Integrating Neuromuscular and Cyber Systems for Neural Control of Artificial Legs

He Huang, Yan (Lindsay) Sun, Qing Yang, Fan Zhang, Xiaorong Zhang, Yuhong Liu, Jin Ren, Fabian Sierra
University of Rhode Island
Department of Electrical, Computer, and Biomedical Engineering, Kingston, RI, 02881
{huang, yansun, qyang}@ele.uri.edu

ABSTRACT

This paper presents a design and implementation of a cyber-physical system (CPS) for neurally controlled artificial legs. The key to the new CPS system is the neural-machine interface (NMI) that uses an embedded computer to collect and interpret electromyographic (EMG) signals from a physical system that is a leg amputee. A new deciphering algorithm, composed of an EMG pattern classifier and finite state machine (FSM), was developed to identify the user's intended lower limb movements. To deal with environmental uncertainty, a trust management mechanism was designed to handle unexpected sensor failures and signal disturbances. Integrating the neural deciphering algorithm with the trust management mechanism resulted in a highly accurate and reliable software system for neural control of artificial legs. The software was then embedded in a newly designed hardware platform based on an embedded microcontroller and a graphic processing unit (GPU) to form a complete NMI for real time testing. Our preliminary experiment on a human subject demonstrated the feasibility of our designed real-time neural-machine interface for artificial legs.

Categories and Subject Descriptors

J.3. [**Computer Applications**]: Life and Medical Sciences-*health*

Keywords

Neural-machine interface, prosthetics, high-performance computer, trust management.

1. INTRODUCTION

Rapid advancement of computer technology has completely changed the way of our life in terms of how we work, learn, conduct business, manufacture, and play. High speed and real time embedded computing systems that are widely available as commodity have made it possible to automate manufacturing, transportation, robotic control, wired/wireless communication, healthcare systems, and more. In particular, tight coupling of cyber systems and biomedical systems has drawn great interests in the research community recently. One prominent example is computerized prosthetic legs, in which motion and force sensors and a microcontroller embedded in the prosthesis form a close-loop control and allow the user to produce natural gait patterns [1-2]. However, the function of such a computerized prosthesis is still limited due to the lack of neural control. The primitive prosthesis control is based entirely on mechanical sensing without the knowledge of user intent. Users have to "tell" the prosthesis the intended activity manually or using body motion, which is cumbersome and does not allow smooth task transitions.

To allow the user to control the artificial leg as if it is his/her own limb, a seamless integration of human neuromuscular system and computer system is essential. This integration leads to a cyber-physical system (CPS), in which a complex physical system (i.e. neuromuscular control system of a leg amputee) is monitored and deciphered in real time by a cyber system. The key to the success of such integration is the neural-machine interface (NMI) that senses neural signals from leg amputees, interprets such signals, and makes accurate decisions for prostheses control. Electromyographic (EMG) signals represent neuromuscular activity and are effective neural signals for expressing movement intent [3]. Although EMG-based NMI has been tested for artificial arms [4-5], no EMG-controlled prosthetic leg is available, and published studies in this area are very limited. This is because inevitable challenges in both hardware/software design of embedded computer systems (cyber) and accurate interpretation of neuromuscular system (physical) make the NMI design for neural control of lower limb prostheses difficult.

1) In human physiological system, EMG signals recorded from leg muscles during dynamic movements are highly non-stationary. Accurate decoding of user intent from such signals requires dynamic signal processing strategies [6].

2) Accuracy in identifying the user's intended lower limb movement is essential. A 90% accuracy rate might be acceptable for control of artificial arms, but it may result in one stumble out of ten steps, which is obviously inadequate to ensure the patient's safety in prosthesis use.

3) There might not be enough EMG recording sites available in leg amputees [6]. Design of an algorithm to maximally extracting neural information is necessary.

4) Environmental uncertainty, such as perspiration, temperature change, and movement between the residual limb and prosthetic socket may cause unexpected sensor failure, influence the recorded EMG signals, and reduce the trustworthiness of the NMI. It is critical to develop a reliable and trustworthy NMI for safe use of prosthetic legs.

5) Implementing the neural interfacing algorithms on an embedded computer system is essential to make the EMG-based NMIs practical and available to patients with leg

amputations. The speed of the embedded system must be adequate because any delayed decision-making from the NMI also introduces instability and unsafe use of prostheses. Streaming and storing multiple sensor data, deciphering user intent, and running sensor monitoring algorithms at the same time superimpose a great challenge to the design of an embedded system for the NMI of artificial legs.

This paper presents the first real-time, EMG-based neural-machine interface for artificial legs. Whereas existing work [6] [7] provided some piecemeal solutions to certain aspects of the above challenges, this new design tightly integrated the human neuromuscular system with an embedded system and addressed the difficulties in the cyber-physical system described above for accurate and reliable user intent identification.

The neural interfacing algorithm takes EMG inputs from multiple EMG electrodes, decodes user intended lower limb movements, and monitors sensor behaviors based on trust models. An improved EMG pattern recognition (PR) algorithm together with a finite state machine (FSM) effectively tackle the difficult challenges, resulting from non-stationary EMG signals of leg muscles, for accurately deciphering user intent. The neural decoding algorithm consists of two phases: offline training and online testing. To ensure the trustworthiness of NMI under uncertain environment, a trust management (TM) module was designed to examine the changes of the EMG signals, estimate the trust level of individual sensors, and determine the overall trust level of the NMI. The trust information can be used to reduce the impact of untrustworthy sensors on the system performance.

To realize the NMI to be carried by leg amputees, we designed new embedded hardware architecture for implementing the designed algorithms. The two key requirements for the hardware architecture were high speed processing of training process and real time processing of interfacing algorithm. To meet these requirements, the newly designed embedded architecture consisted of an embedded microcontroller, a flash memory, and a graphic processing unit (GPU). The embedded microcontroller provided necessary interfaces for AD/DA signal conversion and processing and computation power needed for real time control. We implemented our control algorithm on the bare machine with our own memory and IO managements without using existing OS to avoid any unpredictability and variable delays. The flash memory was used to store training data. EMG PR training process involved intensive signal processing and numerical computations, which needs to be done periodically when the system trust value is low. Such computations can be done efficiently using modern GPUs that provide supercomputing performance with very low cost. New parallel algorithms specifically tailored to the multi-core GPU were developed exploiting memory hierarchy and multithreading of the GPU. Substantial speedups of the GPU for training process were achieved making the classifier training time tolerable in practice.

Finally, to prove the NMI design concept, we tested the design methods and our first NMI prototype on an able-bodied subject to recognize his intent for sitting and standing, two basic but difficult tasks for patients with transfemoral amputations due to the lack of power from the knee joint. The system performance was quantified and evaluated.

This paper made the following contributions:

- Design of the first architecture of a NMI for artificial legs;
- Novel design of EMG pattern classification combined with FSM for decoding the user's intended lower limb movements for neural control of artificial legs;
- Development of abnormal detection and trust evaluation models to solve the problem of uncertainty in a biomedical application.
- Design and offline test of a trustworthy sensor interface on realistic EMG data collected from a human subject.
- Optimal neural interfacing algorithm implementation specifically tailored to the MPC5566 embedded system and GPU architecture for real time operation;
- Demonstration of the feasibility of designed neural interfacing algorithm for deciphering user intent by real-time prototype testing on a human subject.

This paper is organized as follows. The next section presents the system architecture and design of algorithms and embedded system. The third section describes the experimental settings of our first NMI prototype on an able-bodied subject. The results of the study are demonstrated in the section 4, followed by related work in the section 5 and conclusions in the section 6.

2. SYSTEM ARCHITECTURES

2.1 System Architecture

The architecture of neural-machine interface is demonstrated in Figure 1. Multiple channels of EMG signals are the system inputs. EMG signals are preprocessed and segmented by sliding analysis windows. EMG features that characterize individual EMG signals are extracted for each analysis window. The system consists of two major pathways: one path for classifying user movement intent and the other for sensor trust evaluation (the dashed blocks in Figure 1). To identify user intent, EMG features of individual channels are concatenated into one feature vector. The goal of pattern recognition is to discriminate among desired classes of limb movement based on the assumption that patterns of EMG features at each location is repeatable for a given motion but different between motions [5]. The output decision stream of EMG pattern classifier is further processed to eliminate erroneous task transitions. In the path for sensor trust evaluation, the behaviors of individual sensors are closely monitored by abnormal detectors. A trust manager evaluates the trust level of each sensor and then adjusts the operation of the classifier for reliable EMG pattern recognition.

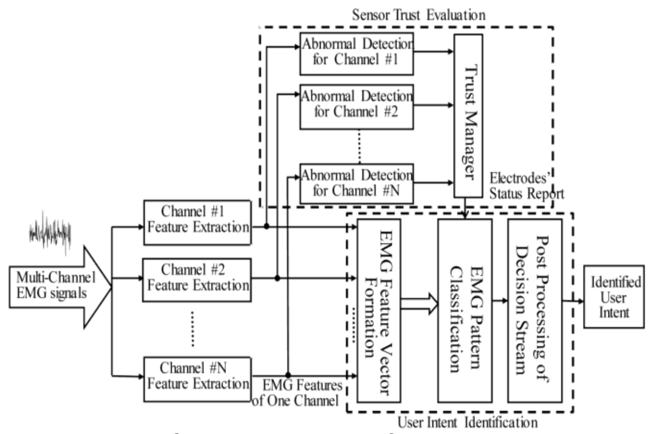
Sensor Trust Evaluation
Abnormal Detection for Channel #1
Abnormal Detection for Channel #2
Abnormal Detection for Channel #N
Trust Manager
Electrodes' Status Report
Multi-Channel EMG signals
Channel #1 Feature Extraction
Channel #2 Feature Extraction
Channel #N Feature Extraction
EMG Features of One Channel
EMG Feature Vector Formation
EMG Pattern Classification
Post Processing of Decision Stream
Identified User Intent
User Intent Identification

Figure 1. Software architecture of EMG-based neural-machine interface for artificial legs.

The hardware architecture of the NMI (Figure 2) for artificial legs consists of seven function blocks: EMG electrodes, amplifier circuits, analog-to-digital converter (ADC), flash memory, RAM, GPU, and embedded controller. The EMG electrodes collect the raw EMG data from human muscles. The amplifier circuits are necessary to make the polarity, amplitude range, and signal type – whether differential or single-ended – of EMG signals compatible with the input requirements of ADCs. The outputs of the amplifier circuits are then converted to digital format by the ADCs and stored in a flash memory or a RAM. The embedded hardware works in two modes: offline training and real time testing. In the training mode, the digital EMG data are stored in the flash memory. The PR algorithm for training phase includes complex signal processing and numerical computations for a large amount of data. These computations are done efficiently in a high performance GPU. The parameters of trained classifier are stored in the flash memory. In the testing mode, the ADCs sample the EMG signals continuously, and the converted digital data are stored in a RAM of the embedded microcontroller. The microcontroller then runs the trained classifier in the testing phase and makes decisions of user intent in real time.

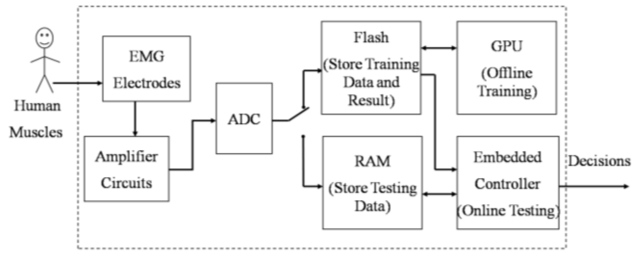

Figure 2. Hardware architecture of designed neural-machine interface.

2.2 Identification of User Intent

Decoding intended movements using leg EMGs recorded from transfemoral amputees is challenging because (1) the recordable EMG sites are limited due to the limb loss and (2) the EMGs are highly non-stationary. A dynamic EMG pattern classification strategy was adopted to address these challenges in this study. Additionally, post-processing methods on the decision stream were used for improved system accuracy.

EMG Signals: EMG signals recorded from gluteal and thigh muscles were considered because these muscles are still available for patients with transfemoral amputations.

EMG Features: Four time-domain (TD) features [8] (the mean absolute value, the number of zero-crossings, the waveform length, and the number of slope sign changes) were proposed for real-time operation because of their low computational complexity [5] compared to frequency or time-frequency domain features. The detailed equation and description of these four TD features can be found in [8].

EMG Pattern Classification: Various classification methods, such as linear discriminant analysis (LDA) [8], multilayer perceptron [9], Fuzzy logic [10], and artificial neural network [6], have been applied to EMG PR. Due to the computation simplicity, LDA has been widely applied to real time control of upper limb prostheses [5, 11]. The idea of discriminant analysis is to classify the observed data to the movement class in which the posteriori probability $P(C_g \mid \bar{f}\ )$ can be maximized. Cg ($g\in$[1, G]) denotes the movement classes; $\bar{f}$ is the feature vector in one analysis window. The posteriori probability is the probability of class Cg given the observed feature vector $\bar{f}$ and can be expressed as

$$P(C_g \mid \bar{f}\ ) = \frac{P(\bar{f} \mid C_g)P(C_g)}{P(\bar{f})} \quad (1),$$

where $P(C_g)$ is the priori possibility, $P(\bar{f} \mid C_g)$ is the likelihood, and $P(\bar{f})$ is the possibility of observed feature vector $\bar{f}$. Given movement class C_g, the observed feature vectors have a multivariate normal (MVN) distribution, i.e. $P\ (\bar{f} \mid C_g) \sim MVN(\mu_g, \Sigma_g)$, where μ_g is the mean vector and Σ_g is the covariance matrix of the class C_g. Additionally, the priori possibility is assumed to be equivalent for each movement class, and every class shared a common covariance, i.e. $\Sigma_g = \Sigma$. Hence, the maximization of posteriori possibility in (1) becomes

$$\tilde{C}_g = \arg\max_{C_g} \{\bar{f}^T \Sigma^{-1} \mu_g - \frac{1}{2} \mu_g{}^T \Sigma^{-1} \mu_g\} \quad (2).$$

$$d_{C_g} = \bar{f}^T \Sigma^{-1} \mu_g - \frac{1}{2} \mu_g{}^T \Sigma^{-1} \mu_g \quad (3)$$

is defined as the linear discriminant function.

In the offline training μ_g and Σ were estimated by feature vectors calculated from a large amount of training data and were stored in the flash memory.

$$\tilde{\mu}_g = \frac{1}{K_g} \sum_{k=1}^{K_g} \bar{f}_{C_g,k} \quad \text{and}$$

$$\tilde{\Sigma} = \frac{1}{G} \sum_{g=1}^{G} \frac{1}{K_g - 1} (F_g - M_g)(F_g - M_g)^T$$

where Kg is the number of observations in class C_g; $\bar{f}_{C_g,k}$ is the k observed feature vector in class Cg; Fg is the feature matrix $F_g = [\ \bar{f}_{C_g,1}, \bar{f}_{C_g,2}, ... \bar{f}_{C_g,k}, ..., \bar{f}_{C_g,K_g}]$; Mg is the mean matrix $M_g = [\tilde{\mu}_g, \tilde{\mu}_g, ... \tilde{\mu}_g]$ that has the same number of columns as in Fg. Therefore, the parameters in the linear discriminant function (3) were known, i.e.

$$\tilde{d}_{C_g} = \bar{f}^T \tilde{\Sigma}^{-1} \tilde{\mu}_g - \frac{1}{2} \tilde{\mu}_g{}^T \tilde{\Sigma}^{-1} \tilde{\mu}_g \quad (4).$$

In the real time testing, the observed feature $\bar{f}$ derived from each analysis window was fed to the classifier to calculate $\tilde{d}_{C_g}$ in (4) for each movement class and was classified into a specific class $\tilde{C}_g$ that satisfied

$$\tilde{C}_g = \arg\max_{C_g} \{\tilde{d}_{C_g}\}, C_g \in \{C_1, C_2, ..., C_G\}\ .$$

Dynamic Pattern Classification Strategy: When EMG signals are non-stationary, the EMG features across time show large variation within the same task mode (class), which results in overlaps of features among classes and therefore low accuracy for PR [6]. By assuming that the pattern of non-stationary EMGs has small variation in a short-time window and EMG patterns are repeatable for each defined short-time phase, we designed a phase-dependent EMG classifier, which was successfully applied to accurately and responsively recognize the user's locomotion modes [6]. For non-locomotion modes such as sitting and standing, the classifier can be built in the movement initiation phase by the same design concept. The structure of such a dynamic design of the classifier can be found elsewhere [6].

Post-processing of Decision Stream: FSM and majority vote were used to eliminate erroneous decisions from the classifier. Finite state machine [12] models a task as a state. An action of task (state) transition can be executed only if a valid transition condition is met. Majority vote [5] simply removes the decision error by smoothing the decision output. Note that these methods can further improve the accuracy of NMI, but may sacrifice the system response.

2.3 Trustworthy Sensor Interface

The NMI for artificial legs must be reliable and trusted by the prosthesis users. To achieve this goal, we designed a trust management module that contains three parts: abnormal detection, trust manager, and decision support.

Abnormal Detection: Detecting abnormality in EMG signals is a challenging task because the events causing sensor malfunctions can be diverse and unexpected. It is difficult to construct the training data that can represent all types of reasons behind sensor malfunctions. Without relying on the training data, we proposed to detect abnormality in EMG signals by a *change detector* that identifies changes in the statistics of EMG signals. Particularly, we focused on the two-sided mean change detector.

Many statistical methods can be used to build the change detector [13]. In this work, we chose the Cumulative Sum (CUSUM) algorithm [14] because it is reliable for detecting small and graduate changes, insensitive to the probabilistic distribution of the underlying signal, and optimal in detection speed [15].

Two-sided CUSUM detector was used in this work [14].

$$S_{hi}(i) = \max(0, S_{hi}(i-1) + x_i - \hat{\mu}_0 - k) \qquad (5)$$

$$S_{lo}(i) = \max(0, S_{lo}(i-1) + \hat{\mu}_0 - k - x_i) \qquad (6)$$

where x_i represents the i^{th} data sample, $\hat{\mu}_0$ is the mean value of data without changes, and k is CUSUM parameter. The smaller the k is, the more sensitive the CUSUM detector is to small changes. Here, we set k as 0.008. In (5) and (6), S_{hi} and S_{lo} are used for detecting the positive and negative changes, respectively. If S_{hi} exceeds a certain threshold, a positive change is detected. If S_{lo} is smaller than a certain threshold, a negative change is detected. The initial values of S_{hi} and S_{lo} were set to 0.

Trust Manager: After the abnormal detector detects the disturbance in an EMG signal, the EMG sensor is either permanently damaged or perfectly recoverable. To evaluate the trust level of the sensor, let p_1 denotes the probability that a sensor behaves normally after one disturbance is detected. Assume all disturbances are independent. The probability that a sensor is still normal after *i* disturbances, denoted by p_i, is $p_i = p_1^{\,i}$. The trust value is computed from the probability value by the entropy-based trust quantification method[16], as

$$T = \begin{cases} 1 - H(p_i), if & 0.5 \le p_i \le 1 \\ H(p_i) - 1, if & 0 \le p_i < 0.5 \end{cases},$$

where T is the trust value and $H(p_i)$ is the entropy calculated as

$$H(p_i) = -p_i \log_2(p_i) - (1 - p_i)\log_2(1 - p_i) \qquad (7)$$

Different p_1 values should be set according to the nature of the disturbance. The larger the p_1 value, the less likely the disturbance can damage the sensor. The calculation of trust is extendable to the case that different disturbances are detected for one sensor. If two disturbances, whose p_1 values are 0.8 and 0.9, respectively, are detected for a sensor, the p_i value in (7) can be replaced by 0.8×0.9.

Decision Making and Report: The trust information is provided to the user intent identification (UII) module to assist trust-based decisions. There are two levels of decisions.

1) Sensor level: When the sensor's trust value drops below a threshold, this sensor is considered as damaged, and its reading is removed from the UII module.
2) System level: After removing the damaged sensors, we can calculate the *system trust* by the summation of trust values of the remaining sensors. If the system trust is lower than a threshold, this entire UII model is not trustworthy, and actions for system recovery must be taken. One possible action is to re-train the classifier. Another possible action is to instruct the patient to manually examine the artificial leg system.

2.4 Hardware Design

Technical challenges in hardware design are twofold. First of all, in order to increase the decision accuracy, frequent training computations are often necessary. Such training computations need to be done not only periodically with predetermined time intervals but also whenever the system trust level goes below our predetermined threshold. The training algorithms require intensive numerical computations that take very long time in the range of a few minutes to hours on a general purpose computer system [7]. It is very important to substantially speed up this training computation to make the training time of our NMI practically tolerable. The second challenge is the real time processing of decision making in order to have smooth control of artificial legs. Such real time processing includes signal sampling, AD/DA conversion, storage of digital information in memory, executing PR/FSM algorithms, periodical trust management, and decision outputs. To meet these technical challenges, we presented a new hardware design incorporating a multi-core GPU and an embedded system with a built-in flash memory.

High performance and low cost multi-core GPUs [17-20] have traditionally been thought of as commodity chips to drive consumer video games. However, the push for realism in such games along with the rapid development of semiconductor technologies has made GPUs capable of supercomputing performance for many applications at very low cost. There are many low-end to medium GPU controller cards available on the market for under $50. However they deliver extraordinary computation power in the range of several hundreds of GFLOPS.

Besides high performance and low cost, there has also been a technology drive for reliable and low power GPUs alongside FPGAs and CPUs for embedded applications such as military systems. For example, an embedded system using the ATI Radeon HD 3650 GPU [21] draws very little power but delivers performance levels of hundreds of GFLOPS. The next-generation mobile GPUs are expected to nearly double this performance with a similar power envelope.

Our NMI makes the first attempt to exploit such high speed and low cost GPU for the purpose of speeding up complex PR training computations. Our design for the training of the classifier used a NVIDIA 9500GT graphic card that has four multiprocessors with 32 cores working at the clock rate of 1.4GHz. Each multiprocessor supports 768 active threads giving rise to a total of 3072 threads that can execute in parallel. These threads are managed in blocks. The maximum number of threads per block is 512. The size of the global memory is 1GB with bandwidth of 25.6GB/s. 64KB of the global memory is read-only constant memory. The threads in each block have 16KB shared memory which is much faster than the global memory because it is cached. In this study, we connected this GPU card using the x16 PCI Express bus. Whenever the training computation was triggered, the GPU was called in to perform the training process and store the parameters of trained classifier in the flash memory to be used for real time decision-making.

The second part of the hardware design is based on Freescale's MPC5566 132 MHz 32 bits microcontroller unit (MCU) with the Power Architecture as shown in Figure 3. The MCU has 40 channels of ADCs with up to 12 bit resolution and two levels of memory hierarchy. The fastest memory is 32KB unified cache. The lower level memories include 128KB SRAM and 3MB flash memory. The default system clock of the MCU is 12 MHz. The frequency modulated phase locked loop (FMPLL) generates high speed system clocks of 128 MHz from an 8 MHz crystal oscillator. The direct memory access (DMA) engine transfers the commands and data between SRAM and ADC without direct involvement of the CPU. Minimizing the intervention from CPU is important for achieving optimal system response. The device system integration unit (SIU) configures and initializes the control of general-purpose I/Os (GPIOs). The real-time decision of the EMG classifier is sent to a GPIO pin and displayed by a LED light on MPC5566 EVB.

3. EXPERIMENTS AND PROTOTYPE

3.1 Evaluation of Designed Algorithms

Assigned Tasks: To prove the design concept, the NMI system was designed to decipher the task transitions between sitting and standing. These tasks are the basic activity of daily living but difficult for patients with transfemoral amputations due to the lack of knee power. During the transition phase, EMG signals are non-stationary. The classifier was designed in the short transition phase. Although it is possible to activate the knee joint directly based on the magnitude of one EMG signal or force data recorded from the prosthetic pylon, unintentional movements of the residual limb in the sitting or standing position may accidently activate the knee, which in turn may cause a fall in leg amputees. Hence, intuitive activation of a powered artificial knee joint for mode transitions requires accurate decoding of EMG signals for identifying the user's intent from the brain.

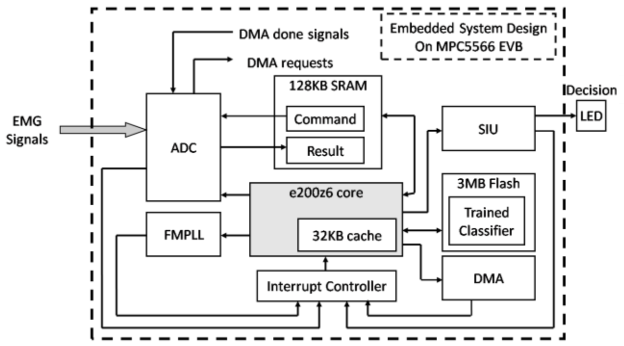

Figure 3. Block diagram of embedded system design on MPC5566 EVB for real-time testing. MPC5566: device modules; ADC: analog-to-digital converter; FMPLL: frequency modulated phase-locked loop; SRAM: internal static RAM; SIU: system integration unit; DMA: direct memory access.

Data Collection: This study was conducted with Institutional Review Board (IRB) approval and informed consent of subjects. One male subject, free from orthopedic or neurological pathologies, was recruited. The seven monitored gluteal and thigh muscles in one side of the lower limb included the gluteus maximus (GMA), gluteus medius (GME), rectus femoris (RF), vastus lateralis (VL), vastus medialis (VM), biceps femoris long head (BFL), and biceps femoris short head (BFS). The EMG electrodes were placed over the anatomical locations described in [22]. The EMG electrodes contained a pre-amplifier, which band-pass filtered the EMG signals between 20 Hz and 450 Hz with a pass-band gain of 1000. The EMG System (Myomonitor®, Delsys Inc., MA) recorded the signals with a 16 Bits signal resolution.

The states of sitting and standing were indicated by a pressure measuring mat. The sensors were attached to the gluteal region of the subject. During the weight bearing standing, the recording of the pressure sensors were zero; during the non-weight bearing sitting, the sensors gave non-zero readings. In addition, force sensing resistors (FSR) were used to record the timings, when the disturbances were introduced to an EMG sensor. All the signals were digitally sampled at a rate of 1000Hz.

Experiment Protocol: In one trial, the subject was instructed to sit on a chair (60 cm high), stand up without any assistance, and then sit down again. In the sitting or standing position, the subject was allowed to move the legs and shift the body weight. A total of 25 trials were conducted. Rest periods were allowed between trials in order to avoid fatigue.

To evaluate the sensor trust algorithm, another 50 trials were tested. In each trial, one of two types of sensor disturbances, i.e. motion artifacts and loss of electrode contact, were introduced randomly in each task phase. To add motion artifacts, the subject tapped an EMG electrode. To simulate loss of electrode contact, the EMG electrode was detached from the skin purposely. Each type of disturbance was tested 25 times. The timings, when the disturbances were introduced, were recorded by a FSR attached on the electrode.

Offline Evaluation of EMG Pattern Recognition: Four classes during the movement initiation phase were considered: sitting, sit-to-stand transition, standing, and stand-to-sit transition. Note that the classes of sitting and standing were not stationary because the subject was instructed to move the legs and shift the body weight in these positions. The actual movements (classes) were identified by the pressure data under the gluteal region. If the pressure data was zero, the subject was in the standing position; if the pressure data was above 80% of maximum value recorded, the subject was in the sitting position. Otherwise, the subject was in a transition from either sit to stand or stand to sit, depending on the previous state. Four TD features defined in [8] and LDA-based classifier were used. Overlapped analysis windows were used in order to achieve prompt system response. For the offline algorithm evaluation, 150ms window length and 20ms window increment were chosen. Due to the relatively small number of tested trials, leave-one-out cross-validation (LOOCV) was utilized in order to receive precise estimation of the classification performance. In the LOOCV procedure, data in one trial was applied as the testing data; the data in the remaining trials were used as the training data. This procedure was repeated so that each trial was used once as the testing data. In addition, a simple FSM was designed and used to improve the system accuracy (Figure 4). Only if the transition condition shown above the arrows was classified, the action of switching states was executed.

Offline Evaluation of Abnormal Detection and Trust Management: EMG electrodes recorded EMG signals under the task transitions, unintentional leg movements, as well as disturbances. There were 3 different states: transitions between sitting and standing (S), normal leg movements (N), and disturbances (D). The detectors detected two types of results: normal (N) or disturbance (D).

For the data sets with motion artifacts, the data in each trial were divided into analysis windows. A state (S, N or D) was assigned to each window. We assumed that the state S is perfectly identified by the classifier and therefore, did not consider state S when evaluating the performance of abnormal detector. There were four detection results: (1) Hit (H): Truth = 'D', Detection = 'D'; (2) False Alarm (F): Truth = 'N', Detection = 'D'; (3) Miss Detection (M): Truth = 'D', Detection = 'N'; and (4) Correct no detection (Z): Truth = 'N', Detection = 'N'. The performance of designed detector were evaluated by

$$\text{Probability of detection : } PD = \frac{H}{H+M}$$

$$\text{Probability of false alarm : } PFA = \frac{F}{F+Z}$$

The trust value was also quantified. In this study, the probability that a sensor behaves normally after one disturbance (P_1) was set to 0.05 for loss of electrode contact and 0.8 for motion artifacts.

3.2 Program Implementation on NMI Hardware System

Both training algorithm and real time testing PR algorithm were implemented on the NMI hardware described in the previous section. The window length and increment were set to 140ms and 80ms, respectively. This is because the speed of *MPC5566* is limited; *MPC5566* needs approximate 80ms to compute the EMG PR algorithm on data in a 140ms window. Therefore, the window increment should be no less than 80ms. If the window length is over 120ms, enlarging the window length does not affect the classification performance [6] but increases the time for a decision-making, which causes delayed system response.

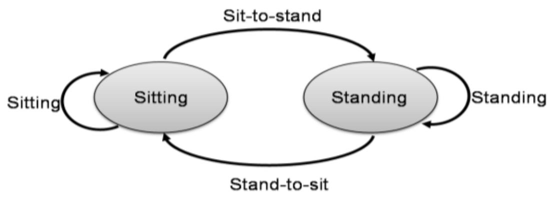

Figure 4. Finite state machine (FSM) model. Ellipses: states (i.e. tasks). Arrows: transitions between tasks.

The computation intensive part of the training algorithm was parallelized for the GPU architecture using CUDA: Compute Unified Device Architecture, which is a parallel computing engine developed by NVIDIA. At the time of this experiment, our GPU was not directly connected to the embedded MCU. Rather, we used NVIDIA 9500GT graphic card plugged into the PCI-Express slot of the PC server to perform the training computation. The training results were then stored in the flash memory of the embedded system board for real time testing. The GPU took the 7 original EMG inputs, each of which had about 10,000 data points. We divided the EMG data into windows with 140 ms in length. As a result, each window contained a 140×7 matrix. The training algorithm first extracted 4 TD features from each channel, producing a 28×1 result matrix for each window. Our parallel algorithm on the CUDA spawned 7 threads for each window resulting in totally 2,800 threads for 400 windows. All these threads were executed in parallel on the GPU to speed up the process. The resultant features were stored in a 28×W matrix, where W is the number of windows. The algorithm then set up K thread blocks, where K is the number of observed motions of the user. Each one of the K thread blocks had 28×14 threads, and a total of K×28×14 threads could execute simultaneously in parallel on the GPU architecture.

To demonstrate the speedup provided by our parallel implementation of this algorithm on the GPU, we conducted an experiment that compared the training times of our training algorithm on both the GPU system and the fully equipped 3GHz Pentium 4 PC server.The real time decision algorithm was implemented on Freescale's MPC5566 embedded system. The parameters of the trained classifier, a 28×4 matrix and a 1×4 matrix, calculated in the training phase by GPU were stored in the built-in flash memory on the MPC5566 EVB in advance. The ADC sampled raw EMG data of 7 channels at the sampling rate of 1000 Hz continuously. Same as in the training phase, the EMG data were divided into windows of length 140ms. A 28×1 feature vector was derived from each window and then fed to the trained classifier. After the EMG pattern classification, one class out of four was identified. The result was post- processed by the FSM and the majority vote algorithm to produce a final decision – sit or stand.

Timing control and data storage are two challenges for the implementation due to the speed and memory limitations of the embedded controller. We developed our own hardware management mechanism on the bare machine of the MPC5566 without depending on any real time OS to avoid unpredictability and delay variations. A circular buffer was used to allow simultaneous data sampling and decision making. The circular buffer consisted of three memory blocks B1, B2 and B3 that were used to store the ADC sampling data. Each block stored the data sampled in one window increment. Another memory block, B4, was used as a temporary storage during algorithm computation.

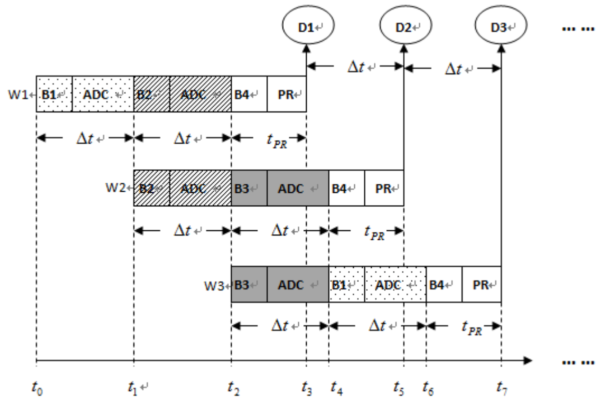

Figure 5. Timing diagram of real-time decision making process.

Figure 5 shows the timing diagram of the control algorithm during the real-time decision making process. In Figure 5, Δt equals to the window increment and t_{PR} is the execution time of PR algorithm. Two conditions need to be satisfied to ensure the smooth control of decision making without delay: (1) $t_{PR} < \Delta t$; (2) $t_w < 2\Delta t$, where t_w is the window length. At point t_0, the ADC begins to sample EMG signals continuously and the digital data are stored in B1. From point t_1, B1 is filled up and the incoming data are stored in B2. At t_2, the data for the first window W1 are available (stored in B1 and B2), and an interrupt request is generated to notify the MCU that the PR algorithm program is ready to run. The PR computation starts. At the same time, new data kept coming in to be stored in B3. After the time interval of t_{PR}, at point t_3, the PR computation of W1 completes. The first decision D1 is made, identifying user's intent in window W1 whether to sit or stand. At time t_4, B3 is filled up and data for W2 are ready for the PR computation again. At this time, B1 is no longer in use so it can be replaced by new sampling data. At time t_5, the decision D2 of window W2 is made. At time t_6, data for W3 (stored in B3 and B1) are available, PR computation for W3 begins. At time t_7, D3 is done and B2 can be reused.

3.3 Real-Time Testing of NMI Prototype

Using the NMI prototype design described above, we carried out real time test on the same human subject recruited for the offline study. The subject performed transitions between sitting and standing continuously. Neural signals as results of such sit-stand transitions were sensed and collected from the subject's muscle. These EMG signals were then fed directly to the embedded system as analog signals. Our decision algorithm was trying to decipher what the subject's intended movement was in real time. The movement decisions made by the classification system were displayed on a LED light and a computer monitor in real time. In our experiment, a 5-window majority vote was applied to the decision stream to further eliminate the classification errors.

4. RESULTS AND DISCUSSIONS

4.1 Performance of EMG Pattern Classifier

The performance of designed EMG pattern classifier in one representative trial is shown in Figure 6. The value of the pressure under the gluteal region of the subject indicates the timing when the subject was either in non-weight bearing sitting (non-zero values) or weight-bearing standing state (zero values). When the number of training windows was 100, decision errors occurred in half of the tested trials. When the training data size increased, the classification error reduced. When the number of training windows was 1000, only 3 out of 25 tested trials had decision errors. Therefore, a large number of training data are desired for improved system accuracy, which, however, increases the computation requirements and therefore challenges the hardware design in practice.

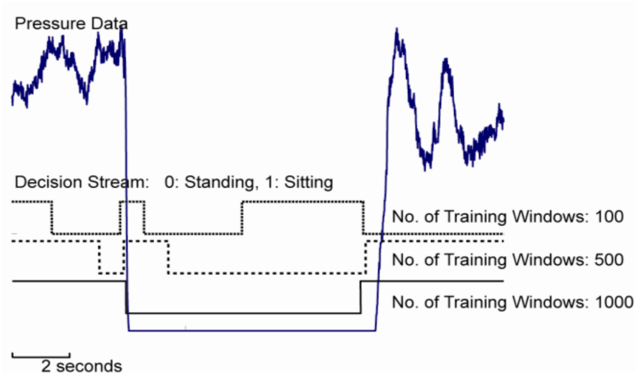

Figure 6. Offline evaluation of designed EMG pattern classifier for identifying user intent. Blue line: pressure under the gluteal region of the subject. Black lines: decision output of designed NMI system when the number of training windows are 100, 500, and 1000, respectively. The value of the decision stream denotes two states, i.e. 0: standing; 1: sitting.

It is noteworthy that the output decisions for mode switches always occur before the rising and falling edges of the pressure signal for all the trials with correct decisions (e.g. the solid decision line in Figure 6). That means our designed EMG classifier can accurately identify the user's intent before the subject fully completed the task transitions. Such a system response is desirable because the neural control of prosthetic joint should occur in the early phase of task transition so that the control signal can trigger the action of the joint.

When 1000 training windows were used, the decision errors were caused by the misclassification between the sit-to-stand class and the stand-to-sit class during the task transition phase. If a 5-window majority vote method is used, the error can be reduced to 0; however, the system response is deteriorated. Other solutions to further improve the system accuracy are to develop more complex EMG features and PR algorithms than the TD features and LDA method. Time-frequency domain features can be efficient in extracting information from non-stationary signals. However, the computation complexity for the time-frequency features is too high, which challenges the hardware design for a fast training process. Our previous study demonstrated that GPU can produce up to 100 times of speedup for computing time-frequency features compared to the regular CPU [7]. Therefore, our designed hardware with GPU power extends our design capability to develop more sophisticated EMG PR methods for improved performance of NMI.

The performance of EMG classifier under two studied disturbances is shown in Figure 7. If an electrode was detached from the skin, a large EMG spike was observed, which resulted in decision errors (Figure 7A). Tapping the electrodes also elicited signal spikes with relatively small magnitude and sometimes signal baseline drifts (Figure 7B). Not all the motion artifacts led to decision errors; the ones resulting in the baseline drifts had more negative impact on the performance of the EMG classifier (Figure 7B). Hence, the design of a trust manager proposed in this study is essential to improve the robustness of designed NMI system.

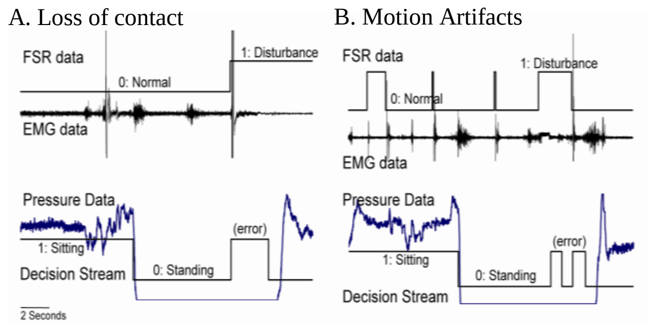

Figure 7. Offline performance of EMG pattern classifier under two disturbances: (A) loss of electrode contact and (B) motion artifacts. The EMG signals under disturbances are demonstrated. The timings when the disturbances were introduced are indicated by FSR data (0: normal; 1: under disturbance). The output decisions (0: standing; 1: sitting) are aligned with the pressure data (the lower panels).

4.2 Performance of Sensor Trust Algorithm

Figure 8 shows the performance of designed trust management method. The CUSUM detection curves were sensitive to two studied disturbances, but insensitive to the muscle activity due to the normal leg motions. Additionally, the CUSUM had very small detection delay. The detection curves immediately yielded high spikes after the disturbances occurred. The trust value for loss of electrode contact directly fell under zero after one event was detected; the trust value for motion artifacts gradually reduced when consistent disturbances were detected. In the future work, we will explore other methods for trust value calculation. For instance, for sensors with non-perfect trust values, we can check whether their future readings are consistent with other sensors that have high trust values. By doing so, the sensors that experienced an occasional disturbance and were not damaged can gradually regain the trust. We compared the ROC curves of the CUSUM detector with two other often suggested change detectors: sliding window detector and mean change detector [23] (Figure 9). The CUSUM detector shows the best performance with very high accuracy. The performance of CUSUM detector achieved 95% detection rate and less than 2% false alarm rate.

The designed CUSUM detector is accurate and prompt. The limitations of current study are that we disturbed only one electrode and the trust manager evaluated the trust at the sensor level. In addition, the algorithm has not been implemented in real time. In the next design phase, we will consider the situation with multiple sensor failures and implement the communication between the trust manager and the classifier for improved system trustworthiness.

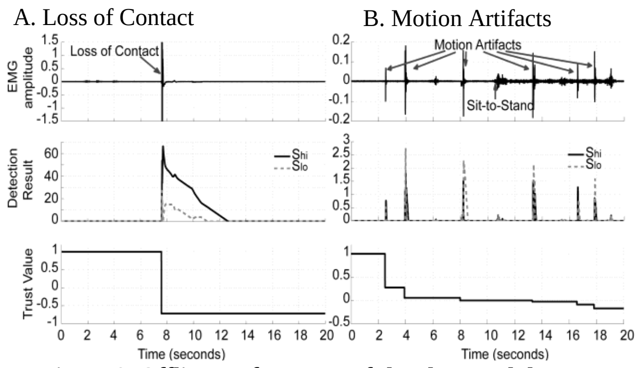

Figure 8. Offline performance of the abnormal detector under (A) loss of electrode contact and (B) motion artifacts. The representative EMG signals (upper panel), the detection curves of CUSUM (middle panel), and the trust value (lower panel) are demonstrated.

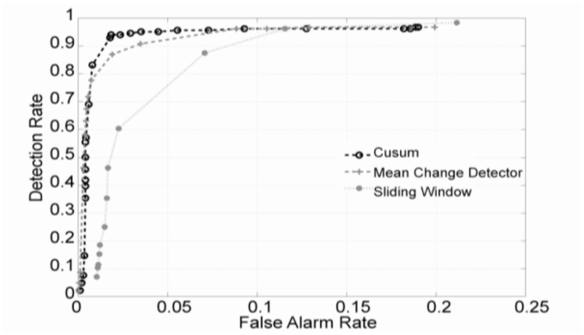

Figure 9. Comparison of ROC curves of the detectors based on CUSUM, change detectors, and the sliding window.

4.3 Performance of CPU vs. GPU for Training Procedure

Table 1 shows the measured speedup of our parallel algorithm on the NVIDIA GPU over the PC server for different window sizes. It is clear from this table that our parallel implementation on the GPU gives over an order of magnitude speedup over the PC server. This order of magnitude speedup is practically significant. Consider the case where the training time took half hour on a PC server [7]. The same training algorithm takes less than a minute using our new parallel algorithm on the GPU. From an amputee user point of view, training for less than a minute for the purpose of accurate and smooth neural control of the artificial leg is fairly manageable as compared to half hour training every time when training is necessary. Furthermore, the speedup increases as the number of windows increases (Table 1). As a result, parallel computation of the training algorithm on GPU helps greatly in the NMI design since the larger the number of windows, the higher its decision accuracy will be, as shown in Figure 6.

Table 1. Speedups of our GPU parallel training algorithm over the 3GHz PC server.

Window size	100	200	400	600	800
Speedup	22.98	29.50	35.94	37.16	39.21

4.4 System Performance in Real-Time

The real time performance of the NMI prototype is shown in Figure 10. The continuous testing lasted for over 10 minutes. All the transitions between sitting and standing were accurately identified. Although the subject moved the instrumented side of leg in the sitting and standing position and shifted the weight in the standing position, no erroneous mode switch was presented. Since a 5-window majority vote method was applied, around 400ms decision delay for the sit-to-stand transitions were observed, compared to the falling edges of pressure data (indicated by arrows in Figure 10). The video of real-time system performance can be found at http://www.ele.uri.edu/linc/htm/video.html. Based on the subject's performance, the decision switched from sitting to standing before the full weight-bearing standing position. Clearly, the majority vote method significantly improves the system accuracy but sacrifices the system response.

The 100% classification accuracy in the real time testing demonstrates the potential of our designed NMI prototype. Although the system response for sit-to-stand should occur before the subject lifts all the weight from the chair, 400ms delay may be acceptable for leg amputees. This is because the prosthetic joint does not react and the amputee cannot stand up until the decision from the NMI is made. In addition, the system response can be improved by using other types of embedded system faster than Freescale's MPC5566. Due to the speed limitation of MPC5566, the decision could only be updated every 80ms in this study. If a faster embedded system were used, the window increment size could have been reduced, and the decision update rate can be improved.

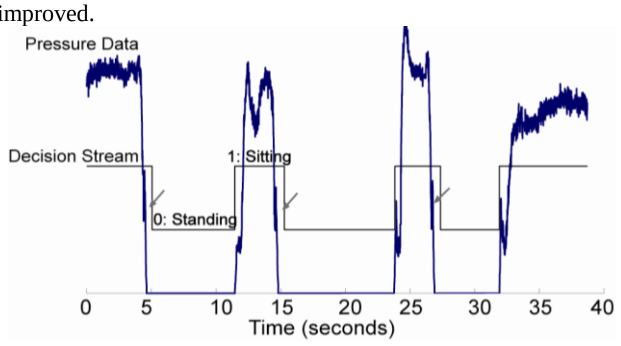

Figure 10. Real time performance of designed NMI system. The decision stream (0: standing, 1: sitting) is aligned with the pressure data (blue line) measured from gluteal region of the subject. Arrows indicates the delays of the decision.

5. Related Work

Real-time EMG pattern recognition has been designed to increase the information extracted from EMG signals and improve the dexterity of myoelectric control for upper limb prosthetics [5, 11]. However, no EMG-controlled lower-limb prostheses are available. Recently, the need for neural control of prosthetic legs has brought the idea of EMG-based control back to attention. Two previous studies have attempted to use EMG signals to identify locomotion modes for prosthetic leg control [6, 24]. Jin et al. [24] used features extracted from EMG signals from a complete stride cycle. Using such features, the algorithm results in a time delay of one stride cycle in real-time. In practical application, this is inadequate for safe prosthesis use. Our previous study designed a phase-dependent EMG pattern recognition method [6], which is a dynamic classifier over time. The result indicated over 90% classification accuracy, which can be applied for real time NMI. While both studies demonstrated that EMG information recorded from transfemoral amputees is sufficient for accurate identification of user intent, there has been no experimental study on design and implementation of embedded system to realize the NMI for reliable and real time control of prosthesis.

Trust has been a well-studied concept in sociology and psychology [25]. Recently, it was introduced to the distributed computer networks for the purpose of (a) enhancing network security and reliability and (b) stimulating cooperation among network entities [26]. Various trust models have been developed to quantify trust in authorization and access control, electronics commerce, peer-to-peer networks, ad hoc and sensor networks, electronic communities, and pervasive computing. However, the concept of trust is rarely used in biomedical systems.

There has been extensive research in using GPUs for general purpose computing (GPGPU) to obtain exceptional computation performance for many data parallel applications [21, 27-31]. A good summary of GPGPU can be found in [27, 29]. Our prior study made the first attempt to use GPU in EMG-controlled artificial legs and other medical applications [7]. Our results on individual computation components on EMG signal pattern recognition showed good speedups of GPU over CPU for various window sizes. The focus of the work reported in [7] was on parallel implementations of individual algorithms on GPU whereas this paper makes the first attempt to integrate the entire system for neural-machine interfacing (i.e. a CPS system) for real time control of artificial legs. Our prior works [7] report offline analysis, while the work presented in this paper implements online decoding method for real-time testing. To the best knowledge of the authors, there has been no existing study on implementing the entire training algorithm on GPU for different numbers of windows and integrating the training algorithm together with real time testing on the same subject.

6. Conclusions

A new EMG-based neural-machine interface (NMI) for artificial legs was developed and implemented on an embedded system for real time operation. In such a cyber-physical system, the cyber and the physical system were tightly integrated to achieve high accuracy, reliability, and real-time operation. This cyber-physical system consists of (1) an EMG pattern classifier for decoding the user's intended lower limb movements and (2) a trust management mechanism for handling unexpected sensor failures and signal disturbances. The software was then embedded in a newly designed hardware platform based on an embedded microcontroller and a GPU to form a complete NMI for real time testing. To prove our design concepts, we conducted preliminary experiments on an able-bodied human subject to identify his intent for sitting and standing. The result showed high system accuracy and reasonable time response for real time operation. Our NMI design concept has a great potential to allow the leg amputees to intuitively and efficiently control the prosthetic legs, which in turn will improve the function of prosthetic legs and the quality of life of patients with leg amputations. Our future work includes the consideration of other movement tasks such as walking on different terrains and test of the designed system on patients with transfemoral amputations.

7. ACKNOWLEDGMENTS

This work is supported by NSF/CPS #0931820 and NSF/CCF #0811333, DoD/TATRC #W81XWH-09-2-0020, Department of Education/NIDRR #H133F080006, and Nation Nature Science Foundation of China NSFC #60736013. Authors thank Andrew Burke for his assistance in this project.

8. REFERENCES

[1] Herr, H. and Wilkenfeld, A. 2003. User-adaptive control of a magnetorheological prosthetic knee. Industrial Robot: An International Journal, 30, 1 (2003), 42-55.

[2] Psonak, R. 2000. Transfemoral prosthetics. Butterworth-Heinemann Publications.

[3] Basmajian, J. and De Luca, C. 1985. Muscles alive: their functions revealed by electromyography. Williams & Wilkins.

[4] Parker, P. A. and Scott, R. N. 1986. Myoelectric control of prostheses. Crit Rev Biomed Eng, 13, 4 (1986), 283-310.

[5] Englehart, K. and Hudgins, B. 2003. A robust, real-time control scheme for multifunction myoelectric control. IEEE Trans Biomed Eng, 50, 7 (Jul 2003), 848-854.

[6] Huang, H., Kuiken, T. A. and Lipschutz, R. D. 2009. A strategy for identifying locomotion modes using surface electromyography. IEEE Trans Biomed Eng, 56, 1 (Jan 2009), 65-73.

[7] Xiao, W., Huang, H., Sun, Y. and Yang, Q. 2009. Promise of embedded system with GPU in artificial leg control: Enabling time-frequency feature extraction from electromyography. In Proceeding of IEEE International Conference on Engineering in Medicine and Biology Society, 1 (NM, US, Sep 3-6, 2009), 6926-6929.

[8] Hudgins, B., Parker, P. and Scott, R. N. 1993. A new strategy for multifunction myoelectric control. IEEE Trans Biomed Eng, 40, 1 (Jan 1993), 82-94.

[9] Guler, N. F. and Kocer, S. 2005. Classification of EMG signals using PCA and FFT. J Med Syst, 29, 3 (Jun 2005), 241-250.

[10] Ajiboye, A. B. and Weir, R. F. 2005. A heuristic fuzzy logic approach to EMG pattern recognition for multifunctional prosthesis control. IEEE Trans Neural Syst Rehabil Eng, 13, 3 (Sep 2005), 280-291.

[11] Kuiken, T. A., Li, G., Lock, B. A., Lipschutz, R. D., Miller, L. A., Stubblefield, K. A. and Englehart, K. B. 2009. Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms. Jama, 301, 6 (Feb 11 2009), 619-628.

[12] Gill, A. 1962. Introduction to the theory of finite-state machines. McGraw-Hill, New York.

[13] Yang, Y., Sun, Y., Kay, S. and Yang, Q. 2009. Securing Rating Aggregation Systems using Statistical Detectors and Trust. IEEE Transactions on Information Forensics and Security, 4, 4 (2009), 883-898.

[14] Page, E. S. 1954. Continuous Inspection Scheme. Biometrika 41, 1/2 (1954), 100-115.

[15] Philips, T., Yashchin, E. and Stein, D. 2003. Using Statistical Process Control to Monitor Active Managers. Journal of Portfolio Management 30, 1 (2003), 186-191.

[16] Sun, Y., Yu, W., Han, Z. and Liu, R. 2006. Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks. IEEE JSAC special issue on security in wireless ad hoc networks, 24, 2 (2006) 305-317.

[17] AMD ATI Mobility Radeon™ HD 4800 Series Graphics. http://ati.amd.com/products/mobilityradeonhd4800/.

[18] NVIDIA Corporation, CUDA: compute unified device architecture programming guide. www.nividia.com.

[19] Seiler, L., Carmean, D., Sprangle, E., Forsyth, T., Abrash, M., Dubey, P., Junkins, S., Lake, A., Sugerman, J., Cavin, R., Espasa, R., Grochowski, E., Juan, T. and Hanrahan, P. Larrabee: A Many-Core x86 Architecture for Visual Computing. http://download.intel.com/technology/architecture-silicon/Siggraph_Larrabee_paper.pdf.

[20] Quantum3D introduces industry-leading graphics accelerator XMC: SENTIRIS, 5140. http://www.quantum3d.com.

[21] Commike, A. Maximizing GPGPU computing for embedded systems. Military Embedded Systems, http://www.mil-embedded.com/articles/id/?37422009.

[22] Perotto, A. O., Delagi, E. F., Iazzetti, J. and Morrison, D. 2005. Anatomical Guide For The Electromyographer: The Limbs And Trunk. Charles C Thomas Pub Ltd.

[23] Yang, Y., Sun, Y., Kay, S. and Yang, Q. 2009. Defending Online Reputation Systems against Collaborative Unfair Raters through Signal Modeling and Trust. In Proceeding of The 24th ACM Symposium on Applied Computing (Hawaii, US, Mar 8 - 12, 2009).

[24] Jin, D., Yang, J., Zhang, R., Wang, R. and Zhang, J. 2006. Terrain Identification for Prosthetic Knees Based on Electromyographic Signal Features. Tsinghua Science and Technology, 11, 1 (2006), 74-79.

[25] Gambetta, D. 2000. Can we trust trust? University of Oxford.

[26] Josang, A., Ismail, R. and Boyd, C. 2007. A survey of trust and reputation systems for online service provision. Decision Support Systems, 43, 2 (2007) 618-644.

[27] Owens, D. J., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A. E. and Purcell, T. J. 2007. A Survey of General-Purpose Computation on Graphics Hardware. Computer Graphics Forum, 26, 1 (2007), 80-113.

[28] Silberstein, M., Schuster, A., Geiger, D., Patney, A. and Owens, D. J. 2008. Efficient Computation of Sum-products on GPUs Through Software-Managed Cache. .

[29] Houston, M. General Purpose Computation on Graphics Processors (GPGPU). http://www-graphics.stanford.edu/~mhouston/public_talks/R520-mhouston.pdf.

[30] Segal, M. and Peercy, M. A performance-oriented data parallel virtual machine for GPUs. http://ati.amd.com/developer/siggraph06/dpvm_e.pdf.

[31] Volkov, V. and Demmel, J. W. 2008. Benchmarking GPUs to tune dense linear algebra. In Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, (TX, US, Nov 15-21, 2008)