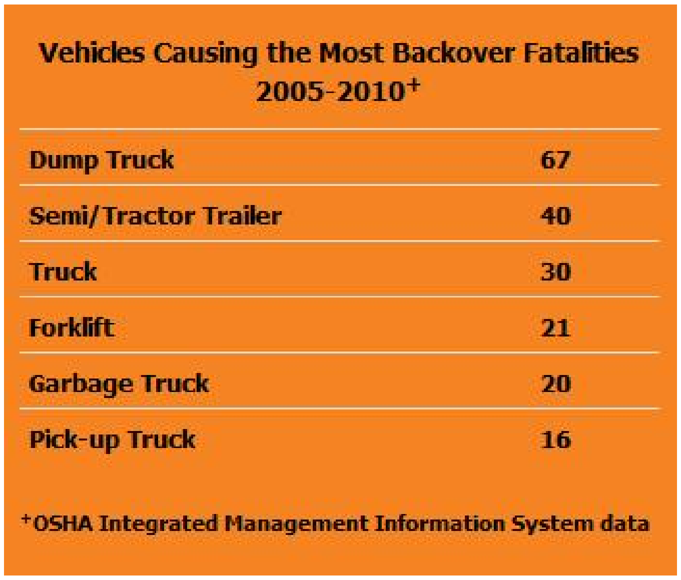
# E-μ Armband

Christopher Allum, CSE & EE, Shehzeen S. Hussain, EE, and Jeffrey A. Maloney, EE

***Abstract*—Miscommunication can quickly lead to accidents resulting in death or injury. The E-μ armband is a wearable device which serves as a supplemental interface between two people, with the purpose of reducing communication errors. As a test case, we assist in navigating large vehicles through difficult or heavily populated terrain. The device interprets a gesture by analyzing muscle activity and motion of the wearer's arm on an embedded processor. Then, a voice command associated with the gesture is sent wirelessly over a Bluetooth connection to a speaker in the driver's car.**

## I. Introduction

IN corporate and military settings, large vehicles must be navigated through both tight areas and through areas with heavy foot traffic. This leads to accidents which often cause injury and death. According to the Department of Labor, nearly two hundred people were killed by large vehicles backing up between the years of 2005 and 2010 [1]. Figure 1 shows a table of how these deaths were distributed between large commercial vehicles.



Vehicles Causing the Most Backover Fatalities 2005-2010⁺

| | |
|---|---|
| Dump Truck | 67 |
| Semi/Tractor Trailer | 40 |
| Truck | 30 |
| Forklift | 21 |
| Garbage Truck | 20 |
| Pick-up Truck | 16 |

⁺OSHA Integrated Management Information System data

Fig. 1. Deaths from various vehicle backups from 2005 to 2010 [1]

The Texas Department of Insurance suggest using a spotter to guide the driver with hand signals [2], and the Armed Forces deploy a ground guide as a standard operating procedure while driving through certain high risk zones [14]. Despite these practices, mistakes are still made.

In order to reduce personal injuries, deaths, and damage to property, we propose a system that would reinforce these existing hand signals via in-cab audio communications. Our gesture recognition system will be designed to be generic enough that it could be used in other applications as well.

C. Allum from East Longmeadow, Ma (e-mail: callum@umass.edu).
S. Hussain, from Bangladesh (e-mail: shehzeen@umass.edu).
J. A. Maloney from Waltham, Ma (e-mail: jamalone@umass.edu).

## II. Design

TABLE I
DESIGN SPECIFICATIONS

| Specification | Target Value | Achieved Value |
|---|---|---|
| Weight | < 3 kg | 1.5 kg |
| Height | < 12cm | 2 cm |
| Length | < 5 cm | 40 cm |
| Battery Life | > 3 hours | > 8 hours |
| Range | > 10m | 10 m |
| Hardware Production Cost | < 80$ | 78 $ |

### *A. Overview*

By reinforcing the communication between the driver and the ground guide, we can avoid injuries, and reduce the costs of preventable accidents. We aim to build a system that will not require additional training, and can seamlessly be implemented into existing practices.

To monitor the arm signals used by guide personnel, we will use an inertial measurement unit (IMU) and electromyography (EMG) sensors. The IMU senses with accelerometers and gyroscopes, and its data can be used to determine movement. The EMG sensors read electrical muscular activity at the surface of the skin. By utilizing both types of sensors, we predict we will be able to accurately recognize the motion of the arm and the movements of the wrist and hand.

Once this data is analyzed and a pattern is recognized, our system will select a voice command from an existing library, and transmit a character corresponding to that command from the guide to a speaker inside the cab using Bluetooth communications. An additional speaker can be placed on the system with the guide, so they can monitor the functionality of the system.

Alternative solutions include using handheld radios to transmit this data, or using radio frequency identification (RFID) tags on ground personnel that interact with sensors on the vehicle. The E- μ is preferable to handheld radios because it does not require existing hand signals to be altered. Similarly, RFID tags fall short of preventing vehicle backups involving multiple vehicles, or other physical obstacles. RFID tags also would not be worn by non-permanent personnel. Our system still involves a human element in case of hardware failure.

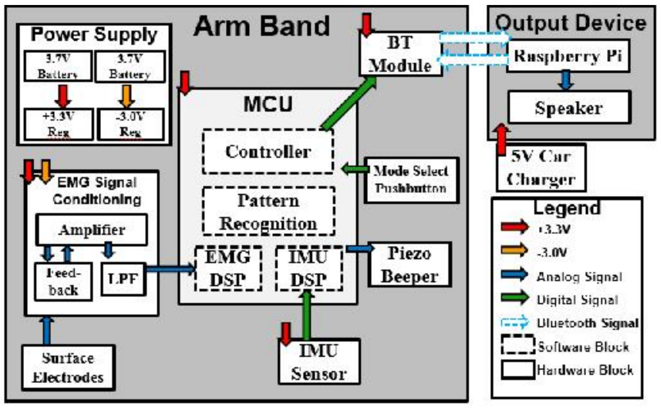


Fig. 2. System level block diagram of the E-u

Figure 2 shows the system level block diagram of the E-µ armband. The armband will be powered by two rechargeable batteries used as a bipolar supply. A 3.3V regulator will be used to power the microcontroller unit (MCU). The system will receive input from the IMU and EMG sensors. After some analog processing the EMG signal will be passed to the MCU for digital processing. After the EMG and IMU signals undergo their respective digital signal processing (DSP) stages, information will be passed to the pattern recognition subsystem. This block will utilize the k-nearest neighbors (KNN) algorithm to determine if a specific gesture is being performed. The controller will then interpret the output of the pattern recognition subsystem and send a character over Bluetooth to a Raspberry Pi computer within the vehicle. The Raspberry Pi will then transmit audio to a commercial speaker. The gestures we intend to use and their anticipated sensor dependencies are shown in Table 2.

TABLE II
GESTURE RECOGNITION

| Gesture | Motion | Primary Dependence |
|---|---|---|
| Stop | Wrist Extension | EMG |
| Slow Down | Repeated Wrist Flexion & Extension | EMG&IMU |
| Forward | Arm Flexion & Extension (Palm Backward) | IMU |
| Reverse | Arm Flexion & Extension (Palm Forward) | IMU |
| Left | Repeated Lateral Motion (Palm Left) | IMU |
| Right | Repeated Lateral Motion (Palm Right) | IMU |

## B. *Electromyography Circuitry*

The purpose of the EMG circuit is to take the raw data from the surface of the user’s skin and to amplify it into a usable signal. Initially, the electrical signals read at the electrode are too weak to be used for meaningful data. These signals are taken from two muscles on the forearm. As such, there are two channels for the EMG circuitry. Before any processing can be done, we must amplify these signals from a few millivolts to a few volts. Also, filtering the signal is necessary, as the only meaningful content resides in the frequency band 60-500Hz [3].

Two electrodes are placed on each muscle to feed a differential signal into a two stage instrumentation amplifier. Typically, the first stage amplification is set with a gain resistor, $R_g$. The gain on this stage is inversely proportional to $R_g$, and for the chosen IC, the Texas Instruments INA2128UA [4], can be set as high as 80dB. The Common Mode gain remains at 1 regardless of what size resistor used. The second stage has unity gain by default. Such a configurations was not possible for our application, however. The EMG signals we were after were being drowned out by other electrical signals at the surface of the skin.

The electrodes are placed in close proximity along the axis of the muscle. The distance between two electrodes was set to be about 2cm. Even at this close range, DC voltage differences are present on the arm, and can sometimes be greater than 20mV, which far exceeds the amplitude of the desired signals. This gave an undesired DC component to the output of the amplifier, and also limited the gain by pushing the amplifier into saturation if $R_g$ was set too small. Furthermore, the DC component was unpredictable, and often changed with the position of the arm.

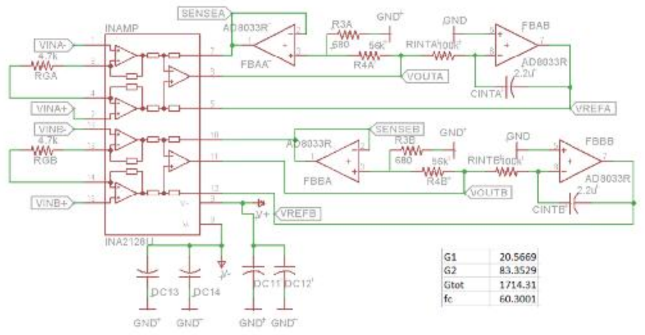


Fig. 3. EMG amplifier circuit diagram

This problem was addressed by the addition of two feedback networks, shown in figure 3. First, a feedback network was added to remove the DC component from the output of the amplifier. This network is a first order active low-pass filter. This feeds the DC component back into the IC on the reference pin, effectively making the instrumentation amp function like a high-pass filter. This was simulated with LT spice and shown to have a cutoff frequency of 60Hz. The second feedback network addressed the clipping problem. Because the gain on the first stage is limited, additional gain needed to be added on the second stage. This is accomplished with a buffered voltage divider feeding the sense pin on either channel. To avoid saturation, the first stage gain is set to 26dB. The second stage gain is 38dB, giving the amplifier a total gain of 64dB.

The anti-aliasing filter design was added in order to filter out noise that could corrupt the meaningful data while sampling, as discussed in the signals and systems analysis class, ECE 313. This was designed to have a cutoff frequency of 560Hz, which implies the sampling rate should be at least

1120 samples per second. The ADC on the MCU can far exceed that rate. In practice, the signals were sampled at 9kHz. The filter, shown in figure 4, is a Sallen-Key topology filter. This is a second order filter and was designed to have a Q-factor of 0.707, giving the magnitude response the shape of a Butterworth filter.

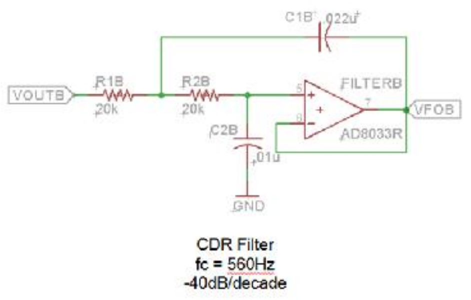


Fig. 4. Anti-aliasing Filter

To test this system, a commercially available EMG sensor was used as a known working system. The sensor used was the Pololu V3 Muscle Sensor [5]. Electrodes were placed on the forearm of one subject, and two different hand motions were performed. The hand motions were done once with the designed system, and again with the Polulu sensor. Comparing outputs from both sensors in real time using an oscilloscope resulted in similar waveforms with slight variations in noise and amplitude.

The output was finally AC coupled to the ADC of the microcontroller. This was done so that a 1.5V DC offset could be applied to place the signal in the middle of the ADC range. The final result was two amplified EMG channels ranging from 0.15V to 2.85V

### C. *Electromyography Digital Signal Processing*

EMG signals are composed of action potentials fired by muscle fibers when performing certain gestures. They are acquired using surface electrodes placed on the flexor muscles of the forearm. Multiple approaches have been suggested to characterize hand movements using both the time and frequency domain representations of EMG signals. This is because for a particular hand gesture, EMG signals provide distinctive features in both the time and frequency domain. These features are extracted and compared for classification of samples and building a gesture library. Papers have identified features such as standard deviation, mean absolute value, number of zero line crossings, mod frequency, waveform length and variance of EMG useful for classifying gestures [10]. EMG signals were initially processed in MATLAB to understand key features that would be useful for characterizing the gestures. The final version of signal processing is done in real time with the microprocessor on two sets of data from right and left channels obtained using two pairs of surface electrodes on the forearm muscles. During real time signal processing, feature extraction from time and frequency domain is performed in two modes, namely the training and run mode, and these features are used in the pattern recognition subsystem. In training mode, the EMG features extracted were used to build a library to characterize each of the gestures.

Initially to collect EMG signal samples, we observed the data with an oscilloscope and then stored the data on a computer in '.csv' format. A parser was used to format the stored data in MATLAB before it was ready for processing.

Typically a frequency range of 20-500Hz is used for frequency domain analysis of the EMG signals [9] as shown in Figure 5. This establishes a cutoff frequency of 500Hz and therefore sets our sampling frequency to the Nyquist rate of 1000Hz. The next step is Fast Fourier Transform (FFT) on the EMG signal and extraction of the desired features. FFT produces an N point Discrete Fourier Transform of the signal and provides us with a single frequency spectrum for our EMG signal. In order to confirm that our FFT produced correct results we performed this analysis on a sinusoidal input from the function generator. Parameters such as the fundamental frequency, mean and standard deviation of the sinusoid input matched our expected result. Having confirmed that our MATLAB program works accurately and sampled at a frequency of 1000Hz, we performed FFT on EMG signals obtained for six different gestures. Signal processing techniques we used were mostly an application of concepts explored in ECE 563 "Introduction to Communications and Signal Processing". The results from the MATLAB program helped develop an idea of signal features in both the time and frequency domain. Our next steps lead to real time processing of EMG signals using the microprocessor.

Our software performs real time signal processing on signals obtained from the armband for six different gestures, in both the time and frequency domain. Incoming analog signals are averaged to produce digitized values. Moving windows are used as a digital smoothing technique in order to reduce noise in signals before performing real time FFT on them. In the time domain, the extracted features include mean amplitude, root mean square of amplitude and number of zero crossings. The mean amplitude is used to capture signal strength and root mean square of amplitude is used to quantify the power of the signal, calculated by squaring each data point, summing the squares, dividing the sum by the number of observations, and taking the square root of result. Number of zero crossings is determined by counting the number of times the amplitude of the signal crosses the pseudo zero line. A DC offset is added to the analog EMG signal being read digitally, to avoid loss of EMG data with negative amplitudes. A more active muscle will generate more action potentials causing more zero crossings in the signal. In the frequency domain we extract features such as mod frequency, mean amplitude and root mean square of amplitude. Some processing was performed to achieve the variance of the signal, however this proved to be less useful when characterizing hand gestures. After performing FFT to break the EMG signal into its frequency components, mod frequencies are used to detect the frequencies at which the highest amplitude occurs.

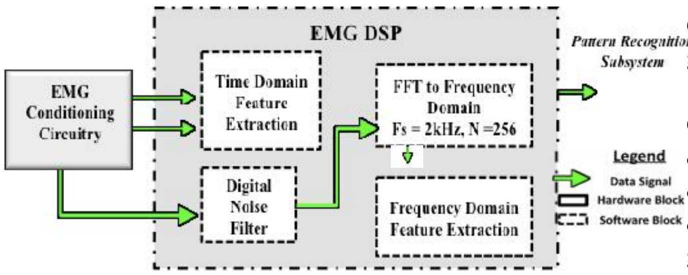

EMG DSP
EMG Conditioning Circuitry
Time Domain Feature Extraction
FFT to Frequency Domain Fs = 2kHz, N =256
Digital Noise Filter
Frequency Domain Feature Extraction
Pattern Recognition Subsystem
Legend
Data Signal
Hardware Block
Software Block

Fig. 5. Overview of the digital signal processing subsystem

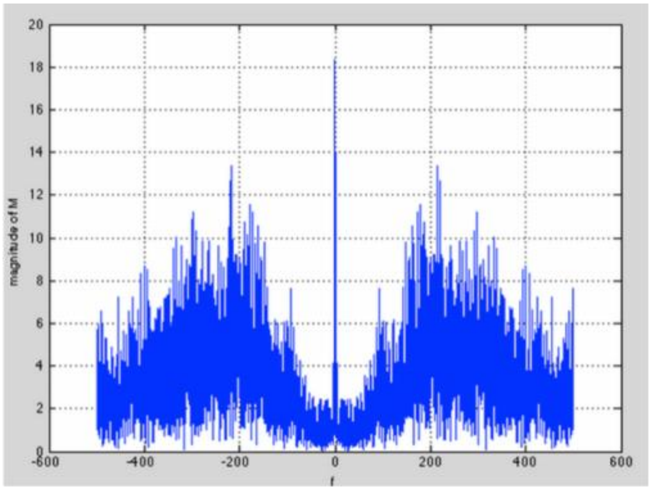


Fig: 6. MATLAB unprocessed signal FFT

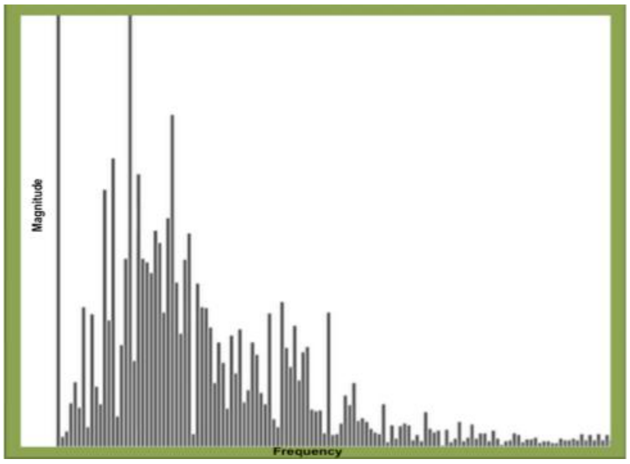


Fig: 7. Microprocessor processed signal FFT

## D. Inertial Measurement Unit

The purpose of the inertial measurement unit is to detect arm movements of the ground guide. This is accomplished through both the digital sensor MPU-6050 and through associated software. The MPU-6050 contains a three axis accelerometer and a three axis gyroscope. Since the sensor is digital, it is integrated into the device through $I^2C$ compatible pins on the microcontroller. Each reading from the sensor is interpreted as six column vector including roll, pitch, yaw, and acceleration in the x, y, and z directions.

There are two components in the digital processing of the data from the IMU. The first component is a digital low pass filter. This acts as a way to make the sensors less sensitive to very small movements. The purpose of this is to get more consistent data without losing characteristic information of the gestures.

The second component of the digital processing works to counter the effects of gravity which initially distort the acceleration readings. The act of holding up one's arm in place accelerates the sensor upward against gravity and this acceleration changes as the arm rotates [7]. However, since gravity is a constant acceleration, this is avoided by using the MCU to determine the derivative of acceleration. While not a perfect work-around, the resulting readings strongly mitigate the changes in acceleration from gravity during rotation and greatly emphasize changes acceleration due to the wearer's actions. The units of the processed accelerometer data are in meters per second cubed, or jerks. Figure 4 depicts the IMU DSP process.

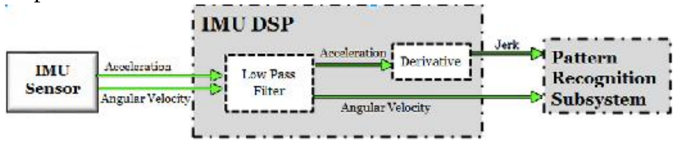


Fig. 8. Digital signal processing for the inertial motion sensor

## E. Pattern Recognition

The purpose of this subsystem is to interpret readings from the sensors and decide when different gestures are being performed. The algorithm we use is a variation of the K-nearest neighbors (KNN) algorithm which was learned both from a class on intelligent system design, ECE 597C, and its textbook "Introduction to Pattern Recognition" [12]. The benchmark for success with the pattern recognition system is the ability to reliably differentiate between six gestures using the data from the sensors with a useable response time.

Two dimensional vectors are used to represent patterns of data that correspond to each motion. Each column is a different property of the motion as determined by the IMU DSP and EMG DSP subsystems. Each row of this two dimensional pattern vector is a unique reading in time.

The input vector is compared against this stored-pattern vector with the result being a score for each gesture based on the similarity of the input. To get this score, the distance between the input and the stored pattern is determined using the Euclidean distance formula:

$$distance(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 \cdots + (p_n - q_n)^2}$$

Where 'n' is the number of features being compared in the data, 'p' is the input vector from the sensor and 'q' is a single row of the stored pattern.

For each row of each pattern, distances are stored relative to a single input, along with a record of which pattern each specific distance measurement belongs. The distances are then sorted from lowest to highest and N of the smallest distances are considered for scoring. If a distance is above a defined threshold, no score is assigned. This allows us to ignore movements of the user which are nothing like any of the gestures. The pattern with the least distance has its score value incremented by one.

In order to smooth out the results after multiple runs we use a moving window on the score values. There is a delay of approximately one second using a window size of 80, but a larger window size means more consistent and accurate results. After fine tuning a window size of 44 of readings was chosen. This value allows for both consistency and viable response time of about a third of a second. Figure 5 depicts an overview of how this subsystem works.

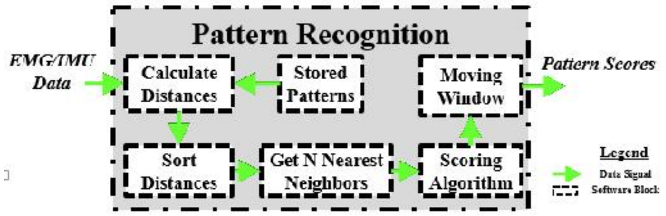


Fig. 9. Overview of the pattern recognition process

The microprocessor we have chosen to do these calculations is a 32 bit ARM Cortex-M4 that operates at 72 MHz [13]. The pattern recognition is the most intensive operation that the microprocessor will have to perform, and its processing speed was more than adequate for our needs.

In order to test that this subsystem worked before joining it with the other subsystems, two pseudo gestures and a set of mock input data were constructed. The mock input data was purposely created to match one pseudo gesture more so than the other. Additionally, the subsystem was tested to work in real time using sensor data from the IMU and proved to be accurate.

In order to use the device properly it is necessary to record your gestures at the start of each use. A training mode was implemented, allowing the system to accommodate for deviation in the user's physical characteristics and deviation in sensor placement between uses. In this mode the user is allotted a fixed amount of time to perform all of the six gestures in order. A series of beeps from a piezoelectric buzzer helps the user know when to switch gestures and when the recording starts. Recording new gestures takes about a minute.

When connected to a computer, the E-u outputs the results of the training to a file so that it can be analyzed. We performed PCA analysis on a recorded sample of the final product and graphed the three most dominant axes. Our PCA graph is included in Figure 6 below. This helped us interpolate details about our gestures. For example, we discovered that "Stop" tended to have the highest cluster density and was most isolated from the others.

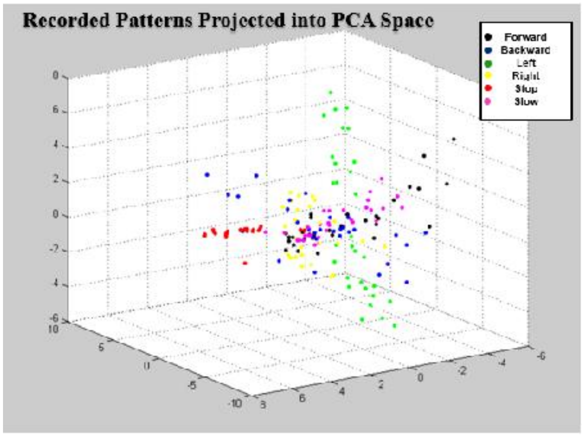
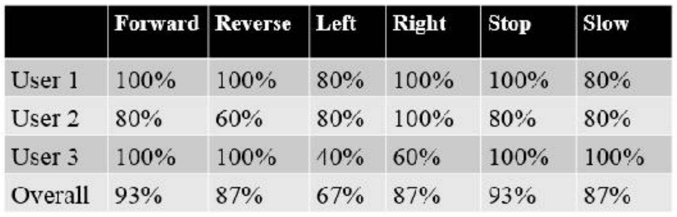


Fig. 10. PCA visualization of recorded gestures after training

After integrating all of the systems, we did some final testing to determine the accuracy of the recognition. Three subjects performed all the gestures multiple times and for one recording, the number of successful gesture attempts was logged. We discovered that success with the product requires a degree of human training, in that the user should be consistent with their definition of a gesture. If they do not perform the gesture similarly to how it was recorded, then it will not be picked up properly. User one in Figure 7 had the most success, while users two and three had less consistent results.



| | Forward | Reverse | Left | Right | Stop | Slow |
|---|---|---|---|---|---|---|
| User 1 | 100% | 100% | 80% | 100% | 100% | 80% |
| User 2 | 80% | 60% | 80% | 100% | 80% | 80% |
| User 3 | 100% | 100% | 40% | 60% | 100% | 100% |
| Overall | 93% | 87% | 67% | 87% | 93% | 87% |

Fig. 11. Results of testing the pattern recognition system.

## F. User Interface

There are three modes of operation of the device. In "Run Mode" the device reads the sensors, compares the readings to stored values, and outputs via Bluetooth. In "Sleep Mode" the device does nothing. This mode is meant to have low power consumption. The "Training Mode" is where the user records their gestures. By pressing the button on the arm band the user can switch between modes. A state diagram depicting the modes and their transitions is shown below:

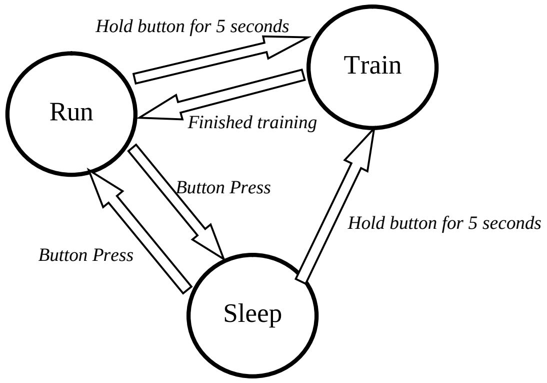


Fig. 11. Mode transition abstraction

The user interacts with the device mostly through a button and a piezoelectric beeper. However, an optional graphical user interface (GUI) can be provided during training for the user to visualize the sensor readings and output of the pattern recognition system. This GUI is shown below in Figure 8.

The GUI was developed in a language called Processing. It received information from the E-u over the same serial port which we used to program the microprocessor. In run mode it displays all of the important sensor data as well as the results of the pattern recognition – including a score for each individual gesture.

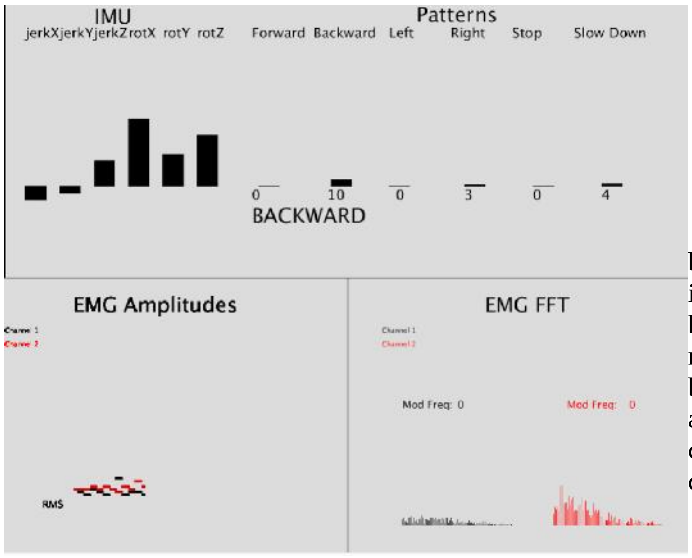


Fig. 12. The optional graphical interface

## G. PCB and Armband Design

The EMG circuitry, microcontroller, and Bluetooth module were mounted together on a PCB. The power, IMU and pushbutton interface were mounted adjacent and wired into connections on the PCB. The PCB was placed midway down the forearm between the IMU and the EMG electrodes as seen in the figure below. Hardware components were sewn into the fabric sleeve, and Velcro straps were used to hold everything firmly in place.

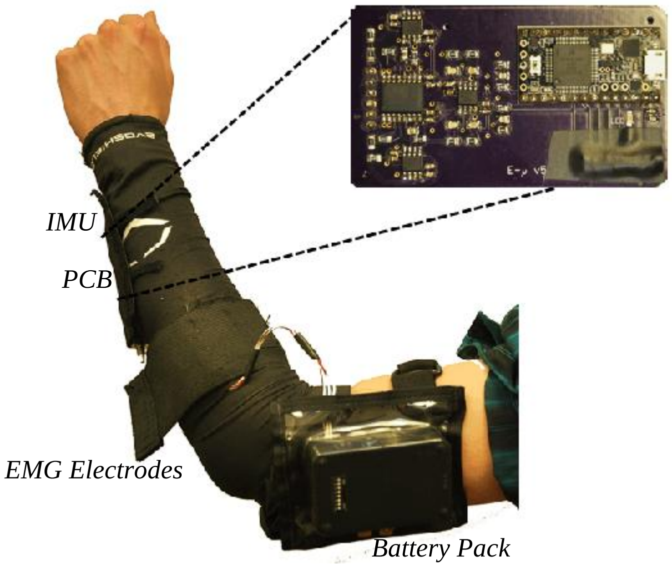


Fig. 13. The final E-u arm band and a close up of the PCB

## H. Power

The entire device is powered by two 3.7 Lithium Ion batteries. A bipolar supply was chosen because the instrumentation amp selected has better performance with a bipolar supply than a single ended one. Linear voltage regulators are used to output +3.3V and -3.0V from the two batteries. Aside from the EMG circuitry, everything on the armband runs off of the 3.3V supply. In testing, the batteries could power the entire system for nearly 9 hours between charges. This far exceeded the initial specification of 3 hours.

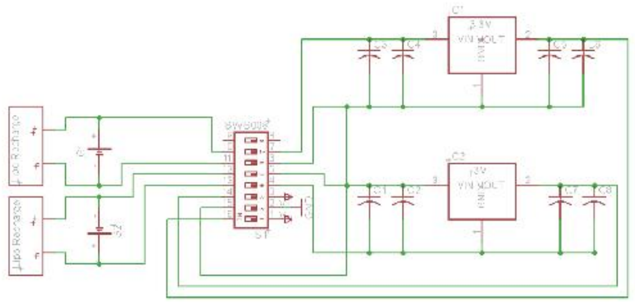


Fig. 14. Voltage regulatory circuit

The power supply was kept separate from the rest of the armband. This is not necessary for future implementations. The electronics used are not large enough for space to be a concern, and overheating was not a problem. Aside from the batteries themselves, the entire power supply could be integrated into the PCB design.

## I. Audio Output Feedback

Our project aims to output audio signal delivered by a speaker inside the car upon command from armband. Vehicle operator receives audio commands inside the vehicle more than ten meters in range from ground guide. Gestures are translated to commands sent wirelessly over Bluetooth link to car driver. The two principle components used in the audio output subsystem are the BlueSMiRF Gold Bluetooth module located on the armband PCB and Raspberry Pi connected to speakers inside the car. A power adapter inside the car is used to power the Raspberry Pi. The microprocessor on the armband communicates commands to receiver inside car via Bluetooth TX RX. Raspberry Pi connected to car charger is set to pair with armband upon request. A Python script running on the raspberry Pi is programmed to play one of the six sound files stored in the SD card in order to voice each of the gestures to the driver, upon command from the guide's armband.

## III. Project Management

The EMG Circuit successfully amplifies and filters the raw signal from the surface of the skin for processing. The non-stable DC has been resolved. The EMG Digital Processing currently reads in EMG data, further filters the data, and performs frequency domain analysis on the data, and accomplishes this all in real time. Waveforms are characterized by their mod frequency, amplitude, and RMS energy. The IMU system currently sends data to the pattern recognition system in the form of a six column vector containing three dimensions of jerk, roll, pitch, and yaw. The pattern recognition system is able to use EMG and IMU data to distinguish between six distinct patterns.

Each subsystem is working, and all subsystems have been successfully integrated together. Further improvements could be made to achieve a higher level of success and consistency. The physical placement of the armband on the user seems to be the largest factor in inconsistencies between users. If the electrodes do not form good connections, noise on the EMG lines renders them useless. The IMU is also able to move over the course of the day. This results in degraded performance after about 30 minutes, and the user is forced to rerecord. The physical layout of the arm band could be improved by integrating the power system onto the PCB and using a hard enclosure to ensure the device does not move relative to the user. This would also allow the user to don the armband more quickly, and with less difficulty. Furthermore, changes to the software could result in more accuracy. Among other things, the group has discussed using PCA or Bayesian matching algorithm.

The E-µ armband currently is robust enough that it could be implemented with several other output devices. The Bluetooth transceiver could be used to talk to a laptop, smartphone, or a slew of other devices. By allowing users to record any set of unique gestures, and by outputting something as general as a character, the armband allows for an output device to make whatever type of decision that suits it.

## IV. Conclusion

The E-µ armband is able to read data from two different types of sensors and accurately determine if one gesture out of a set is being performed. Custom electronics had to be designed to reliably isolate EMG muscle signals. Challenges in digital signal processing had to be met. Data had to be combed through to determine which characteristics made the best predictors, and all the hardware had to be fused into a single wearable device.

Certain inconsistencies between users were addressed by the implementation of a training mode. Each user can record a set of gestures that will be unique to their anatomy and different sensor placements. This can also be customized to suit different drivers if they prefer one type of signal over another.

By allowing users to define their own gestures, the E-µ can interface with countless different devices with Bluetooth connections. The information being conveyed is not limited to "left", "right", "forward", and so on. A person using the E-µ can make their own rules. They can decide what gestures to perform, and they can decide how that information can be used.

## References

[1] United States Department of Labor, “Preventing Backovers,” Occupational Safety & Health Administration, 2011 [Online], Available at https://www.osha.gov/doc/topics/backover/. [Accessed Nov. 20, 2014].

[2] Texas Department of Insurance, “Vehicle Backing Safety Factsheet,” Division of Workers’ Compensation, 2013 [Online], Avalable at http://www.tdi.texas.gov/pubs/videoresource/fsvehiclebackin.pdf. [Accessed Dec. 1, 2014].

[3] Delsys, “Fundamental Concepts in EMG Signal Aquisition,” Delsys Inc. 2003 [Online], Available http://www.delsys.com/. [Accessed Oct. 23, 2014].

[4] Texas Instruments, “Micropower Instrumentation Amplifiers, Single and Dual Versions,” INA2126 Datasheet, January 1996 [Revised August 2005].

[5] Advancer Technologies, “Three-Lead Differential Muscle Electromyography Sensor for Microcontroller Applications,” AdvancerTechnologies.com, Feb 2013. Available at http://www.pololu.com/product/2726/resources.

[6] Analog Devices, Appl. Note 649, pp. 21-24.

[7] Veresano, Fabio. “A Guide to Using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications.” Starlino Electronics. N.p., n.d. Web.

[8] Analog Devices, “A Designer’s Guide to Instrumentation Amplifiers,” Analog Devices Inc. 2006 [Online], Available http://www.analog.com/. [Accessed Oct. 18, 2014].

[9] De Luca, Carlo J. "The Use of Surface Electromyography in Biomechanics."Journal of Applied Biomechanics (1997): 135-63. Human Kinetics Publisher Inc. Web. 5 Sept. 2014.

[10] Artemiadis, P.K., and K.J. Kyriakopoulos. “EMG-Based Control Of a Robot Arm Using Low-Dimensional Embeddings.” Robotics, IEEE Transactions On 26.2 (2010) : 393-398. © 2010 IEEE

[11] Analog Devices, Appl. Note 649, pp. 21-24.

[12] Theodoridis, Sergios, Aggelos Pikrakis, Konstantinos Koutroumbas, and Dionisis Cavouras. Introduction to Pattern Recognition: A MATLAB Approach. Burlington, MA: Academic, 2010. Print.

[13] Semiconductor, Inc. Freescale. "K20P64M72SF1." K20 Sub-Family Web.<http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Dev/Arduino/Boards/K20P64M72SF1.pdf>.

[14] U.S. Marine Corps Technical Manual, “TM 11240-15/3F Motor Vehicle Licensing Official’s Manual,” U.S. Marine Corps. Jan 2009 [Online] Available at http://www.med.navy.mil/sites/nmcphc/Documents/.