove Visualizer.exe' and follow the installation instructions.

Mac OS X – Open the AcceleGlove Visualizer folder and then the Mac OS folder. Open the AcceleGloveVisualizer.dmg disk image and drag the Visualizer icon to your applications folder.

Linux – Open the AcceleGlove Visualizer folder and then the Linux folder. Copy the AcceleGloveVisualizer.tar.bz2 to a target directory. Run 'tar xjf AcceleGloveVisualizer.tar.bz2'.

3.1.3 Installing the Software Development Kit (SDK) Files in Windows

The **SDK** folder on the CD contains the following files required for Windows installation:

Common folder that contains:
AcceleGloveSDK.jar
h2-1.1.111.jar
RXTXcomm.jar

Database folder that contains:
DatabaseUtils.props
db folder with predefined gesture libraries

Windows folder that contains:
AcceleGloveSDK-Win.jar
registry.jar
rxtxSerial.dll
ICE_JNIRegistry.dll

Step 1: Copy rxtxSerial.dll and ICE_JNIRegistry.dll to %JAVA_HOME%\lib\ext.
(%JAVA_HOME% is the folder where the JRE is installed on your system.)

Alternatively, you may set the java.library.path property to point to the folder containing the dll's when running java.

Example: java -Djava.library.path=/MyPathToTheSDK/Windows/ -jar MyJarToRun.jar

Step 2: Add the 5 .jar files (AcceleGloveSDK.jar, h2-1.1.111.jar, RXTXcomm.jar, AcceleGloveSDK-Win.jar, and registry.jar) to your project libraries.

Alternatively, you may copy the files to any location on the Java CLASSPATH.

Step 3: Copy the DatabaseUtils.props and db folder to your project base.

3.1.4 Installing the Software Development Kit (SDK) Files on Mac OS X

The **SDK** folder on the CD contains the following files required for Mac OS X installation:

Common folder that contains:
AcceleGloveSDK.jar
h2-1.1.111.jar
RXTXcomm.jar

Database folder that contains:
DatabaseUtils.props
db folder with predefined gesture libraries

Mac OS X folder that contains:
AcceleGloveSDK-OSX.jar
RXTXcomm.jar
librxtxSerial.jnilib

Step 1: Install the RXTX JNI Serial Libraries.
Copy librxtxSerial.jnilib to /Library/Java/Extensions/.

Alternatively, you may set the java.library.path property to point to the correct location when running Java.

Example: java -Djava.library.path=/MyPathToTheSDK/OSX/ -jar MyJarToRun.jar

Step 2: Copy the 4 jar files (AcceleGloveSDK.jar, h2-1.1.111.jar, RXTXcomm.jar, AcceleGloveSDK-OSX.jar) to your Library/Java folder.

Alternatively, you may copy the jar files to any location on the Java CLASSPATH.

Step 3: Copy the db folder and DatabaseUtils.props from the Database directory to your project base.

3.1.5 Installing the Software Development Kit (SDK) Files on Linux

The **SDK** folder on the CD contains the following files required for Linux installation:

Common folder that contains:
AcceleGloveSDK.jar
h2-1.1.111.jar
RXTXcomm.jar

Database folder that contains:
DatabaseUtils.props
db folder that contains predefined gesture libraries

Linux folder that contains:
AcceleGloveSDK-Linux.jar
JNI folder
i686 folder
librxtxSerial.so
x86_64 folder
librxtxSerial.so

Step 1: Install the correct RXTX JNI Serial libraries.
If you are using a 32-bit version of Java (check output of "java -version"), use the librxtxSerial.so in JNI/i686/.

If you are using a 64-bit version of Java, use the librxtxSerial.so in JNI/x86_64/.
Copy the appropriate librxtxSerial.so to $JAVA_HOM/jre/lib/ext.

Alternatively, you may set the java.library.path property to point to the correct location when running Java.

Example: java -Djava.library.path=/MyPathToTheSDK/JNI/i686/ -jar MyJarToRun.jar

Step 2: Copy the jar files to $JAVA_HOME/jre/lib.
Alternatively, you may copy the jar files to any location on the Java CLASSPATH.

Step 3: Copy the db folder and DatabaseUtils.props from the Database directory to your project base.

3.2 *Plugging In the AcceleGlove*

Plug the AcceleGlove USB cable into a USB port on your computer (Figure 1). The USB cable connects to the AcceleGlove via a mini-B connector underneath the "pocket" on the backside of the glove. The USB cable should already be connected to the AcceleGlove. You can plug in more than one AcceleGlove.

Figure 1 – AcceleGlove™ - Computer connection

3.3 *Using the Wireless Module with the AcceleGlove*

Plug the white connector of the AcceleGlove's Wireless Module cable into the 4-pin header on the left hand side of the AcceleGlove's main electronics board underneath the "pocket" on the backside of the glove (Figure 2a). The micro USB connector on the opposite side of the cable must be connected to either one of the micro USB ports on the Wireless Module (Figure 2b). The AcceleGlove's wired USB cable should be disconnected from the mini USB port on the right hand side of the main electronics board. After the cable is connected, turn the Wireless Module's power switch on (towards the outside of the electronics case). The red LED on the Module should illuminate for about a second, and then cycle off and then back on. At this point the glove is ready to make a connection to the computer application. See Appendix section 11.3 for more details regarding the AcceleGlove Wireless Module hardware.

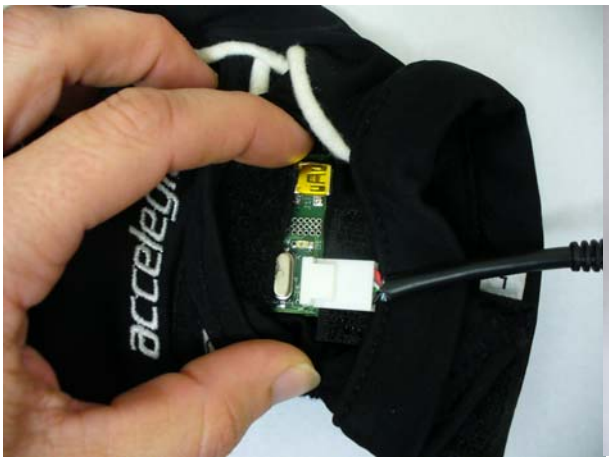


Figure 2a- White Connector at Main Board

Figure 2b- Connector at Wireless Module

3.4 *Configuring the Serial Port*

Note: When configuring the serial port, the following settings should be used:

Baud Rate: 38400
Data Bits: 8
Stop Bits: 1
No Parity, No Flow Control

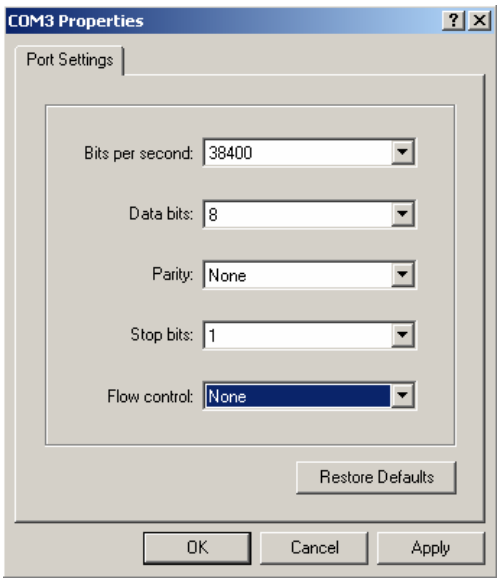


Figure 3 – COM port settings for AcceleGlove communication (shown using HyperTerminal)

3.5 *Putting on the AcceleGlove*

When putting on the AcceleGlove, it is important to adjust each of the four fingers such that the seam at the end of each finger opening is positioned at the first joint from the end of your fingertip (Figure 4). Your entire fingernail should be exposed. This properly positions each finger sensor on the second segment of your finger.

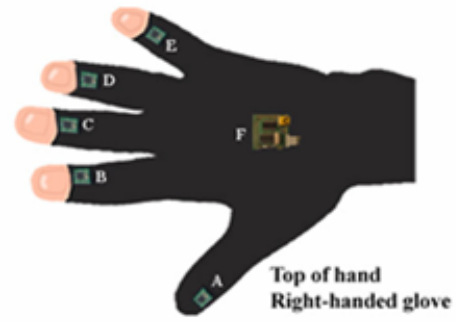


Figure 4 – Electronics layout (top view, right hand)

3.6 *Starting the AcceleGlove Visualizer*

Windows - Click on the Start Menu, select All Programs, select AcceleGlove Visualizer, and click on AcceleGlove Visualizer. Select the appropriate connection type (USB or Bluetooth). With a Bluetooth connection, the AcceleGlove number must be selected from the list in the wireless device window. Click on "Connect" (Figure 5). The Visualizer will then start. Suggestion: Create a shortcut icon for your Desktop.

Mac OS X - Under Applications, open the AcceleGlove Visualizer folder and double-click on the AcceleGlove Visualizer application. Select the appropriate connection type (USB or Bluetooth). With a Bluetooth connection, the AcceleGlove number must be selected from the list in the wireless device window. Click on "Connect" (Figure 5). The Visualizer will then start.

Linux - Open a terminal, change the directory to the location of the AcceleGlove Visualizer application, and run AcceleGloveVisualizer.sh. Select the appropriate connection type (USB or Bluetooth). With a Bluetooth connection, the AcceleGlove number must be selected from the list in the wireless device window. Click on "Connect" (Figure 5). The Visualizer will then start.

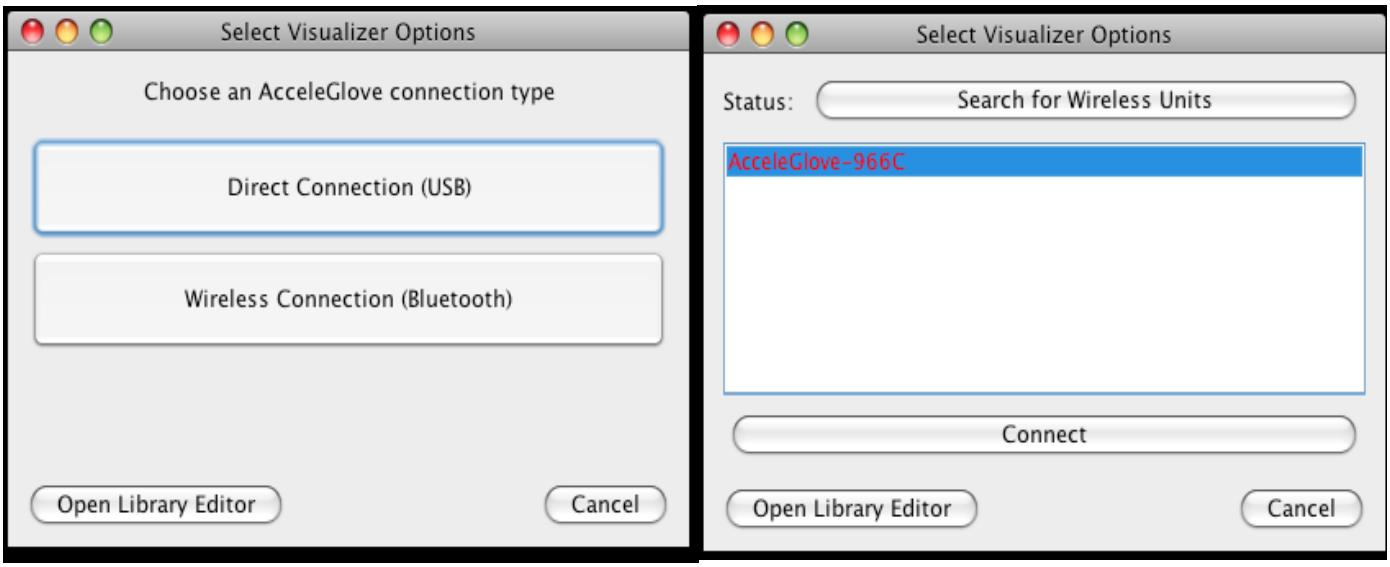


Figure 5 - Visualizer Start Up Menus

For wired USB operation, if the glove is not plugged into a USB port on your computer, an error message, "Failed to find an available AcceleGlove," will appear. Otherwise, the AcceleGlove COM port will be detected automatically and will be noted at the top of the Visualizer screen.
For wireless Bluetooth operation, if the AcceleGlove Wireless Module is not turned on or the battery is too low to operate the device, clicking on "Search for Wireless Units", or selecting a previously detected Wireless Module in the main window will result in a "Could not connect" message.

3.7 *Verifying Operation Using the Visualizer*

The AcceleGlove Visualizer allows you to view the plotting of the output values of each AcceleGlove accelerometer's reading along X, Y, and Z axes versus time.
Once you have plugged in the glove and started the Visualizer, you can quickly verify that your glove and software are installed and working properly.

Figure 6 – Visualizer

As in Figure 6, you should be able to move finger and hand orientation and see the values change. Learn more about the AcceleGlove sensors, communications, and AcceleGlove Visualizer by reading the additional sections in the User's Guide.

4 HOW THE ACCELEGLOVE WORKS

The AcceleGlove has 6 small accelerometers – one for each finger and one for the back of the palm – integrated into the glove to detect finger and hand position and motion.

Each accelerometer has three sensing elements, corresponding to the X, Y, and Z signals output from the AcceleGlove. If your hand is horizontal, the Z-sensing element is oriented along an axis (the "gravity vector") that is perpendicular to the Earth's surface. X and Y both lay in a plane that is perpendicular to the Z-axis, offset from each other by 90 degrees (Figure 7).

The AcceleGlove assigns the following coordinates to the system of accelerometers:

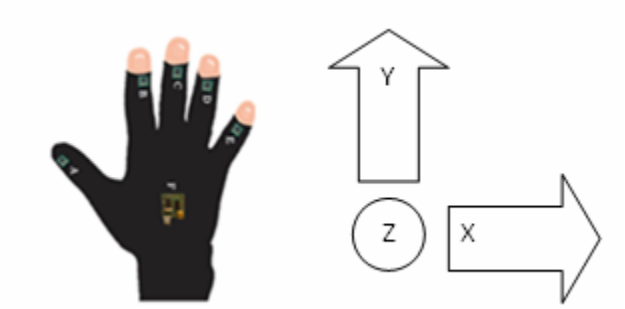
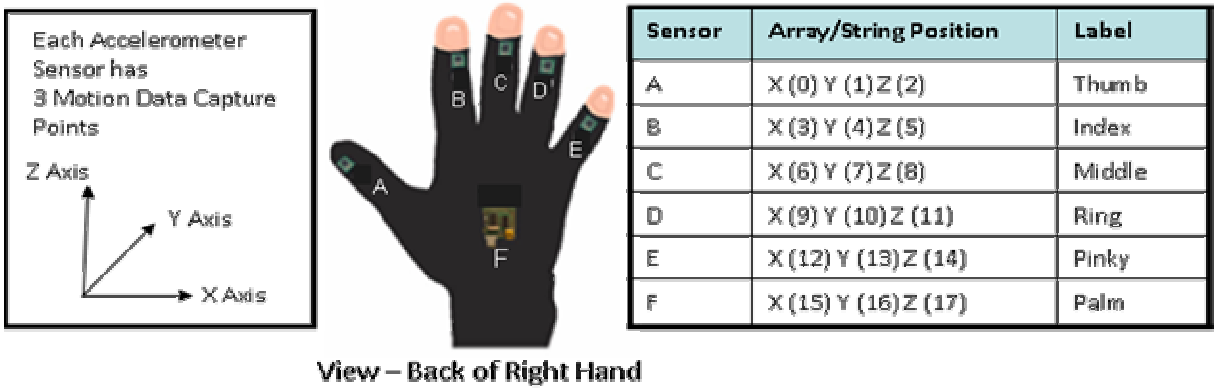


Figure 7 – AcceleGlove output signal convention (top view, right hand)

Tilting the AcceleGlove away from level about the Y-axis (hand supination or pronation) or accelerating the hand in the X-direction will result in a change of the accelerometer signal on the X-axis. Similarly, a signal response will occur on the Y-axis when the hand is tilted up and down around the X-axis (wrist flexion or extension), or moved with acceleration in the Y-direction. The Z-axis will register a response when the hand is accelerated up and down or is rotated about the X-axis and/or Y-axis.

4.1 *Mapping the Sensors*

Figure 8 shows the sensor mappings and corresponding position of the returned sensor values in either the raw data string or the API array. The right and left hand sensor positions are identical, in that A= Thumb, B=Index finger, etc. This is described in further detail in the USB protocol and Java API sections.



Sensor	Array/String Position	Label
A	X (0) Y (1) Z (2)	Thumb
B	X (3) Y (4) Z (5)	Index
C	X (6) Y (7) Z (8)	Middle
D	X (9) Y (10) Z (11)	Ring
E	X (12) Y (13) Z (14)	Pinky
F	X (15) Y (16) Z (17)	Palm

Figure 8 – Sensor Positions

4.2 *AcceleGlove Calibration*

The AcceleGlove is pre-calibrated out-of-the-box for a default range of gestural accelerations mapped from 0 to 255. Recalibrating the AcceleGlove will allow you to refine the range of accelerations. This would be useful if there is a problem with the factory calibration or if the user wants to change the range of acceleration the glove detects. The value of the Y offset value is close to 490. The X and Z offset values range from 200 to 250. Recalibration might cause the AcceleGlove to no longer work with any recorded gesture libraries (for more on calibration, see Section 11.2 of the Appendix).

5. ACCELEGLOVE COMMUNICATIONS PROTOCOL

5.1 *Simple AcceleGlove Communications Protocol (Wired USB Connections Only)*

The data from the glove can be accessed from:

- The glove query communication protocol
- The Java library API

This section documents the glove query communication protocol.

The AcceleGlove uses a simple USB query communications protocol in which the PC software queries the AcceleGlove by sending ASCII characters. The data returned can be in ASCII or binary (byte) format.

Using a terminal emulator, such as HyperTerminal (Windows XP) or MacWise (Mac OS X), you can test and calibrate the glove using this protocol. From a script or custom application of your own, you can access the raw and calibrated glove data.

Command Name: ASCII Query (calibrated values)
Query String: a
Description: Queries the glove for the current calibrated sensor readings. The glove responds with a stream of 18 space-separated calibrated values in ASCII format. The values will be returned in the following order (left to right): thumb, index, middle, ring, pinky, hand, arm link.

Command Name: Binary Query (calibrated values)
Query String: b
Description: The glove responds with a stream of 18 values, each a single byte. The order of the readings is the same as for the ASCII Query command.

Command Name: ASCII Raw Query
Query String: w
Description: Queries the glove for the current raw (uncalibrated) sensor readings. The glove will respond with 18 space-separated raw sensor readings as ASCII text. The order of the readings is the same as for the ASCII Query command. Values are between 0 and 1023 inclusive.

Command Name: Binary Raw Query (raw values)
Query String: r
Description: The glove responds with a stream of 18 raw (uncalibrated) values, 2 bytes per value. Therefore, a total of 36 bytes will be sent.

Command Name: Set Periodic Delay
Query String: t
Description: When "t" is sent to the glove, the glove waits for the user to provide a time value as a 16-bit integer, between 0 and 65,535, representing the wait time in milliseconds between each read/send loop, for "periodic" mode.

Command Name: Enter Periodic Mode
Query String: p

Description: When "p" is sent to the glove, the glove enters in "periodic" mode. The glove will enter a read/send loop with a wait time between loops specified by the "t" command. The default "t" wait time is 1 millisecond. In periodic mode, the glove will send raw (uncalibrated) values, 2 bytes per value, for a total of 35 bytes .

Command Name: Enter Calibration Mode
Query String: k
Description: Causes the glove to enter calibration mode. (See Calibration section above for more information.)

Command Name: Firmware Version
Query String: v
Description: Causes the glove to return a text string of the version number of the firmware that is terminated by a newline.

Command String	AcceleGlove response	Mode
a	18 ASCII values	Calibrated
b	18 byte values	Calibrated
w	18 ASCII values	Uncalibrated
r	18 values (18 two byte values)	Uncalibrated
p	(Loop) 18 byte values then wait 't' milliseconds	Uncalibrated/Periodic
t	Set periodic wait, in milliseconds (2 byte int)	Periodic
k	Enter Calibration mode	Calibration
v	ASCII string of firmware version number	N/A

Figure 9 – AcceleGlove Communication Commands

5.2 *Advanced AcceleGlove Communications Protocol*

The AcceleGlove Advanced Protocol supports communication with the AcceleGlove via either wired USB serial or through the AcceleGlove Wireless Module via Bluetooth serial. The protocol supports a limited subset of the normal AcceleGlove protocol but supports all major commands used by normal AcceleGlove operation.

The protocol uses a simple message format for all commands that consists of the following items:

OP Code	Length	Data (optional)
1 byte	1 byte	Length bytes

The following host commands are supported. Host commands are sent from the controlling computer to the AcceleGlove.

OP Code	Byte Value [Hex (decimal)]
Get Transport Version	0xF0 (240)
Set Data Options	0xF2 (242)
Read Data Options	0xFD (253)
Start Data Scan	0xF4 (244)
Stop Data Scan	0xF5 (245)

The following response messages may be sent from the AcceleGlove or AcceleGlove Wireless Module.

OP Code	Byte Value [Hex (decimal)]
Report Glove Version	0x8F (143)
Report Transport Version	0x8E (142)
Report Data Options	0x8D (141)
Report Calibration Data	0x81 (129)
Report Data	0x80 (128)
Report Data Scan Stopped	0x82 (130)
Report Data Scan Start	0x84 (132)
Report Error	0xFF (255)

OP Code Descriptions:

Get Transport Version
This zero length command requests the communication transport version. The responding device will send a Report Transport Version message.

Set Data Options
This variable length command will set the data option bytes. The message will send no response message. To read the data option bytes, send a Read Data Options message.

Read Data Options
This zero length command requests the current values of the data options bytes. The responding device will send a Report Data Options message.

Start Data Scan

This zero length command requests that the responding device start sending Report Data messages. The device will continuously send Report Data messages (as limited by the current data options) until a Stop Data Scan message is received. It is important to note that once data reporting starts, a Stop Data Scan message should be sent before sending any other message.

Stop Data Scan

This zero length command requests that the responding device stop sending Report Data messages. The responding device will send a Report Data Scan Stopped message. Once the Report Data Scan Stopped message is received, it is safe to send any other command message.

Report Glove Version

This 3-byte message reports the version of the AcceleGlove firmware. This message is sent after the receipt of a Start Data Scan message before sending any Report Data messages. The 3-byte data package consists of the following bytes:

Byte 1	Byte 2	Byte 3
Major Version Number	Minor Version Number	Revision Version Number

Report Transport Version

This 3-byte message reports the version of the transport device. If communicating directly with an AcceleGlove this version will always be 0.0.0. If communicating via an AcceleGlove Wireless Module, this will report the AcceleGlove Wireless Module's firmware version. The data format of the message is identical to the Report Glove Version message.

Report Data Options

This variable length message reports the current values of the data reporting options as set by the Set Data Options message. This message may report more data reporting option bytes than are currently supported by this protocol.

Report Calibration Data

This 36-byte message sends the 18 two-byte integer calibration table of the AcceleGlove. This message is sent after a Start Data Scan message before a Report Data message. If the current data reporting options request calibrated data, this message will not be sent. See the section Data Format below for details.

Report Data

This variable length message reports a current set of AcceleGlove sensor readings. Regardless of the data reporting options set, the format of this message always starts with accelerometer values. If Extra data bit is sent in the first byte of the data reporting options, then an extra 3 bytes are included to report the additional values. See the section Data Format below for details.

Report Data Scan Stopped

This zero length message is sent in response to a Stop Data Scan message. After receipt of this message it is safe to send any other command message.

Report Data Scan Start

This 1 byte message indicates that the Start Data Scan message has been processed. If the payload byte is 0 then the scan has successfully started. Otherwise and error has occurred, and the scan has not started.

Data Reporting Options

Currently 4 data option bytes may be set as reported below.

Byte 1	Byte 2	Byte 3	Byte 4
Data Options	Delay Low	Delay High	Trigger Bit

Data Options:

The first byte of the data options is a bit mask as described below.

Bit	7	6	5	4	3	2	1	0
Function	None	None	None	Trigger Cont	Trigger High	Trigger Low	Extra	Data

bit 7 - 5: Not currently used, set to 0.

bit 4: Trigger Cont - If this bit is set, continuous data will be reported as long as the trigger condition is true. Either Trigger High, or Trigger Low must be set for this to occur.

bit 3: Trigger High - If this bit is set, Data will be reported when the digital input port bit named in the trigger bit data option byte changes from logic 0 to logic 1. If Trigger Cont is set, then data will be reported as long as the input port bit is logic 1.

bit 2: Trigger Low - If this bit is set, Data will be reported when the digital input port bit named in the trigger bit data option byte changes from logic 1 to logic 0. If Trigger Cont is set, then data will be reported as long as the input port bit is logic 0.

bit 1: Extra - If this bit is set, data messages will contain the digital input port values, and the additional analog input value.

bit 0: Data - If set to 1 calibrated accelerometer values will be reported. If 0 uncalibrated accelerometer values will be reported.

Delay Low:
Delay High:

These two bytes form a 16 bit unsigned integer delay time in milliseconds. The AcceleGlove will wait this period of time after sending a data message before attempting to send another data message. This limits the maximum data reporting rate or the AcceleGlove.

Trigger Bit:

This value is between 1 and 6, and indicates which digital input port bit to use for any specified trigger condition. If no trigger condition is set in the data options byte then this byte is ignored.

Data Format

The number of bytes, and content of a Report Data Message varies depending on the Data Options byte of the Data Reporting Options. The table below shows the size of the message based on the Data, and Extra bits of the Data Options byte.

	Data = 0	Data = 1
Extra = 0	36	18
Extra = 1	41	23

If uncalibrated values are requested (Data=0), then the first 36 bytes are the 18 accelerometer axes each reported as a 16 bit (2 byte) unsigned integer. This value is sent with the least significant byte first.

If calibrated values are requested (Data = 1), then the first 18 bytes are the 18 accelerometer axes each reported as an 8-bit unsigned integer.

If the Extra bit is set, the remaining 5 bytes consist of the following:

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Digital Port Values	A/D #1 Low Byte	A/D #1 High Byte	A/D #2 Low Byte	A/D #2 High Byte

The calibration table contained in the Report Calibration Data message contains 18 two-byte values which can be used to convert uncalibrated data to calibrated data. The Calibration data contains the maximum value seen by the glove during calibration (~1g). To create a calibrated value subtract the current reading from the calibration entry. Note that the X and the Z-axis of the palm accelerometer are inverted. To calibrate those axes subtract the calibration value from the current reading.

6 USING THE ACCELEGLOVE VISUALIZER

6.1 *AcceleGlove Visualizer Introduction*

The AcceleGlove Visualizer is a multi-purpose application that works with the AcceleGlove. It provides the ability to view the output values of each accelerometer's readings along X, Y, and Z-axes (Figure 10). The Visualizer also can be used to capture and define gesture libraries, recognize gestures, and display gesture recognition probability, data output, and diagnostics.

The AcceleGlove Visualizer application was developed using the AcceleGlove SDK API. The source code for both a simple Visualizer and the full version can be found on the SDK CD and can be used by developers as an example application.

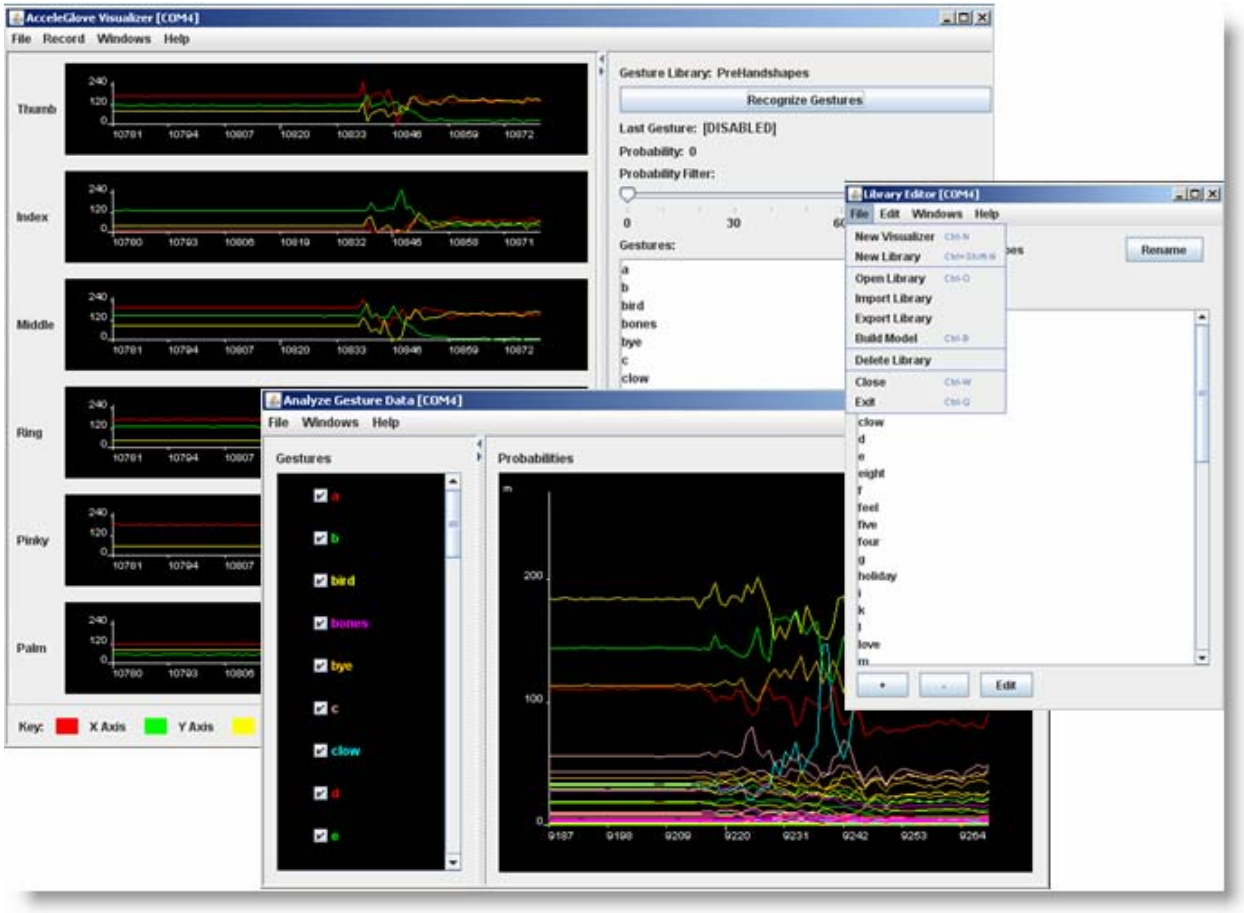


Figure 10 – AcceleGlove Visualizer

6.2 *AcceleGlove Visualizer Basics*

Please refer to section 3.6 on how to start the Visualizer in Windows, MAC OS X, and Linux.
The AcceleGlove Visualizer allows you to view the output values of each AcceleGlove accelerometer's reading along X, Y, and Z axes versus time. Learn more about the AcceleGlove sensors, communications, and AcceleGlove Visualizer by reading the additional sections in the User's Guide.

Figure 11 – Visualizer Main Screen

The main screen of the AcceleGlove Visualizer displays data being streamed from the AcceleGlove (Figure 11).

The far left side of the chart displays each of the AcceleGlove's accelerometer names (e.g., "Index" for the Index finger, etc.). Within the chart area, the X, Y, and Z-axis data are plotted in the corresponding colors shown in the legend (X- Red, Y-Green, and Z-Yellow). The scale displayed on the left (0 – 255) is the factory default data range to which the AcceleGlove accelerometers are calibrated (for more on calibration, see Section 11.2 of the Appendix). Along the bottom of each chart is the number of data samples captured from the start of the session.

Displayed to the right of the accelerometer window is the name of the Gesture Library, the last gesture recognized (if one has been loaded), and probability settings.

6.3 *Viewing Two or More AcceleGloves*

To view or analyze the data from more than one AcceleGlove, you can open a second session of the AcceleGlove Visualizer. To do this, select New Window from the file menu. Use this function when more than one AcceleGlove is plugged in, and you want to read data from a pair of gloves.

6.4 *Recognizing Gestures*

The AcceleGlove Visualizer can be helpful in capturing and then recognizing gestures. First, we will review how to recognize a gesture using a predefined library. A predefined library of American Sign Language Letters (ASL) is shipped with the AcceleGlove. To see how to position your hand to make these gestures, reference Section 8.2 *Using Predefined Gestures*.

Step 1: From the File Menu select Open Library.
Select a library from the list and click OK (Figure 12).

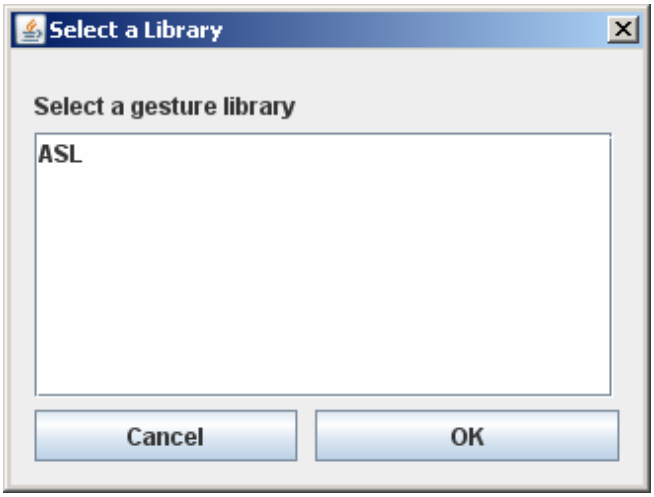


Figure 12 – Selecting a gesture library

The library selected will now be loaded, and each gesture within the library is displayed in the gestures list in the left pane.

Step 2: From the Left Panel of the Main Screen (Figure 13), select Recognize Gesture.

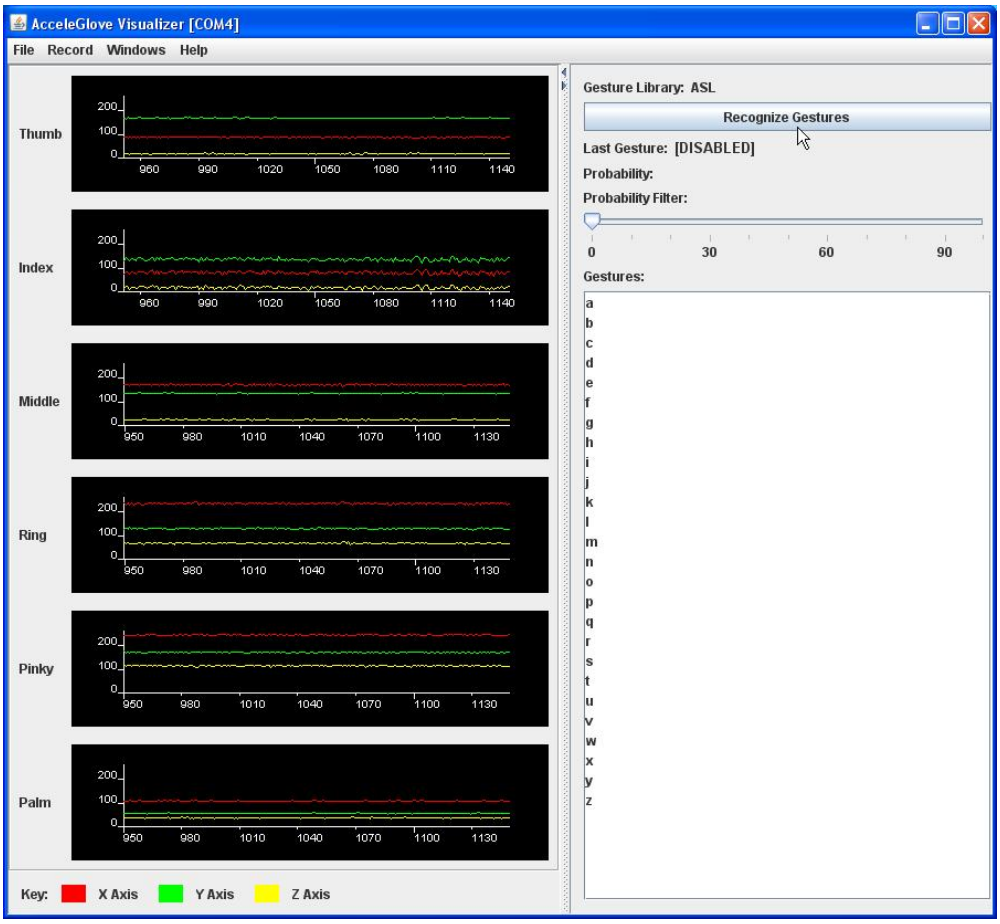


Figure 13 – Visualizer Main Screen

Step 3: Make the Gesture With the AcceleGlove.
The AcceleGlove Visualizer will recognize and display the name of the gesture being made along with the probability that this is the gesture being made.

There is a user-settable probability acceptance threshold that tunes the recognition engine to be more or less discriminating. Only gestures above the probability acceptance threshold will be displayed and reported. For example, if the filter is set to 90%, only gestures that achieve between 90% and 100% probability will be displayed as recognized gestures (Figure 14).

By default, the probability acceptance threshold filter is set to 0 to show all the possible matches. Depending on user experience, typical acceptable recognition probabilities are generally in the range of 40% – 60%.

Simply repeat Steps 2 and 3 to recognize another gesture.

Step 4: Click Stop Gesture Recognition.

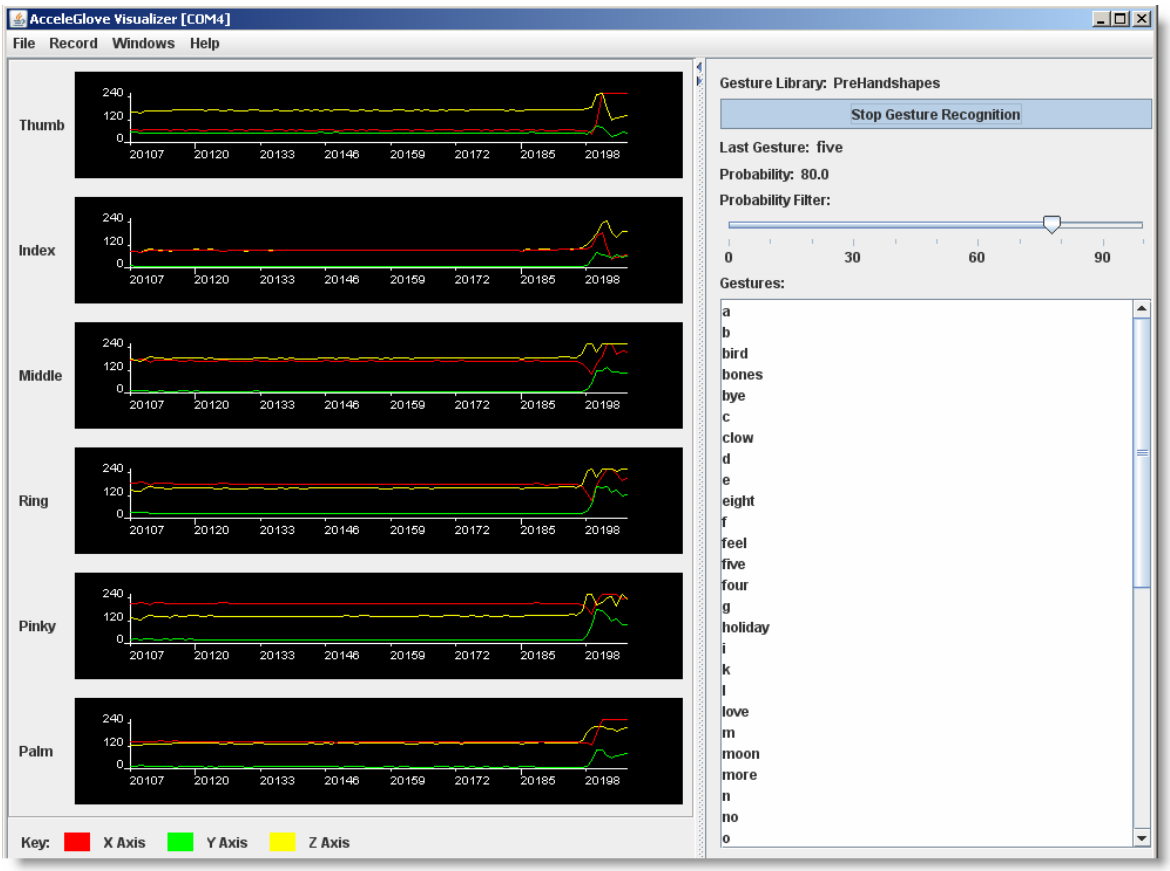


Figure 14 – Recognizing Gesture

6.5 *Training New Gestures*

New gestures can be trained by capturing one or more instances of the gesture. When there are multiple instances recorded of the same gesture, recognition reliability is high because the system is matching against multiple samples.

6.5.1 Expected Volumes and Performance

While there are no data limitations in terms of the number of gestures that can be captured and saved to a library, it should be understood that the more gestures that are captured (samples or training instances), the longer it will take to build the predictive model. For example, on a typical modern CPU, it will take 30 seconds to build a 20-gesture model, and it will take 83 minutes to build a 200-gesture model.

6.5.2 Creating a New Library

In order to train new gestures, you must first create a library in which to store them.

Step 1: From the Windows Menu, select Open Library Editor (Figure 15).

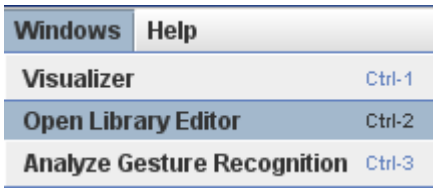


Figure 15 – Open Library Editor

Step 2: From the File Menu, select New Library.
Click Rename to give the library a new name.

6.5.3 Recording and Modifying Gestures

Step 1: Click the "+" at the bottom of the Library Editor to add a new gesture.

Step 2: Make the gesture with the AcceleGlove and then click Capture (Figure 16).

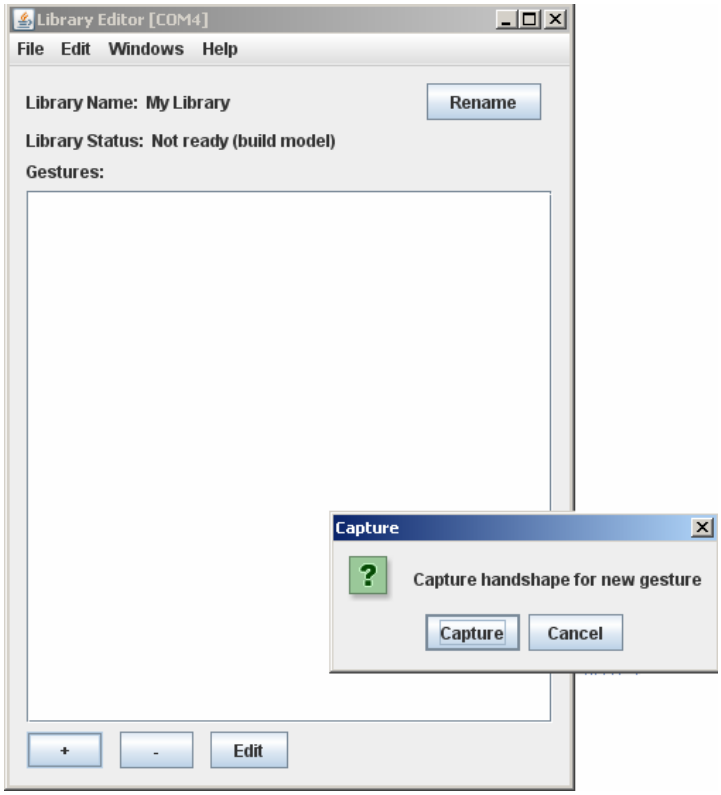


Figure 16 – Capturing a new gesture

Step 4: Give the gesture a name.

Note: the gesture library model must be built (Section 6.5.4) before gestures can be recognized.

Step 5: To edit the gesture, click Edit.

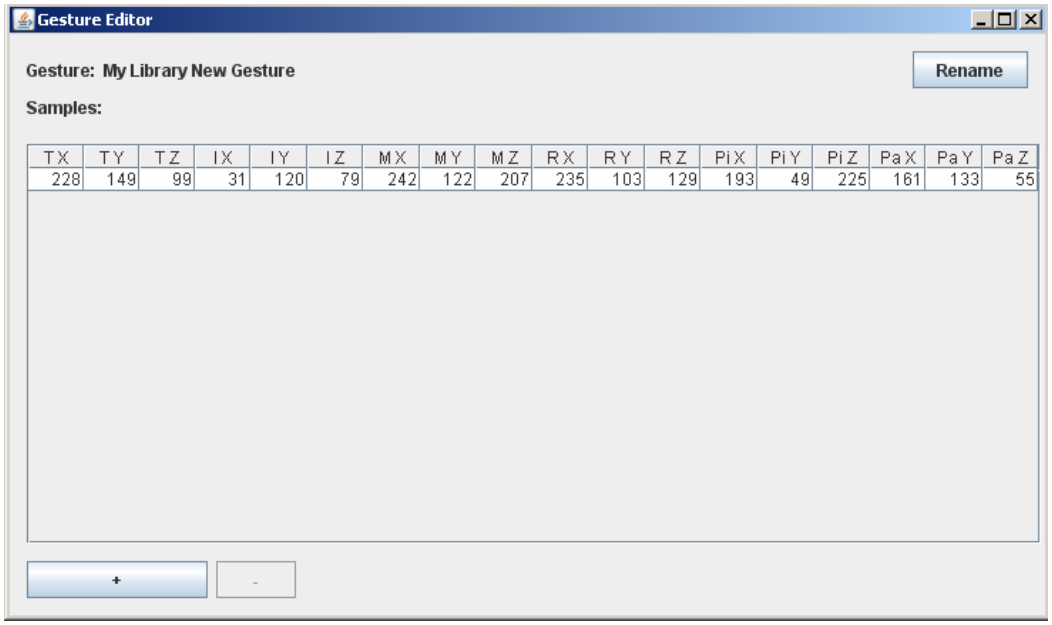


Figure 17 – Gesture Samples

The Gesture Editor Window opens (Figure 17).
Click the "+" to add additional samples to your existing gesture.
Click Rename to change the name of the gesture.
Click the [X] in the upper right corner to close the Gesture Editor window once finished.
Click the [X] in the upper right corner to close the Library Editor window once finished.
(Note: you must record at least two gestures for the system to build.)

6.5.4 Building the Gesture Library

In order for the Visualizer to recognize your gestures, you must build the corresponding predictive model from the trained instances. To do this, select Build Model from the File Menu (Figure 18).

Step 1: From the Windows Menu, select the Library Editor.

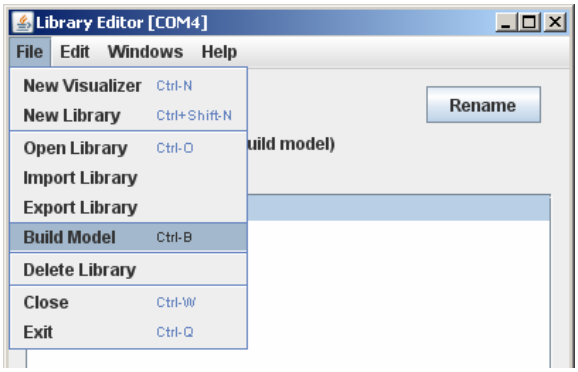


Figure 18 – Build Model

Step 2: Select Build Model.
If the library that opens (the last library opened) is not the library you wish to build, first select Open Library, then select the desired library, and return to Select Build Model.

Once the model is built, you can follow the instructions in Section 6.4 *Recognizing Gestures*.

6.6 *Analyze, Capture, and Output Data*

Another capability of the AcceleGlove Visualizer is to analyze and export data captured from the AcceleGlove.

To Analyze Gesture Recognition Data:

Step 1: Be sure you have selected and loaded a library (From the File menu, select Open Library).

Step 2: From the Windows Menu, select Analyze Gesture Data.

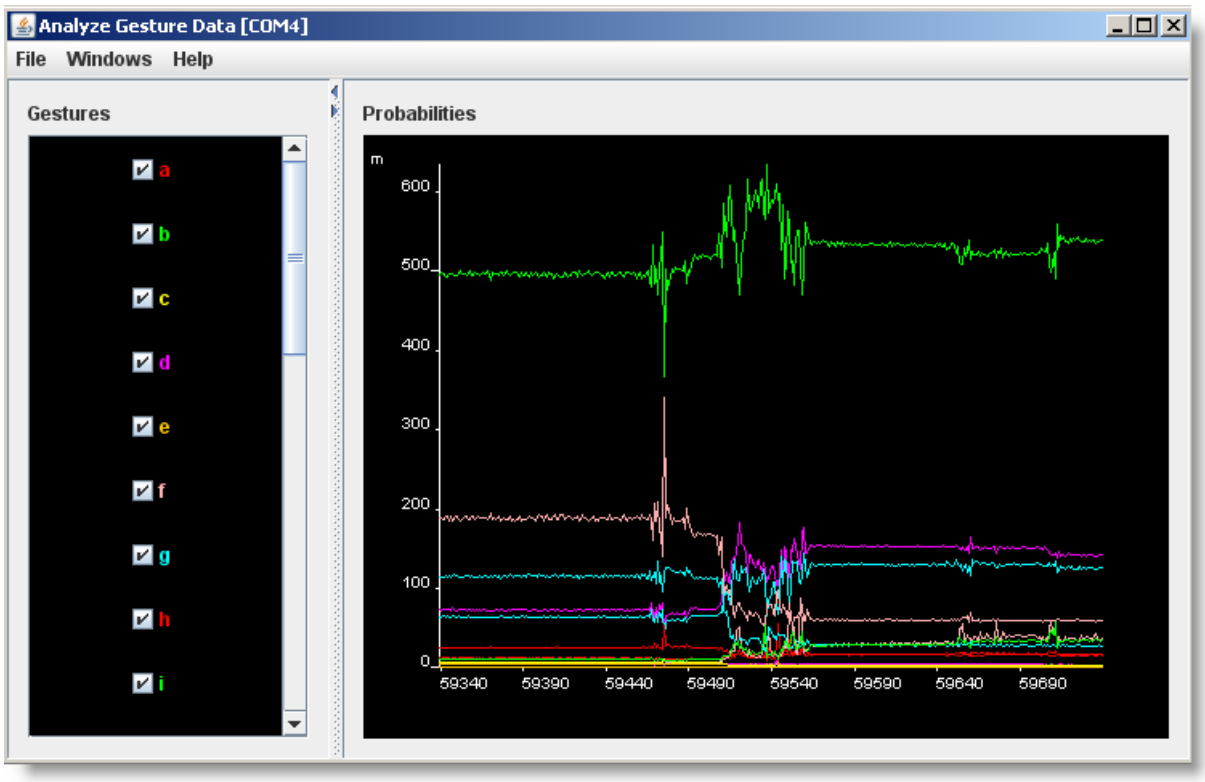


Figure 19 – Gesture Data Analyzer

The gestures for the selected library are displayed on the left panel of the Gesture Data Analyzer window (Figure 21). The data set for each gesture in the library is plotted in the graph to the right. Each gesture is displayed in a different color, allowing you to easily compare the probability of occurrence of the gesture being made to the probability of occurrence to other gestures in the library. The data displayed at the bottom of the graph is the number of samples received from the AcceleGlove since the start of the analyzer session.

To Record Data:

Step 1: From the Record Menu, Select Start Recording.

Based on configuration options set in the recording options (defined below), the data stream will begin to record (Figure 21).

Step 2: Click Stop Recording.

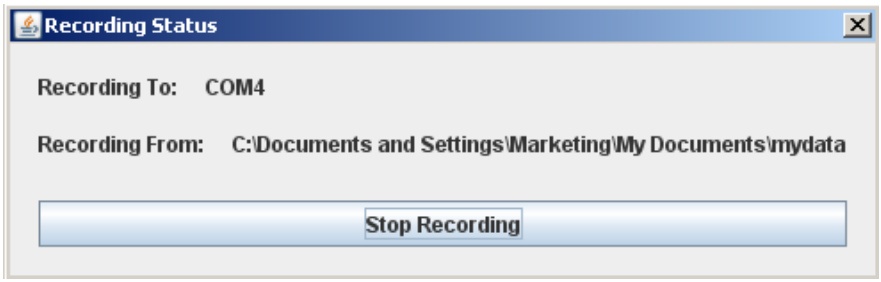


Figure 20 – Recording Status

This will output the data into a comma-separated variable (.csv) file, which can be imported into Microsoft Excel.

The **Recording Options** let you determine what type of data you want to record, the time options, and where you want to store the captured data.

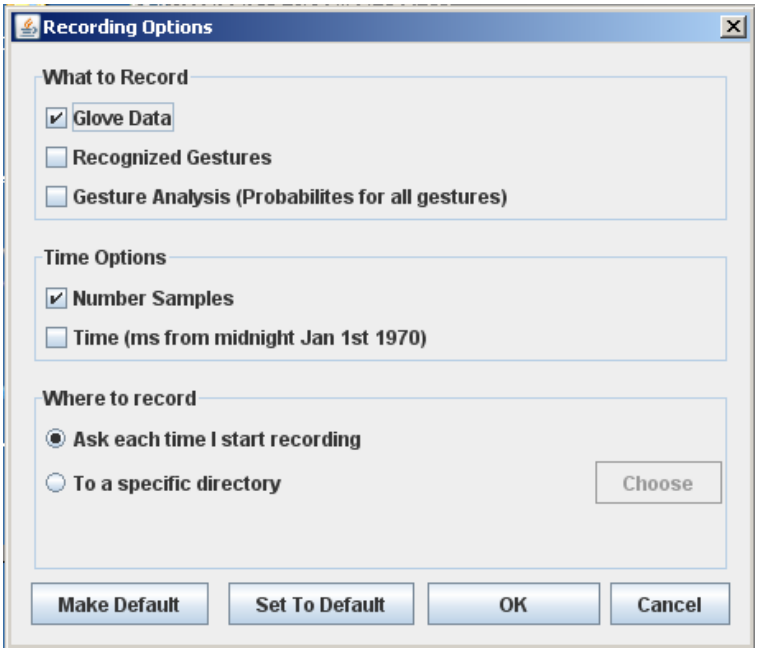


Figure 21 – Recording Options

What to Record includes the option to select one or more of the following for output: Glove Data (sample count and the value of each accelerometer), Recognized Gestures (the name of the gesture recognized and probability), or Gesture Probabilities (all gestures and probability).

In addition, you can choose to capture sample count and/or time and specify the directory to which you would like to record the data file. If you specify the directory, files will be named automatically.

6.7 *Sharing, Saving, and Loading Libraries*

You may wish to share Gesture Libraries or move them from environment to environment.

Under the Library Editor Menu, use the Import and Export options to export a library from one environment and import to another (Figure 22).

Figure 22 – Import Gesture Library

7 USING THE SOFTWARE DEVELOPMENT KIT (SDK)

Note: Be sure to have installed the files as addressed in Section 3.1 to use the SDK.

7.1 *Guide Conventions*

7.1.1 Example Code Format

Example code is presented in the following font:

```
package mypackage;
import java.util.LinkedList;
public class MyIntStack {
      private final LinkedList fStack;
      public MyIntStack() {
            fStack = new LinkedList();
      }
```

7.1.2 Code Comments Format

Comments are presented in the following format:

```
/*
 * This is a sample comment
 */
```

7.1.3 Notes Format

Note: Will be formatted in a box and in blue.

7.2 *SDK Overview*

The AcceleGlove SDK Version 1.1.0 provides the necessary software for developers to interface with the AcceleGlove, as well as build gesture libraries and recognize gestures.

Figure 23 provides an architectural overview of the SDK. The data streamed from the AcceleGlove can be accessed in raw form (string) by using one of the hardware drivers provided and the AcceleGlove protocol, or by using the Java API. The Java API runs on Windows 7, Windows XP, Windows Vista, Mac OS X, or Linux operating systems. JRE (Java Runtime Environment) 5 or higher is required (and distributed with the SDK). Any Java IDE (such as Eclipse) can be used for development.

The AcceleGlove SDK provides two Java API sets: A "Raw DataStream API" that accesses the X, Y, Z axis values, and a "Gestures" API that uses the raw data stream to create and access gestures. (Note: Currently, a gesture includes handshape and palm orientation. Shoulder, elbow, and upper arm sensors are planned for a future release and will be added to the Gestures API.) Sample source code is provided to demonstrate how each API can be used.

In addition, the Visualizer application, which also uses the API, is provided in both compiled and source forms.

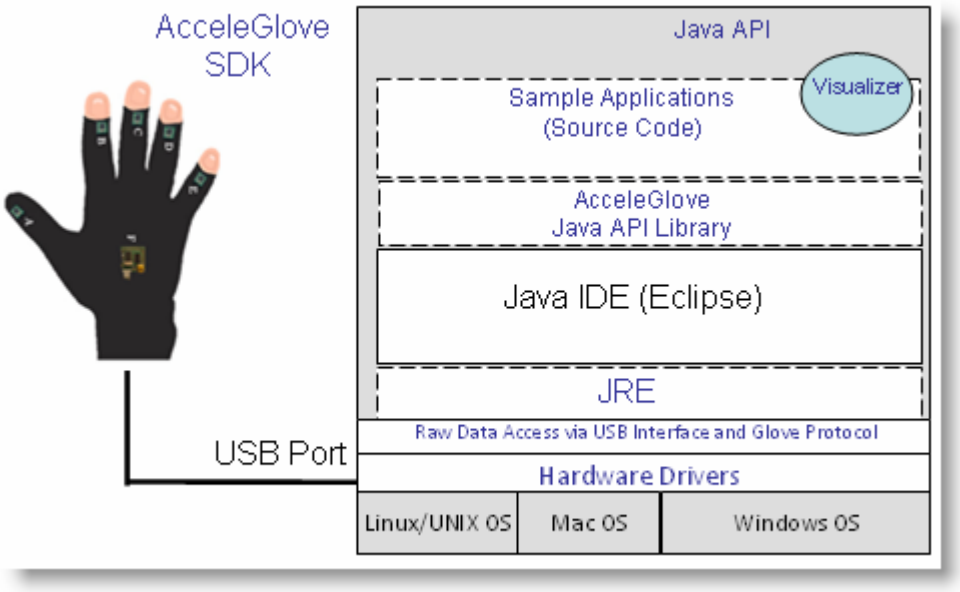


Figure 23 – SDK Component Diagram

7.2.1 SDK Components

The components provided with the SDK include (Figure 23):

Common Components:

Component	**Description**
AcceleGloveSDK.jar	Library of class for glove interface, and gesture recognition Includes the WEKA machine learning library
h2-1.1.111.jar	H2 Database used to store gesture library information
RXTXcomm.jar	The RXTX serial port library
DatabaseUtils.props	Properties used to locate the gesture database
db folder	Predefined Gesture libraries
Examples	Source of the AcceleGlove Visualizer, and other example applications

Windows ONLY Components:

Component	Description
AcceleGloveSDK-Win.jar	Library to support detection of AcceleGloves
RXTXSerial.dll	JNI Library to support RXTX library
registry.jar	Library to support registry functions in AcceleGloveSDK-Win.jar
ICS_JNIRegistry.dll	JNI Library to support registry.jar

Mac OS X ONLY Components:

Component	Description
AcceleGloveSDK-OSX.jar	Library to support detection of AcceleGloves
librxtxSerial.jnilib	JNI Library to support RXTX library

Linux ONLY Components:

Component	Description
AcceleGloveSDK-Linux.jar	Library to support detection of AcceleGloves
JNI/i686/librxtxSerial.so	JNI Library to support RXTX library. (For 32 bit java)
JNI/x86_64/librxtxSerial.so	JNI Library to support RXTX library. (For 64 bit java)

8 APPLICATION EXAMPLE

8.1 Example Application Location

Examples of how to use the SDK are located on the SDK CD in the Examples folder. Source code for the AcceleGlove Visualizer is also on the SDK CD.

8.2 Using Predefined Gestures

The gestures shown in Figure 24 have been pre-recorded and can be recognized by the AcceleGlove.

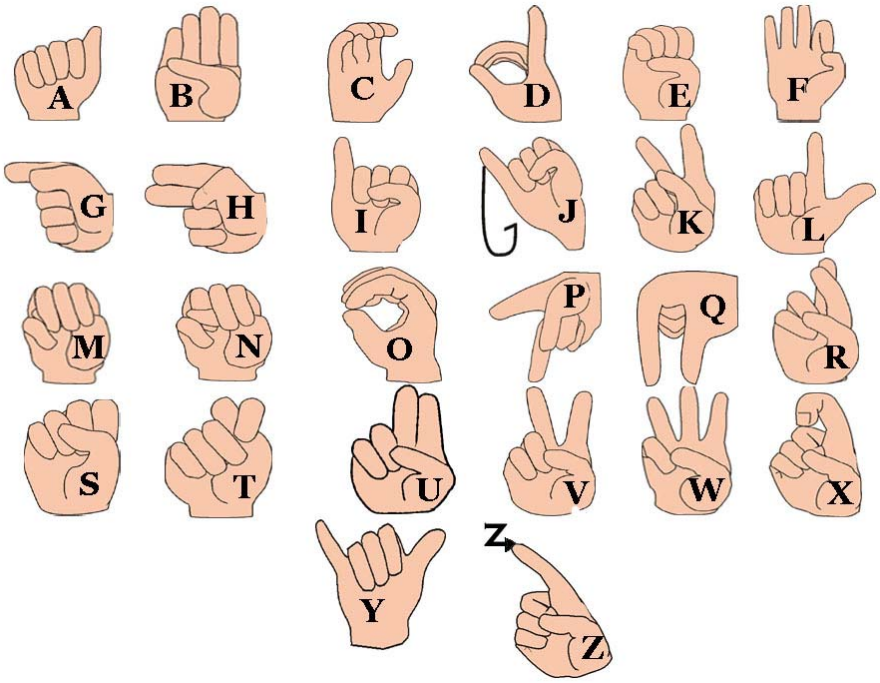


Figure 24 – ASL Fingerspelling Library

9 MICROCONTROLLER INPUTS

The layout of the AcceleGlove main electronics board is shown in Figure 25 (top view). The inputs have been grouped into different ports (e.g., J1). See below for port descriptions. During typical use, the user should not have to interact with the AcceleGlove electronics. The main AcceleGlove electronics board has been designed with some auxiliary inputs (J2, J5) to allow advanced electronics users to access the microcontroller or attach additional sensors to the board for research projects. ***However, any attempt to use these ports or make modifications to the AcceleGlove electronics will void the warranty.***

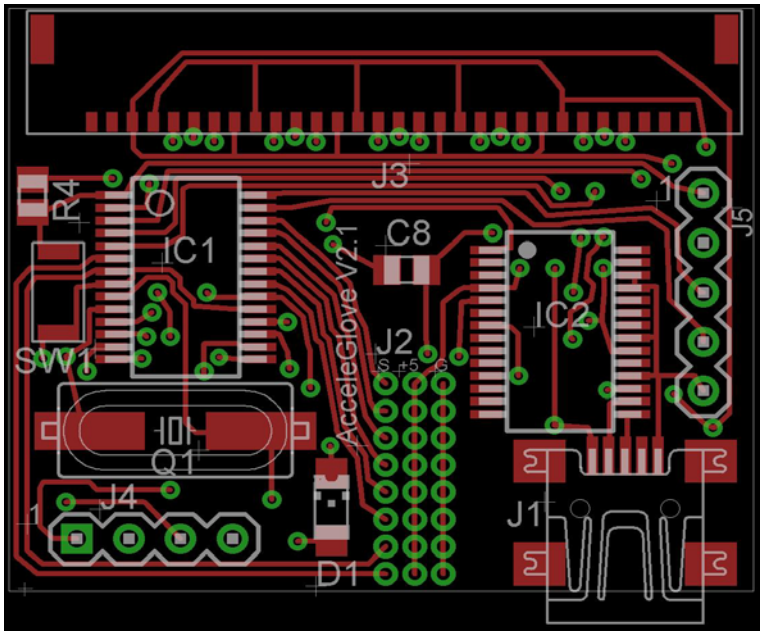


Figure 25 – Main electronics board schematic (top view)

J1 – USB Mini-B Connector (to computer)

J2 – Auxiliary Input Ports to Microcontroller
The AcceleGlove provides extra (auxiliary) inputs to allow designers and researchers to integrate additional sensors with the glove's six accelerometers. These inputs can be accessed at port J2 (3x8 block of through-holes) on the main board. Referring to the schematic figure above, the first column on the left (signal column) is where the auxiliary inputs should be connected – the hole second from the bottom is an analog input, and the rest are digital inputs. The AcceleGlove also can provide limited power to additional sensors via the middle column, which is connected to the USB +5V supply. The rightmost column of through-holes is connected to Ground. ***Be aware that any modifications to the AcceleGlove electronics will void the warranty.***

J3 – Connection Out to Finger Accelerometer Boards

J4 – Arm Link Connector
This 4-pin connector is designed to connect to an additional arm motion capture device, which is still being developed.

J5 – Programming Header
These five pins allow researchers direct access to the microcontroller (PIC16F876A) in order to re-program it. ***Be aware that reprogramming the AcceleGlove microcontroller will void the warranty.***

The port pinout is as follows:

Pin 1: MCLR
Pin 2: VDD (+5V)
Pin 3: PGD (data)
Pin 4: PGC (clock)
Pin 5: GND (ground)

SW1 Reset Switch
Do not use the reset switch.

Pressing the reset switch will reboot the AcceleGlove. The reset switch should only be used when updating firmware.

10 SPECIFICATIONS AND WARRANTY

10.1 *AcceleGlove Specifications*

Sensors: Six 3-Axis Micro Electromechanical Accelerometers

Sensor Positioning: One Accelerometer On The Back Of Each Finger And Thumb
One Accelerometer On The Back Of The Hand

Material: Nylon / Lycra mix

Computer Interface: USB Virtual Serial Port

Port Settings: 38400 bps, 8 data bits, no parity, 1 stop bit, no flow control

Supply Voltage: 5.0V (Via USB)

Output Signal Resolution: 8 to 10-bit

Range:
Orientation: 180 Degrees
Acceleration: +/- 1.5g

Sampling Rate (max): 120 Hz

8 Auxiliary I/O: 7 Digital inputs
1 Analog input

Weight: 54 grams (size large)

System Requirements:
OS: Windows 7/2000/XP/Vista, MAC OS X (10.4 or higher, Intel-based), Linux (Kernel 2.6.9 or higher)
RAM: 512MB or more
Java Runtime Environment (JRE): Version 5 or higher

10.2 *Warranty*

10.2.1 30-Day Customer Satisfaction Assurance

If for any reason you are not satisfied with this product, please contact your supplier to see if they can assist you. If they are unable to resolve your issue, you may return the product within 30 days undamaged and in its original packaging for a full refund of the purchase price. The refund will be issued once the product has been received and declared complete and undamaged. The purchaser is responsible for all shipping costs.

10.2.2 Product Warranty: 90 Days

This product warranty goes to the original purchaser. The AcceleGlove and SDK are warranted to be free of defects in materials and workmanship for a period of 90 days from the original date of purchase. During the warranty period, we will repair or replace components that are defective with new, refurbished, or reconditioned parts.

Exclusions:

- This warranty is provided for private, non-commercial use only.

- This warranty is void under the following conditions:
  a) Misuse of the product, or
  b) Unauthorized modification of the product, or
  c) Violation of the product license.

10.2.3 Filing a Warranty Claim

A. Do not return the product without first contacting your supplier. Your supplier will assist you in determining if a return is required.

B. Your supplier will issue you a return authorization number to be included with the returned product. Do not return the product without this authorization number written **ON THE OUTSIDE** of the packaging.

C. Package the product to be returned, as well as all parts, software, and documentation, securely in the original packaging.

D. Include the following information:
   - Contact Name, Address, Telephone Number, and E-mail
   - Reason for the return
   - Original Bill of Sale
   - Return Authorization Number

E. Ship the product prepaid to your supplier. The supplier will not be responsible for any mailing or shipping charges, and will not accept any COD shipments. The supplier is not responsible for any damages that may occur during shipping.

11 APPENDIX

11.1 *Terms and Definitions*

Handshape: For the purpose of the SDK, the term handshape is being used to define the collective positions of the fingers to form a specific shape with the hand (e.g., a thumbs-up or fist). The Handshape class and AcceleGlove also have the ability to capture the palm orientation, and therefore, handshape is extended to mean the collective positions of the fingers, in addition to the orientation of the palm. For example, thumbs-up and thumbs-down, while sharing the same collective positions of the fingers, have different palm orientations and, therefore, are considered different handshapes.

Gesture: A gesture is an occurrence of a handshape including the palm orientation.

Dynamic Gesture: A dynamic gesture adds motion to the gesture. For example, the starting position of a dynamic gesture might be a thumbs-up and the ending position, a thumbs-down. It is the capture of the entire range of positions or gestures that make up a dynamic gesture. This can include the same handshape with multiple palm orientations (thumbs-up to thumbs-down) or changes in handshape and palm orientation (thumbs-up to thumbs-down to a fist).

Sample: The collection of data from each accelerometer. For a gesture, a single sample would include three data values for each of six accelerometers. It is common to capture multiple samples of a single gesture to improve the probability of detection, as the data values will vary slightly with minor changes to the position of the fingers and palm.

Gesture Library: A collection of gestures, each gesture having one or more samples (sets of data).

Model: The predictive model that is generated using the data captured from the AcceleGlove's accelerometers to read the data from the glove to determine if, in fact, the occurrence of the gesture has happened.

11.2 Calibration

Calibration: To recalibrate the AcceleGlove, you must use a terminal emulator, such as HyperTerminal (Windows XP) or MacWise (Mac OS X), and configure the port as described in the AcceleGlove Protocol section (38400 bps, 8 Data Bits, No Parity, 1 Stop Bit, No Flow Control). Once connected, press "k" to enter calibration mode. The calibrated value equals the offset value minus the raw value. The value of the Y offset is close to 490; the X and Y offset values range from 200 to 250.

Command Name: Calibration

Command String: k

Command Description:
When "k" is sent to the AcceleGlove, the glove enters the calibration routine, answering with a MENU (see screenshot below).

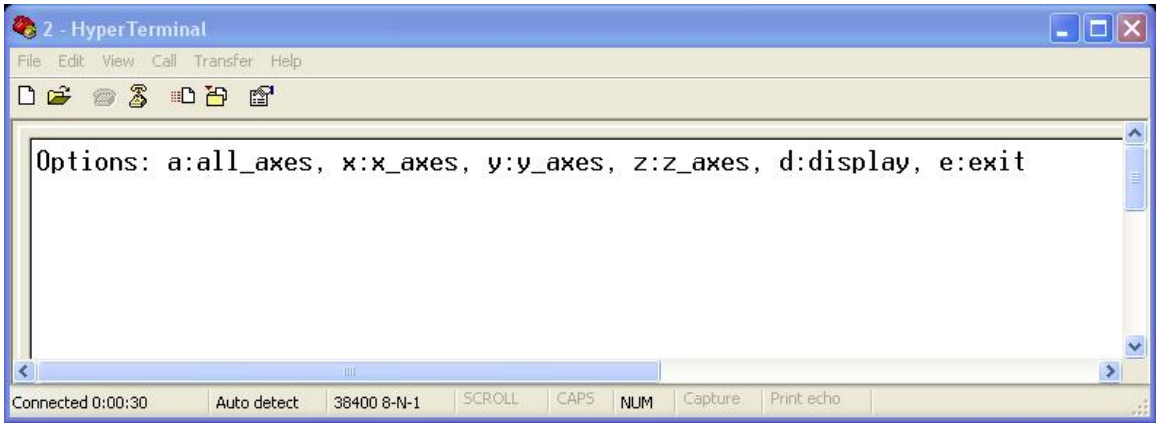


Figure 26 – Calibration routine screenshot (shown using HyperTerminal)

First, choose whether you want to calibrate all axes of all accelerometers at once (“a”), all X axes of all accelerometers at once (“x”), all Y axes of all accelerometers at once (“y”), or all Z axes of all accelerometers at once (“z”). Sending “d” will display all offset values; sending “e” will exit the calibration routine.

Once a calibration option has been selected, another menu is presented. Select from among clearing calibration offset values (clear), displaying offset values (read), starting calibration (calibrate), and exiting the calibration routine (exit).

It will probably make the calibration process easier and more precise if you first use Scotch or painters tape to tape the glove to a moveable flat surface (such as a piece of cardboard). When taping down the glove, make sure to align all fingers straight (along the X axis).

If “all_axes” was previously selected:
Once “calibrate” is chosen, slowly rotate the entire glove +90 and then -90 degrees (from horizontal) about both the X and Y-axes (one at a time). Then, to calibrate the Z-axis, slowly rotate the glove such that it is completely upside down.

If “x_axes” was previously selected:
Once “calibrate” is chosen, slowly rotate the glove +90 and then -90 degrees (from horizontal) about the X-axis (same as supination / pronation movements). Press any button other than “k” to save and exit this calibration.

If “y_axes” was previously selected:
Once “calibrate” is chosen, slowly rotate the glove +90 and then -90 degrees (from horizontal) about the Y-axis (same as wrist flexion / extension movements). Press any button other than “k” to save and exit this calibration.

If “z_axes” was previously selected:
Once “calibrate” is chosen, slowly rotate the glove such that it is completely upside down. Press any button other than “k” to save and exit this calibration.

To set the correct offset values during calibration, make sure to rotate the full 90 degrees (or 180 degrees / upside down for Z) in either direction, aligning each axis with gravity. The offset values correspond to the maximum values reached by each axis.

11.3 *Wireless Module Hardware*

The Wireless Module converts the wired USB AcceleGlove into a battery powered, Bluetooth wireless device. Your computer may or may not have built in Bluetooth capability. If not, we recommend a USB to Bluetooth adapter such as Cirago BTA6210 v.2.1 EDR Class 1 Micro Bluetooth Adapter, or Cirago Micro USB Bluetooth 3.0 Class 2 Adapter for use with the Wireless AcceleGlove.
When using these adapters, do not load the Toshiba Bluetooth Stack included in the adapter's install CD Rom. You should be able to plug the adapter into your computer's USB port out of the box, and your computer will automatically make use of standard Bluetooth drivers.

11.3.1 Charging the Battery

Included with the AcceleGlove Wireless Module is a wall-charging unit that will recharge the Lithium Polymer battery. If the battery is not fully charged, the yellow LED on the Wireless Module will light up when the charger is plugged in (unless the battery is severely discharged, in which case the LED will not turn on until the battery voltage increases above a certain level). When the battery is fully charged, the yellow LED will turn off, and the charging plug should be removed from the Wireless Module. The Wireless Module and glove may be used while the charging plug is inserted and charging, although the battery-charging rate will be significantly reduced. A continuously blinking red LED may indicate a severely discharged battery, and may indicate that the battery needs to be replaced.

11.3.2 Operation

When the Wireless Module is first turned on by moving the power switch towards the outside of the Wireless Module's enclosure, the red LED will turn on and then off over the course of a few seconds, and then stay on continuously. At this point, the Module is ready to make a wireless connection, and the Visualizer Bluetooth option button can be selected. The green LED will turn on continuously when the Wireless Module has made a Bluetooth connection, or will blink in diagnostic mode.

Either of the micro-USB connectors on the AcceleGlove Wireless Module may be used when attaching the AcceleGlove Wireless Module's cable to the 4-pin header on the left hand side of the AcceleGlove's main board.

In order to use your AcceleGlove with the AcceleGlove Wireless Module, the AcceleGlove must be running firmware version 1.3.0 or higher. The Wireless Module and updated glove will not work with SDK Visualizer Version 1.0.0. The version information can be found in the Help/AboutVisualizer tab in the main Visualizer screen. Please contact techsupport@acceleglove.com to be sent a link to the latest SDK software and glove firmware.

The glove will work either as a wired USB device, or a wireless Bluetooth device. Under no

circumstances should both the AcceleGlove's USB cord and the Wireless Module's power and communication cable be attached to the AcceleGlove.

11.3.3 Wireless Module Specifications

Communication Architecture: Wireless Bluetooth

Battery: 3.7 V, 900 mAh Lithium-Ion

Charging Port: accepts 3.7- 7V wall transformer with 2.35mm power plug, positive center

Strap: 13 inch elastic and Velcro loop

Connector: 7-inch micro-B to 0.100" center 4-pin female header

Port Configuration: 38,400 8N1 NHS

Battery Life- 6-12 hours, depending upon use

Weight: 2 oz with connector and strap