CONTROL OF ASSISTIVE MOBILE ROBOTS VIA sEMG SIGNALS

Julia Amaya*, Ricardo Souza*, Leonardo Olivi*, Eleri Cardozo*

*School of Electrical and Computer Engineering
University of Campinas - UNICAMP
Campinas, São Paulo, Brasil

Emails: julia.feec@gmail.com, rssouza@dca.fee.unicamp.br, lrolivi@dca.fee.unicamp.br, eleri@dca.fee.unicamp.br

Abstract— An individual's motor ability, such as walking, can be easily compromised by a wide range of factors, such as traumatic spinal cord injury, cerebral vascular accidents, limb amputations, multiple sclerosis, etc. Some technologies can be applied to assist these individuals improving their mobility and quality of life. Such technologies are often referred to as assistive technologies and one fitting this scenario is mobile robotics. This paper proposes a lightweight sEMG (surface Electromyography) control based approach aimed for assistive systems where all signal processing and robot's control is embedded on a Android based cellphone. The system also provides a simple collision avoidance based on the velocity control upon obstacle detection.

Keywords— Mobile Robotics, Digital Signal Processing, Assistive Technology

Resumo— A capacidade motora de um individuo, como caminhar, pode ser facilmente comprometida por uma variedade de fatores, como traumatismo na coluna, acidentes vasculares cerebrais, amputação de membros, esclerose multipla, etc. Algumas tecnologias podem ser utilizadas para auxiliar estes individuos melhorando sua mobilidade e qualidade de vida. Estas tecnologias são comumente chamadas de tecnologias assistivas e a robótica móvel se encaixa bem nesta categoria. Este trabalho apresenta uma solução de baixo custo computacional que utiliza sensores sEMG (*surface Electromyography*) como interface do sistema assistivo onde todo o processamento dos sinais e controle do robô esta embarcado em um *smartphone* com o sistema operacional Android. O sistema tamém oferece uma proteção contra colisões simples baseado na detecção de obsáculos.

Palavras-chave— Robótica móvel, Processamento digital de sinais, Tecnologia Assistiva

1 Introduction

An individual's motor ability, such as walking, can be severely compromised by a wide range of factors, such as traumatic spinal cord injury, cerebral vascular accidents, limb amputations, multiple sclerosis, etc. Some technologies can be applied to assist these individuals improving their mobility and quality of life, often referred to as assistive technologies. Although usually aimed for paraplegic patients, such technologies can also be applied to patients with more severe movement restrictions (e.g., victims of cerebral vascular accident). In order to make such technologies available, more reliable and appropriate human-machine interfaces (HCI) are required. This is where technologies such as BCI (Brain-Computer Interface) and Electromyography (EMG) can play a major role. All these techniques are able to interpret some specific signals into useful features and commands that could represent user's needs or intentions.

The EMG is a technique that captures muscular activity through carefully placed electrodes. These electrodes can be classified into two groups: superficial or intramuscular (Bendre et al., 2009). Although intramuscular electrodes are able to provide more reliable signal they require medical intervention to be placed. Also, intramuscular electrodes are a better fit when analysing specific muscle fibers. Superficial electrodes, on the other hand, can be easily placed on the subject's skin and are able to monitor the general behavior of a desired muscle (or set of muscles), for instance, facial muscles involved in eye blinking or jaw clenching.

There are several techniques employed to perform EMG signal processing (Thompson et al., 1996) such as wavelets and Fourier transforms (Lathi, 1998). These techniques usually decompose the signals into MUAPs (Motor Unit Action Potentials). The actual EMG reading is the result of the superposition of electric signals from different motor units and in many cases each motor unit has to be analysed individually (Reaz et al., 2006). As such, applications sometimes require powerful computers to perform all the signal detection and processing. The use of such computers can become a drawback in mobile application such as the control of robotized wheelchairs (Felzer and Freisleben, 2002; Wei and Hu, 2010).

This paper proposes an lightweight sEMG-based control aimed to assistive locomotion aiding systems where all signal processing and robot's control is embedded on a mobile device such as smartphones and tablets running Android. The system also provides a simple collision avoidance function.

This paper is organized as follows. Section 2 presents some related work. Section 3 presents the proposed system. Section 4 presents the experiments and evaluation of the proposed solution and finally Section 5 concludes the paper.

2 Related Work

There are a variety of solutions that aim to provide an assistive interface to control assistive mobile robots/wheelchairs through the capturing and processing of EMG signals.

The control interface proposed in (Han et al., 2003) uses sEMG signals from the shoulders and is aimed for quadriplegic patients. The system has two operating modes: in Mode 1 the system goes forward while the user is sending the forward command, such as raising both shoulders, and Mode 2 uses a toggle strategy similar to the one proposed by this paper. The solution proposed in (Moon et al., 2005) also captures the shoulders movement with sEMG sensors and translate the signals into commands for the wheelchair/mobile robot. The authors employ a double threshold method to determine whether a movement was a voluntary or involuntary one. As these solutions aim for quadriplegic patients their interfaces may not be fitted in other scenarios, for instance, patients victims of cerebral vascular accident that might not have the voluntary control over their shoulders.

In (Choi et al., 2006) the authors propose an sEMG interface that employs signals captured from the neck or the arm. The user can use the neck to move the wheelchair in the desired direction or use the arm direction to guide it. As mentioned before the use of neck or shoulders movement may not be available for patients with more complex disabilities and in such drastic scenarios the use of arm movement can not be considered.

Another sEMG control solution is presented in (Silva et al., 2013). The solution uses a single channel to control a wheelchair inside a virtual environment. The system uses signals from the jaw (detecting bites) to generate commands to the wheelchair. Since it only employs one channel there are three possible situations: silence, one peak and two consecutive peaks. When the system detects one peak it changes between forward and turn right and when two peaks are detected it switches between stop and turn left. Besides the system has been validated only on a virtual environment the use of a single channel to obtain four different commands can lead to situations where the commands can be misinterpreted, for instance, when the user tries to send a stop command during an emergency.

In (Wei and Hu, 2011) a multi-modal interface is presented. In this case the control interface uses both sEMG and visual processing to command a robotic wheelchair. The sEMG is used to capture eye blink and jaw clenching while the visual processing is used to validate the eye blink and avoid an misinterpretation from the sEMG classifier. By employing sEMG and visual processing the system is able to offer six different commands. The main drawback of this solution is the need of high computing power where the wheelchair must be equipped with powerful CPUs.

Yet another sEMG-based interface to control robotic wheelchair is presented in (Kim-Tien and Truong-Thinh, 2011). This solution also employs a combination of sensors to acquire the command information from the user. In this case the authors uses sEMG along side an EOG (Electrooculogram). Therefore two types of signals acquired: eyes movement (look right/left) and muscle contraction (raise of eyebrows or blinking). The eyes movement were associated with turn left/right function while the muscle contraction was associated with the forward/backward movement. One drawback of this solution is the use of natural movement, such as looking right, to send commands to the robotic wheelchair, since such movements could be performed in an involuntary manner.

The major difference of all the mentioned works is the fact that all systems are designed to run on powerful and energy consuming computers while the system proposed in this paper favors a simpler and yet effective solution designed to run embedded on mobile devices, e.g., on the patient's own smartphone.

3 The Proposed System

As mentioned before, this paper describes a distributed lightweight system with an EMG HCI that allow disabled patients to safely teleoperate a motored wheelchair using eye blinks. The system is compact enough to run signal processing and robot control algorithms embedded on a Android smartphone or tablet.

The proposed system employs Shimmer Wireless Sensor Modules equipped with sEMG sensors shown in Figure 1, one for each eye. The sEMG signal is captured and transmitted via Bluetooth to an Android Smartphone, in our case a Samsung Galaxy S. Once the signal is received, the embedded algorithm processes it detecting the eye blinks that are translated into teleoperation commands (e.g. turn left, go forward, stop, etc) that are then sent to the robot.

shimmer
* image shows module connected to the Shimmer baseboard
shimmer
* must be connected to the Shimmer baseboard

Figure 1: Shimmer WSN module equipped with sEMG sensor.

The proposed system can be better specified when divided into 4 different functions: Signal capture and transmission, signal processing, robot actuation, and robot protection. Each of these functions will be addressed in the sequence.

3.1 *Signal Capture and Transmission*

To acquire the sEMG signals the Shimmer Platform (Shimmer, accessed on Feb. 2013) was employed. The Shimmer platform offers a wide range of sensors, from simple environmental (e.g., temperature, light) to biophysical sensors (e.g. Electromyograph, Electrocardiograph, Galvanic Skin Response). An Application Programming Interface (API) was developed in such a way that any Android application would be able to read any platform sensor. The API was developed based on the protocol defined by the manufacturer in the provided firmware. The protocol defines message formats that allow clients to read a collection of sensors with a determined sampling rate.

The Shimmer platform has two radio interfaces, Bluetooth and IEEE 802.15.4. We favor the Bluetooth connection since the data will be transmitted to an Android device that already supports this communication protocol. As such, the API must handle all the connections and configuration issues between the Android device and the Shimmer modules.

One important issue related to the signal capturing is the electrodes placement. Figure 2 presents the used electrodes placement on the user's face. Electrodes misplacing can impose severe instability on the signal processing therefore it's important that the electrodes are placed correctly. This set up was chosen so the sEMG sensors could be used to detect eye blinking and jaw clenching. The reference electrodes are placed behind the ears (GND).

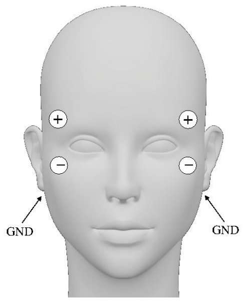
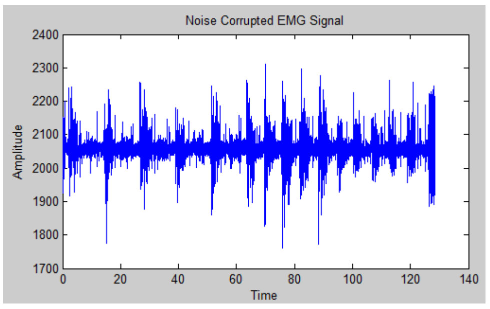


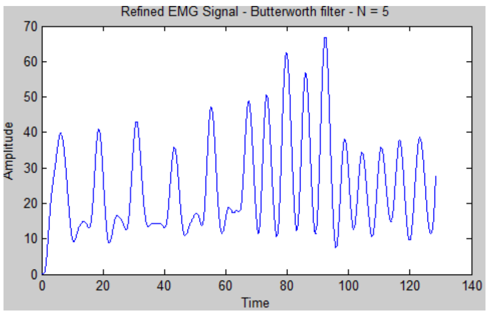
Figure 2: sEMG electrodes placement.

3.2 *Signal Processing*

After the capture and transmission of the sEMG signals, the application must process those signals in order to detect eye blinks. Since two sensors were employed, one for each eye, the application must analyze each data stream to detect the intentional blink, while ignoring involuntary muscular activity. Other important issue is the fact that the proposed solution was developed aiming Android devices, meaning that processing and memory constraints were taken into consideration. A Butterworth filter was chosen in order to both minimize noise interference and perform the actual blink detection. The Butterworth was defined as a second order low pass filter with a cutoff frequency of 0.1Hz. Figure 3 presents the sEMG signal 3(a) along side with the filter output for $N = 2$ 3(b). After filtering the signal the actions are detected using thresholds. An order two filter was chosen due to the memory and processing constraints.



(a)



(b)

Figure 3: (a) sEMG captured signal (b) Butterworth filter output for N = 2.

After selecting the Butterworth filter to perform the signal processing, the filter was tunned using MATLAB and only then ported to the Android device. The MATLAB filter is implemented as a *direct form II transposed* showing in the block diagram of Figure 4.

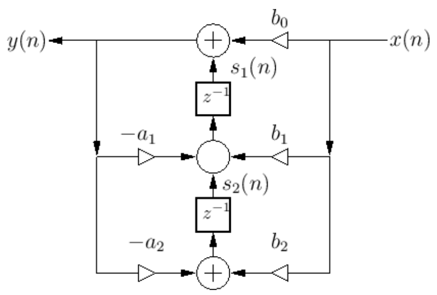


Figure 4: Block diagram for the Butterworth filter.

As mentioned, the first goal was to apply eye blinking and jaw clenching on the interface but due to the limited computing power of the Android device the system currently employs only eye blinks.

3.3 Robot Actuation

After processing the sEMG signal and detecting whether or not the user has blinked an eye this simple action must be translated into a command to the robot. Since we are only using two channels, there are 4 different possibilities: left eye blink, right eye blink, left and right blinks or immobility (no blinks). As such, a control method (teleoperation) must be implemented with only these four possible actions. Therefore we defined different results for each action depending on the current state of the robot, as depicted by Figure 5.

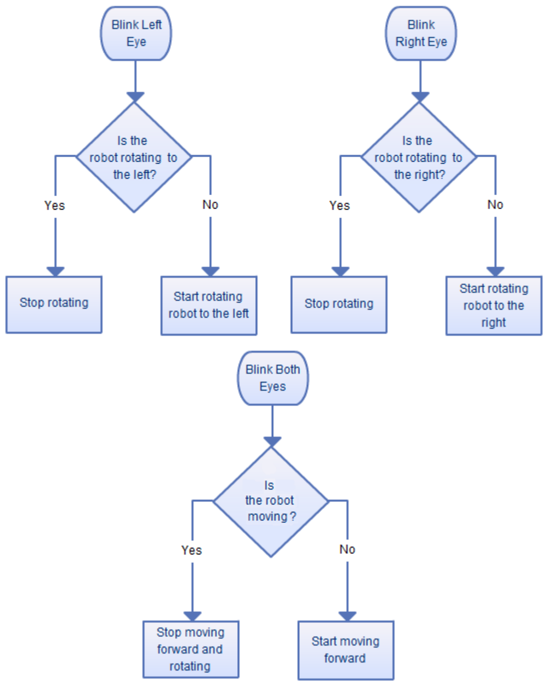

Blink Left Eye
Is the robot rotating to the left?
Yes
No
Stop rotating
Start rotating robot to the left
Blink Right Eye
Is the robot rotating to the right?
Yes
No
Stop rotating
Start rotating robot to the right
Blink Both Eyes
Is the robot moving?
Yes
No
Stop moving forward and rotating
Start moving forward

Figure 5: Robot's commands flow chart.

The robot used to evaluate the solution is accessed through a framework for networked robotics, called HttpThru (Cardozo et al., 2010). All interactions are done using XML over HTTP. Another API was developed for the Android platform to support the XML parsing and manage HTTP requests and responses. The API makes high level methods available for the developer, such as *getVel or setVel* to read and update the robot's velocity, or *getLaserReadings* to acquire the laser rangefinder readings.

3.4 Robot Protection

To allow users to perform a safe navigation a simple protection method was also developed. This protection aims to avoid frontal collision due to sEMG signal processing latencies and errors (misinterpreted commands). A 30°region in front of the robot was determined to be the protection zone, whenever obstacles are detected within this region the protection is activated. The implemented protection decreases the robot's velocity according to the closest obstacle detected, within the protection zone, until bringing the robot to a complete halt when a minimum distance is reached. To perform this task a laser rangefinder was employed.

4 Experiments and Evaluation

The first evaluation was to determine the signal processing delay. To do so a sequence of one second blinks was recorded. Using MATLAB it was determined that the delay time between the blink and the system processing and classifying that blink was about 420 ms. Since two independent filters are employed, one for each eye, after detecting a blink the system has to verify if both eyes were blinked. When performing a double blink (blinking both eyes simultaneously) it was determined an average delay of 100ms between the detection of each blink and an upper bound of 300 ms. After these measurements it was determined that after detecting a blink the system would wait a 400 ms timeout before interpreting the command as a single or double blink. Besides the mentioned detection delay and verification timeout there is also the delay imposed by the APIs processing time, approximately 180 ms. When accounting for all the delays and processing times the average time to send a single blink command to the robot was 1000 ms while a double blink command took 700 ms.

The double blinking detection is performed faster because once the second blink is detected the system doesn't have to wait until the timeout expires, meanwhile on the one blink case the system is upper bounded by the 400ms timeout plus the actual blink detection and API processing times. Another important aspect of the signal processing is event detection error. This experiment has also shown that the desired command was accurately detected 96% of the times which is an excellent rate.

To evaluate the proposed system and its performance a simulated and a real experiment were performed. Both experiments had the same objective: guide the mobile robot (or the simulated robot) to a specific place relying only on the eye blinking control. During both scenarios the user was fitted with the sEMG sensors and all the data processing was performed by the Android smartphone. It's important to note that on this preliminary phase all experiments were conducted by the developers themselves.

Figure 6 presents the general overview of the experiment. First, the *Shimmer API* is used to establish a connection, configure desired parameters (such as sampling rate) and initiate the sEMG signal capture on both devices. Each signal is sent to the Android device through a Bluetooth connection and the Shimmer API and also responsible for parsing these data streams. The sEMG signal is then processed by the Android device that translates the readings into the desired commands and, using the *Robot API*, sends these commands to the robot. All the communication between the Android application and the robot or simulator is performed using HTTP. The only difference between using the simulator or the real robot from the application's point of view is the IP address of the vehicle's embedded server configured on the smartphone.

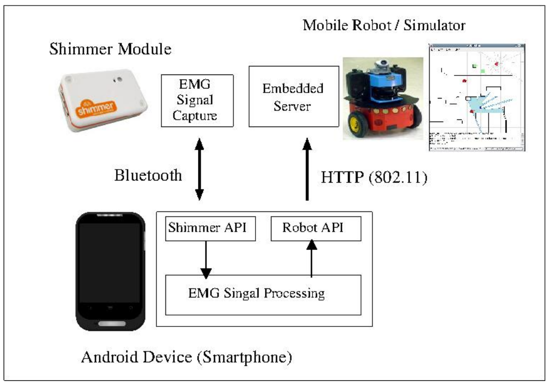


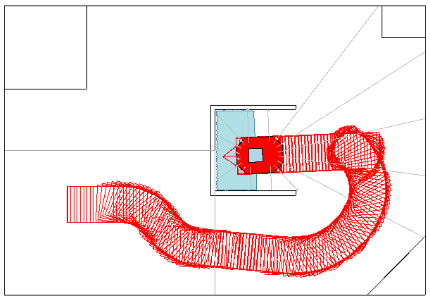
Figure 6: Experiment overview.

The first experiment was performed using a simulated scenario. The objective was to evaluate the response time between the user's command (blink of an eye) and the actual command execution in order to setup the robot's linear and rotational velocities as well as to tune the protection algorithm. Linear and rotational velocities were set to 150 mm/s and 7 degrees/s, respectively. The protection distance was set to 45 cm.

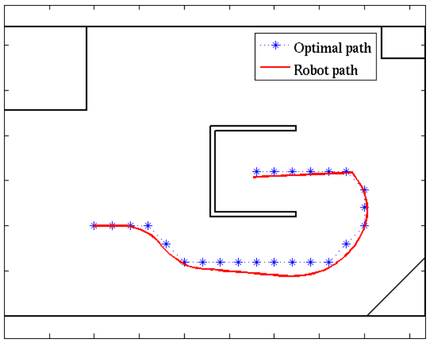
The simulated experiments was done using the MobileSim Simulator (MobileSim, 2013). The map loaded into the simulation was made with the exact same dimensions and obstacles as the environment used to perform the experiments with the real robot.

The simulation environment and final navigation path are both depicted on Figure 7(a). During the experiments it was noticed that once the robot enter the final goal the user didn't have to stop the robot as the protection took over and didn't allow him/her to go any further.

To evaluate the navigation result the path performed by the user was compared to a path calculated by A* path planner (Siegwart, 2004), depicted by Figure 7(b). It's possible to see that the user was able to navigate very close to the planner's path. The average error between the



(a)



(b)

Figure 7: (a) Simulated experiment (b) Trajectory provided by the path planner and experiment path.

user's path and the A* result was 40.35 mm with a standard deviation of 37.53 mm and a confidence interval (90%) of 5.14 mm.

After the simulation, real experiments were conducted in our laboratory within the same environment used on the simulation. Figure 8 presents a snapshot of the experiment. The user was able to guide the robot inside the structure presented by Figure 8 using only the sEMG teleoperation interface and the proposed protection system, as no control algorithm was employed.

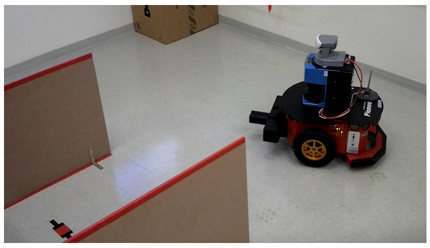


Figure 8: User guiding the robot within the lab with the sEMG interface.

5 Conclusion and Future Work

This paper presented an sEMG interface for controlling an assistive vehicle such as a robotic wheelchair that in our experiments was emulated by a mobile robot (a Pioneer P3-DX). The system was able to provide a safe navigation from the initial position to the desired final one. The total end-to-end delay, that is between the actual eye blink and the command being send to the robot, was 1000ms for blinking one eye and 700ms when blinking both eyes. As mentioned, this time difference is due to the timeout needed to verify is the user blinked one or both eyes. The developed protection was also able to guarantee a safe navigation. It was also verified that the use of a reliable simulated environment can be very useful making users comfortable with the proposed interface before applying it on real devices and also for calibrating the signal processing algorithm.

The major contribution of the paper is the possibility to perform all the signal processing and command interpretation embedded on an Android smartphone, in our case a Samsung Galaxy S. Another contribution is that all the developed tools, such as the API for communicating the Android device with the Shimmer devices and for communicating the Android device with the robot were implemented so it could be reused by any Android application as was the protection system. Future work include aggregating different muscle movement (e.g. jaw clenching), performing patient monitoring employing available sensors (e.g. ECG), interaction with other distributed resources or services on the environment aiming for a more comfortable navigation, applying a more robust protection system and performing experiments with real patients. These extensions are possible thanks to the new generation of smartphones equipped with processors much more powerful than the one employed in this work.

References

Bendre, B., Joshi, A., Raybagkar, V. H., Desa, K. and Mahamuni, R. (2009). *Emerging Physics*, Pearson Education.

Cardozo, E., Guimaraes, E., Agostinho, L., Souza, R., Paolieri, F. and Pinho, F. (2010). A platform for networked robotics, *Proceedings of the IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, pp. 1000–1005.

Choi, K., Sato, M. and Koike, Y. (2006). A new, human-centered wheelchair system controlled by the emg signal, *Neural Networks, 2006. IJCNN '06. Int. Joint Conf. on*, pp. 4664–4671.

Felzer, T. and Freisleben, B. (2002). Hawcos: The "hands-free" wheelchair control system, *In Proc. ASSETS 2002*, ACM Press, pp. 127–134.

Han, J.-S., Zenn Bien, Z., Kim, D.-J., Lee, H.-E. and Kim, J.-S. (2003). Human-machine interface for wheelchair control with emg and its evaluation, *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual Int. Conf. of the IEEE*, Vol. 2, pp. 1602–1605.

Kim-Tien, N. and Truong-Thinh, N. (2011). Using electrooculogram and electromyogram for powered wheelchair, *Robotics and Biomimetics (ROBIO), 2011 IEEE Int. Conf. on*, pp. 1585–1590.

Lathi, B. (1998). *Signal Processing and Linear Systems*, Berkeley-Cambridge Press.

MobileSim (2013). Mobilesim - mobilerobot's research and academic custumer support, `http://robots.mobilerobots.com/wiki/MobileSim`.

Moon, I., Lee, M., Chu, J. and Mun, M. (2005). Wearable emg-based hci for electric-powered wheelchair users with motor disabilities, *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE Int. Conf. on*, pp. 2649–2654.

Reaz, M., Hussain, M. and Mohd-Yasin, F. (2006). Techniques of emg signal analysis: detection, processing, classification and applications, *Biological Procedure Journal* .

Shimmer (accessed on Feb. 2013). Wireless sensor platform for wearable applications, `http://www.shimmer-research.com`.

Siegwart, R. (2004). *Introduction to Autonomous Mobile Robots*, MIT Press.

Silva, A., Morere, Y., Naves, E., de Sa, A. and Soares, A. (2013). Virtual electric wheelchair controlled by electromyographic signals, *Biosignals and Biorobotics Conference (BRC), 2013 ISSNIP*, pp. 1–5.

Thompson, B., Picton, P. and Jones, N. B. (1996). A comparison of neural network and traditional signal processing techniques in the classification of emg signals, *Proceedings of the IEEE Colloquium on Artificial Intelligence Methods for Biomedical Data-Processsing*, Vol. 8, pp. 1–5.

Wei, L. and Hu, H. (2010). Emg and visual based hmi for hands-free control of an intelligent wheelchair, *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, pp. 1027–1032.

Wei, L. and Hu, H. (2011). A multi-modal human machine interface for controlling an intelligent wheelchair using face movements, *Robotics and Biomimetics (ROBIO), 2011 IEEE Int. Conf. on*, pp. 2850–2855.