PROSPECTS OF IMPLEMENTING A VHAND GLOVE AS A ROBOTIC CONTROLLER

CHRISTOPHER CHIDI
School of Electrical & Computer Engineering
Georgia Institute of Technology

DR. AYANNA HOWARD
School of Electrical & Computer Engineering
Georgia Institute of Technology

"There are numerous approaches and systems for implementing a robot controller. This project investigates the potential of using the VHand Motion Capturing Glove, developed by DGTech, as a means of controlling a programmable robot. A GUI-based application was utilized to identify and subsequently reflect the extended or closed state of each finger on the glove hand. A calibration algorithm was implemented on the existing application source code in order to increase the precision of the recognition of extended or closed finger positions as well as enhance the efficiency of the hand signal interpretation. Furthermore, manipulations were made to the scan rate and sample size of the bit signal coming from the glove to improve the accuracy of recognizing dynamic hand signals or defined signals containing sequential finger positions. An attempt was made to sync the VHand glove signals to a Scribbler robot by writing the recognized hand signals to a text file which were simultaneously read by a Python-based application. The Python application subsequently transmitted commands to the Scribbler robot via a Bluetooth serial link. However, there was difficulty in achieving real-time communication between the VHand glove and the Scribbler robot, most likely due to unidentified runtime errors in the VHand signal interpretation code."

I. INTRODUCTION

Robots are a pivotal aspect of modern technology because machines are created for the simple goal of making life easier. Through machines, humans are able to automate redundant processes; eliminate human error from a production system; explore or excavate hazardous environments; and accomplish many other mundane or superhuman activities. As research in artificial intelligence develops, robotic autonomy becomes ever more feasible. The methods of controlling or communicating with robots transcend beyond employing a remote control or transmitting encoded commands through a communication link. The capabilities of artificial intelligence motivate various research endeavors that focus on communication with robots through physical interaction or visual and verbal cues.

A potential communication mechanism currently under investigation involves the visual recognition of hand signals. Humans have already established a code of sign language used for communication between people who have hearing or vocal deficiencies. Robots could be programmed to recognize these signals by using a camera to visually record hand signal images from a human, and through image processing algorithms, interpret these recorded hand signals. If robots can distinguish between varying hand signals, then programmable logic could be used to interpret the hand signal and execute a functional command.

The building blocks to such research endeavors involve manipulating the DGTech VHand Motion Capturing Glove, in order to communicate with a programmable robot. By utilizing a binary search tree populated with signal definitions, it was possible to alter the C++ based GUI hand signal application, provided by DGTech, to associate various finger positions to a calibrated / predetermined hand signal definition [1]. The ultimate goal of hand signal recognition research would be to implement similar algorithms investigated in this study in order to design robots capable of recognizing sign language through simulated machine vision by means of cameras and other visual sensors . This mode of robotic interfacing would expand the realm of artificial intelligence, making it ever more feasible for human to robot interaction in daily life.

II. BACKGROUND AND RELEVANCE OF DIGITAL GLOVE RESEARCH

Before delving into the details of the VHand research project, it is important to address the related research which preceded the VHand glove and provided a technological basis from which to expand upon. Common uses of glove-based input devices include virtual reality applications, human to computer gesture interfacing, and teleoperative (remote) controls [2]. Most of the teleoperative control applications, in which glove-based input is used to remotely manipulate or control a particular device, relate to the aim of VHand research which seeks to control the Scribbler robot teleoperatively using the VHand glove. Researchers from the Universidad Autónoma del Estado de México investigated the potential of teleoperation through glove-based input in a project that utilized a digital glove to control a robotic arm [3]. The capabilities of the digital glove implemented in this project exceed that of the VHand glove; as the glove, developed 5th Dimension Technology, is able to measure the pitch and roll of a user's hand as well as the finger positions. The researchers mapped the signal output from the digital glove to a control scheme that manipulated the rotation of the shoulder, elbow, and wrist of the robotic arm and further indicated whether the gripper claw should open or close. The results of this project yielded support for the teleoperative control of robots using glove-based input, and in addition demonstrated that computer data networks can serve as a communication channel between digital gloves and robots.

Human to computer gesture interfacing is another aspect of glove-based input research explored in a study by The Robotics Institute at Carnegie Mellon University [4]. The researchers developed a hand signal recognition system which could acknowledge sign language executed by a user wearing a Virtual Technologies 'CyberGlove'. The CyberGlove is comparable to the VHand as they use similar sensor technology to record the finger positions of the user [5]. The hand signal recognition system developed at Carnegie Mellon was based on a statistical model known as the Hidden Markov Model (HMM) which essentially incorporates probability distributions to identify sequences. In this case the sequences, or gestures, were comprised of individual hand signals referred to as states of the sequence. The system would associate a series of states as a predefined gesture and could also learn new gestures by receiving user input on the definition of the new gesture and the states it consists of. Though the recognition system was able effectively identify gestures and learn new ones, the complexity of the system's algorithm made it difficult to implement. For testing purposes, the researchers treated the glove hand as a whole; meaning the system could only recognize open hand and closed fist positions, and not individual finger positions.

The research with the VHand glove attempts to integrate the teleoperative and gesture interfacing aspects of glove-based input to develop a system where a robot can recognize gestures executed by a human and consequently operate based off of the given hand signals. The VHand signal interpretation system implements a node-based model, in the form of a binary search tree, to recognize hand signals from the glove. The binary search tree is an algorithm that allows the computer to search through a hierarchical model of nodes, which are defined as individual hand signal positions . As the computer receives input from the glove indicating a particular hand signal, the computer will transition through the model until it reaches a node signifying the completion of a gesture. This article will expound more on the algorithm developed for hand signal recognition and shed light on the possibilities of teleoperative robotic control using sign language.

III. DETAILS ON HARDWARE

A. VHand Glove Specifications

The specifications on the VHand Glove, Figure 1, are as follows [6]:

The glove communicates with a PC using a port connection with a DB9 serial cable. The capturing application, from which the research project is based, is a C++ application compiled with Microsoft Visual Studio.

IV. APPROACH FOR VHAND SIGNAL INTERPRETATION

A. Original Source Code

The original hand signal interpretation application, provided by the DGTech developers, contained the initial source code for the GUI motion capturing application. The application, shown in Figure 2, primarily consisted of a visual display which represented the glove hand of the user.

There are five bars representing the five fingers of a hand. The bars graphically indicate how far each finger can extend or contract as detected by the sensors. When a finger is contracted the signal reading from the sensor will be close to 0, which is reflected on the display as a white or empty bar. As the user begins to extend the finger, a blue filler rises along the bar indicating the percentage of contraction detected by the sensor. The signal coming from the sensors are 10 bits in length which correspond to a minimum value of 0 and a maximum value of 1023 .

Table 1: VHand Glove Specifications

Sensors:	5 resistance bend sensors, one per finger
Resolution:	10 bit (1024 positions)
Transmission:	RS-232 38400 BAUD full duplex
Frequency:	100 Hz
Power Supply:	12 V DC
OS:	Windows 95, 98, 2000, XP

Table 1: VHand Glove Specifications

B. Prior Source Code Modifications

Numerous modifications were made to the developer's source code in order to enable the glove hand capturing application to recognize and interpret specific sign language hand signals, including the addition of two classes: Interpreter.cpp and Tree.cpp. These classes are responsible for defining the finger configurations for each hand signal and subsequently matching the hand signals to it respective robot signal command.

Figure 1. Two images of the DGTech VHand glove. The picture above shows the complete DGTech glove package including the serial link, com port, and power adapter.

A five bit signal was created in the Interpreter class and is composed of one bit for each finger on the glove hand. The most significant bit represented the thumb and each subsequent bit represented the next finger on the hand with the least significant bit representing the pinky finger. A threshold value was calculated by averaging the maximum and minimum sensor values in a series of manual calibration tests to determine a reliable threshold by which the Interpreter could consistently distinguish between an extended or contracted finger. This value was established as ~28% of the maximum signal range, so that if the 10 bit signal from the finger sensors exceeded the 28% threshold, a value of 1 was given to the corresponding bit in the 5 bit signal. For instance, if the user were to extend only their index finger past the sensor value of 288 then the corresponding 5 bit signal created by the Interpreter class would be 01000. From this 5 bit signal, the interpreter class could identify the hand signal executed by the user and iterate through the binary search tree to find specific definition which represents the signal 01000. The following is a list of all predefined hand signals of capturing application [1].

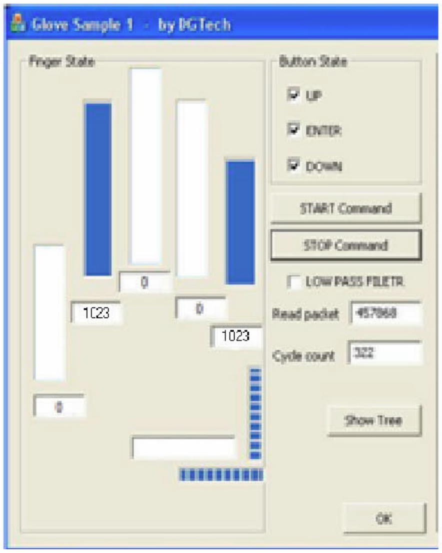

Figure 2. A screenshot of the original GUI interface of the interpreter application provided by DGTech.

C. Recent Contributions and Code Modifications

Following the source code modifications, in Section B, there were several problems remaining that hindered the effectiveness of the application. Foremost, the finger sensors were quite volatile and the current formula used to calculate the threshold was not robust enough to account for the unpredictable nature of the sensors. The sensor output of the maximum and minimum finger position values were never close to 1024 or 0, respectively, in operating conditions. The recorded values of extended and closed finger positions would intermittently fluctuate during each execution of the application. Moreover, the size of the user's hand would also impact the signal readings. These flaws had the potential to cause the application to not recognize the correct finger positions of the glove user. Furthermore, it was difficult to perform a dynamic hand signal that could be recognized by the application because the application could not effectively transition between nodal states due to scanning and timing issues.

To correct these issues, an algorithm was implemented into Interpreter.cpp, which replaced the old threshold calculation formula. The user was given an option after running the application to initialize a calibration sequence. The calibration sequence prompted the user to fully extend all fingers for five scan rate cycles and then fully close all fingers for another five cycles. The recorded maximum and minimum finger position values were registered and used in a new formula which takes the difference between these values and defines the threshold as a variable percentage of the difference.

Table 2: Static Hand Signals

Signal Definition:	Binary Representation:
"Zero"	00000
"One"	01000
"Two"	01100
"Three"	11100
"Four"	01111
"Five"	11111
"Six"	01110
"Seven"	01101
"Eight"	01011
"Nine"	00111
"Distress"	00100
"Good Job"	10000
"Fly"	10001

Table 2: Static Hand Signals

In addition, manipulations were made to the sample size and window size variables in order to increase the scan rate of the application, since the original scan rate was too slow to be effective. Originally the sample size was 20 packets, meaning that the Interpreter.cpp class would register and average 20 consecutive 10 bit packets from the glove sensors to generate one sample to be interpreted by the class. Furthermore, the window size variable dictates how many samples are averaged together to identify one set of finger positions or hand signal. The window size value was originally set to 30. Noting that the default scan rate of the glove sensors is 100 Hz as indicated in the hardware specifications, a sample size of 20 and a window size of 30 would correlate to a functional scan rate of 1/6 Hz or the recognition of one finger position every 6 seconds. Testing proved that this rate was too

Table 3: Dynamic Hand Signals

Signal Definition:	Binary Representation:
"Come here"	{01000, 00000, 01000, 00000}
"Go"	{01000, 00000, 00100, 00000, 01000, 00000, 00100}
"Bye bye"	{01111, 00000, 01111, 00000}
"Wait / Pause"	{11111, 11110, 11100, 11000, 10000}
"Hold / Get"	{11111, 00000}
"Open"	{00000, 00001, 00011, 00111, 01111, 11111}
"Shoot / Fire"	{11000, 01000, 11000, 01000}
"Stop / Hault"	{00000, 11111}

Table 3: Dynamic Hand Signals

slow to effectively detect dynamic hand signals; therefore, the sample size was reduced to 5 packets and the window size reduced to 20 samples which resulted in a functional scan rate of 1 Hz or one hand signal per second. A scan rate of one hand signal per second represents a more realistic rate at which sign language signals are executed, resulting in an increase in the effectiveness of the interpreter application.

V. COMMUNICATION BETWEEN THE VHAND GLOVE AND SCRIBBLER ROBOT

To test the practicality of the VHand glove serving as a robotic controller, an attempt was made to employ the VHand as a controller for a Scribbler robot, shown in Figure 3. However, due to the lack of a Bluetooth serial API for C++ applications, the VHand application could not directly communicate with the Scribbler robot, which is reliant on Bluetooth communications.

A work-around communication medium was conceived which involved writing the hand signal output commands to a text file. Concurrently a Python-based application would read the text file, associate the command with an applicable Scribbler function, and transmit the command to the robot. Given the complexity of the interpreter application's source code, this approach provided a simple and feasible means for porting the signal commands from the C++ based signal interpreter application to the Scribbler robot.

For the sake of simplicity, only basic commands were used for the purpose of testing the communication link with the scribbler. The commands were Forward (FRWD), Backward (BACK), Turn Left (LEFT), and Turn Right (RGHT). The practice of using 4 letters to represent each command made it easier to read in the commands using the Python application.

To establish a Bluetooth serial communication link between the Python application and the Scribbler robot, several key steps found in the Institute of Personal Robots in Education (IPRE) Robotic Services website must be performed [7]. Primarily, BASIC Stamp Editor v2.4.2 was utilized to program the Stamp microcontroller on the robot and download the IPRE services which enable use of the Fluke device, Figure 4.

The IPRE Fluke facilitates a Bluetooth serial link for wireless communication to the Scribbler robot. After reprogramming the Scribbler robot's microcontroller using the IPRE services, the robot is capable of recognizing various functions and commands executed in Python.

Figure 3. The Scribbler robot used to test the teleoperative capabilities of the VHand glove. The Scribbler's API has built-in functions to control the movement of the robot, play sound bytes, and activate the onboard LEDs.

The Python application contained a simple script which read in the command currently stored in the output text file and consequently sent the appropriate Scribbler function command to the robot. The program would loop repeatedly using a While() loop and a time delay of 1 second was implemented to slow the rate at which the application would send commands to the robot. If the recognized command from file had not changed between scan cycles, the application would not send another command.

Although the concept seemed sound in theory, the computer could not handle both the C++ and Python applications running simultaneously. However, the Scribbler robot could execute a list of hand signal commands that were saved to a text file after the glove application recorded a sequence of hand signal commands.

VI. RESULTS AND DISCUSSION

Currently, by utilizing the calibration sequence, the glove capturing application is capable of recognizing all of the hand signals performed by any user no matter the size of their hand or the condition of the glove sensors. The glove was tested by both a male and female researcher. Despite the fact that the glove did not fit properly on the male's hand, both researchers were able to successfully execute all of the defined hand signals including the

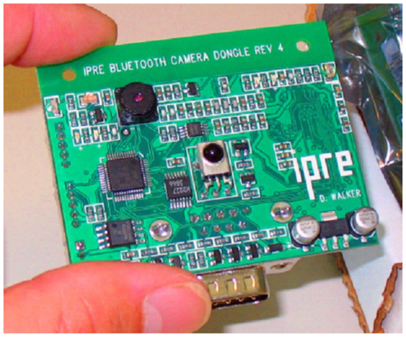

Figure 4. The IPRE Fluke device facilitates Bluetooth serial communication to the Scribbler robot. The Fluke is attached to the serial port located on the front of the robot.

dynamic signals that involved 5 - 6 sequential finger positions. This testing indicated that the signal recognition problems which remained after the initial code modifications have been corrected. Figure 5 displays a screenshot of the signal interpretation GUI after successfully executing a dynamic hand signal command.

In the execution of the "Shoot / Fire" command shown in Figure 5, the user must execute the following sequence of hand signals involving the thumb and index finger: {11000 01000 11000 01000}. The completion of this sequence is indicated in the binary value tracking field on the right side of the GUI.

Note that although the sensor values were not exactly zero for the contracted fingers, the program was still able to recognize these sensors as being in the closed position due to the threshold values set from the calibration algorithm.

Moreover, the current state of the project is able to provide support for the fact that sign language, through glove-based input, can be used to control a robot. Even though the VHand glove was unable to communicate with the Scribbler robot in a real-time environment, it is feasible that a researcher can develop the necessary run-time environment using a different programming language to achieve real-time communication.

In concomitance to the real-time communication issue is the fact that the glove application is prone to arbitrarily crashing during runtime. This runtime error is inherent to the developer's

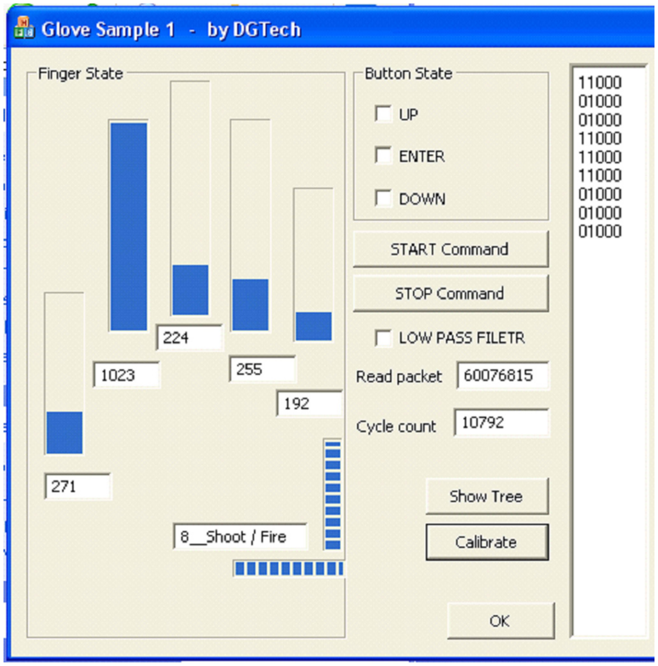

Figure 5. Screenshot of the hand signal interpreter application showing the successful execution of the "Shoot / Fire" dynamic hand signal. The text field on the right side of the GUI tracks the binary signals received from the glove.

source code and was documented in an earlier research report [7] detailing the preceding code modifications mentioned in Section 4B. Efforts were spent debugging the program in order to isolate the cause of this runtime error, yet the cause remains unknown. A probable speculation is a memory leak error in the interpreter application, which causes the computer to crash when the application exceeds the available RAM. Another supposition is that the VHand drivers provided by DGTech are defective.

Regardless of the issues involving communication between the VHand and the Scribbler, there are several lessons to be learned from this research endeavor. Primarily the algorithm for signal interpretation in conjunction with the method for calibrating the glove to a user's hand is an applicable technique for future endeavors involving glove-based input. The VHand signal interpretation system boasts a simple recognition algorithm that is robust enough to identify hand signals involving each of the 5 fingers in the hand. Pending advancements in robotic image processing, concepts of this algorithm can be applied to a robot; enabling it to recognize human sign language visually using a camera as the sensory input.

VII. FUTURE WORK

There are many possibilities for advancement which remain with regards to the development of the hand signal recognition application. One immediate change would be to improve the means of implementing the calibration algorithm. As is, the calibration sequence is optional, and can be activated by pressing its respective pushbutton in the GUI application. However, the configuration settings established after calibration are lost when the program is closed. A useful feature to include would be to have the option of saving the finger position configuration of the user so that they would not have to repeat the calibration sequence on every execution of the program application.

If the reliability of the finger sensors improves, there would be an opportunity to allow for more finger states than simply extended or closed. This could be implemented by increasing the hand signal size from 5 bits to 10 bits which would enable up to four recognized positions for each finger. Another option would be to forego using binary signals for finger position recognition and, instead, use the 0 to 1024 integer value utilized in the developer's source code. Variables could be created to store each integer value representing the fingers of the glove hand. This integer value can be converted into a percent value of finger contraction. This implementation would enable greater possibilities when controlling a robot by allowing enhanced signal recognition capabilities. Glove users would be able to execute forms of sign language, such as the letter C, which require the computer to acknowledge a bend in the fingers as a defined finger position and not simply extended or contracted.

One other focus of improvement for the VHand glove capturing application is to identify the cause of the runtime errors. These errors pose a hindrance to the application's viability and restrict potential utilization of the application. One consideration is to recode the program in a different programming language such as C# or Python, which may eliminate any potential memory leaks existing in the current source code. Moreover, these programming languages are supported by the IPRE robotic services and provide Bluetooth serial capabilities which allow for wireless communication between the VHand glove and a robot.

Refining the programming of the VHand signal interpretation algorithm would pave the way for various captivating applications of the VHand glove. Using a Bluetooth serial link, it would be possible to facilitate teleoperative communication of the VHand glove with an autonomous humanoid such as the Robosapien. The Robosapien is a commercially available, toy robot programmed with artificial intelligence and personality attributes. The robot comes with a remote control to activate any of the various pre-programmed functions; however, the Robosapien can also interact with its environment without human control. Future research could explore the plausibility of reprogramming the Robosapien to receive Bluetooth signals from the VHand glove. Employing the VHand signal interpretation algorithm, a nearby computer could interpret the hand signals coming from the VHand glove and wirelessly relay the appropriate commands to the Robosapien. This would effectively replace the controller that comes with the robot allowing contemporary sign language to dominate the behavior of the robot.

VIII. ACKNOWLEDGMENT

Thanks to Dr. Ayanna Howard and Hae Won Park for the opportunity to engage in this robotic research project during the 2008-2009 academic school year.

VX. REFERENCES

Trilisky, Nazar. "Hand Signal Recognition with VHand." Undergraduate Research Report, Georgia Institute of Technology, Atlanta, GA, 2008.

D. J. Sturman and D. Zeltzer, "A Survey of Glove-based Input," IEEE Computer Graphics and Applications, vol. 14, no. 1, February 1994.

M. Romero Huertas, J. R. Marcial Romero, and H. A. Montes Venegas, "A Robotic Arm Telemanipulated through a Digital Glove," IEEE Fourth Congress of Electronics, Robotics and Automotive Mechanics, 2007.

C. Lee and Y. Xu, "Online, Interactive Learning of Gestures for Human / Robot Interfaces," IEEE International Conference on Robotics and Automation, April 1996.

Virtual Technologies Inc. "CyberGlove Brochure." Internet : http://www.mindflux.com.au/products/vti/brochure99.pdf.

DGTech Engineering Solutions. "VHand DG5 Glove Website." Internet : http://www.dg-tech.it/vhand/eng/.

Institute of Personal Robots in Education. "IPRE Wiki." Internet : http://wiki.roboteducation.org/Main_Page.