Towards a Natural Interface for the Control of a Whole Arm Prosthesis

G. Gini, P. Belluco, F. Mutti, D. Rivela and A. Scannella

Abstract In the present study we illustrate a new concept for a user interface to control whole arm prosthesis. We extend the myo-electric control, taking into account the intended movement of the shoulder and integrating vision analysis to give a "visual control" to the user. While the control of the shoulder is partially obtained from pattern recognition of sEMG signals, the control of the elbow and wrist joints is only derived using the trajectory computed from the initial to the target position. We show results in simulation and discuss about future steps in developing the prosthesis.

Keywords Prosthesis control · sEMG signals · Classifiers · Kinect · User interface

1 Introduction

The interface between the patient and the prosthesis is critical to the success of the device. This is even truer for upper limbs prosthesis that can accomplish a large variety of tasks [5, 8, 16].

G. Gini (✉) · F. Mutti · D. Rivela · A. Scannella
DEIB, Politecnico di Milano, Milan, Italy
e-mail: giuseppina.gini@polimi.it

F. Mutti
e-mail: f.mutti@b10nix.com

D. Rivela
e-mail: diletta.rivela@mail.polimi.it

A. Scannella
e-mail: alessia.scannella@mail.polimi.it

P. Belluco
BIONIX, Milan, Italy
e-mail: p.belluco@b10nix.com

H. Bleuler et al. (eds.), *New Trends in Medical and Service Robots*,
Mechanisms and Machine Science 38, DOI 10.1007/978-3-319-23832-6_5

Today prosthetic solutions include aesthetical, functional passive, and actuated ones. While the first kind cannot make movements but only adapt to external forces, the functional ones offer a limited mobility that requires using other parts of the body to exert forces to actuate each dof. The fully actuated prostheses encounter both technical problems, due to the need of capturing the user intention, and psychological problems, due to the acceptability of the prosthesis. Often the user generates the command signal with myoelectric control [19, 25], but this method today applies only to a few dof. In the worst situation, a patient with an inter-scapular-thoracic amputation has lost all the seven degrees of freedom of the arm, and usually can control only elbow and wrist flexion, and hand closing [18]. Even though the controller requires extensive input from the user who usually moves one joint at a time and needs a long training time.

As reported in [4] up to now the most effective driving signals acquired from the patients have been based on the head motion. For instance the target to be reached by the hand is maintained on the axis, fixed to the head, perpendicular to the line connecting the eyes, while another head lateral movement drives the distance from the target to the face. A device including accelerometers and gyroscopes, positioned on the head, detects the head motion. This solution can be integrated with voice-activated movements.

We need to mention that, to deal with cognitive burden and user acceptance, an invasive surgical technique was conceived, called targeted muscle reinnervation (TMR) [15]; a residual muscle is enervated and then reinnervated with residual nerves of the amputated limb. After nerve growing, the EMG signals of the targeted muscle can be used to control a prosthetic device. However, classifying shoulder movements in TMR has generally been ignored to date.

Our goal here is to show a novel interface for the user of total upper limb prosthesis. It extends the EMG classification control to include other signals from the user. First we had to approach the novel task of classifying shoulder movements from sEMG signal, then to develop a vision system to find the target position. We are considering in particular grasping activities, the ones that are of primary importance for every day life.

In the past we developed control strategies for hand prosthesis [8, 17] and verified the real time performance of the EMG classifier [9]. In the present study we illustrate this new concept for a user interface in total upper limb prosthesis that extends the myo-electric control integrating vision analysis and trajectory computation: the sEMG signals indicate the shoulder movement to start, the visual analysis computes the target location on the object the user is locking at, and finally the trajectory to the grasping point is computed.

The paper is so organized. Section 2 presents an overview of the interface MyArm system (Fig. 1) we designed. Sections 3–5 give more details about the sub-modules that cooperate to obtain the controller. Section 6 summarizes the results and proposes possible future developments.

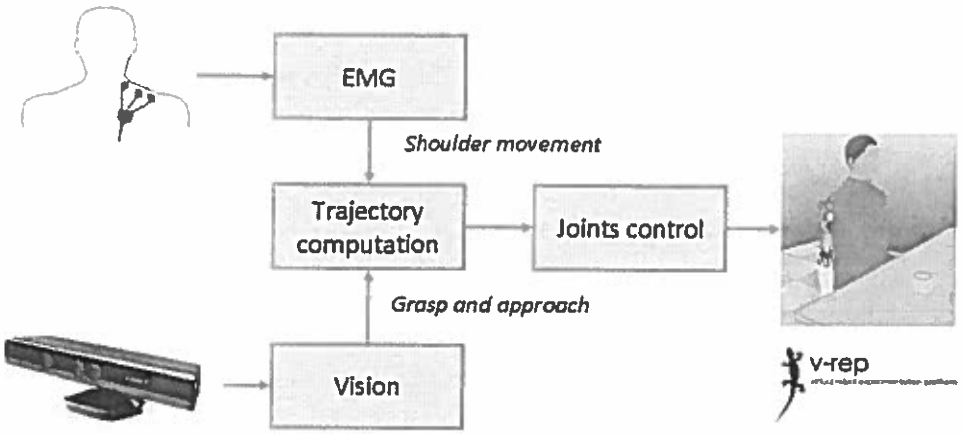

Fig. 1 The modules of MyArm: EMG, vision, trajectory generation and control

2 Design Principles of MyArm

The control system "MyArm" integrates different modules as in Fig. 1. Three modules give input to the joint controller: EMG classification, vision, and trajectory computation. The EMG module in MyArm adopts the pattern recognition-based control, in Fig. 2, that involves computing the subsequent stages here briefly described.

1. *Data segmentation* Each channel has to be segmented into a series of time windows, adjacent or overlapped. The window length has to satisfy real-time constraints that require the actuation delay must not be greater than 300. Smith [26] suggested that the optimum window length is between 150 and 250 ms; Phinyomark [20] highlighted that the best result in term of robustness is obtained with a length of 500 ms and an increment of 125 ms. The processing of myoelectric signals can be done including transients (i.e. at the contraction

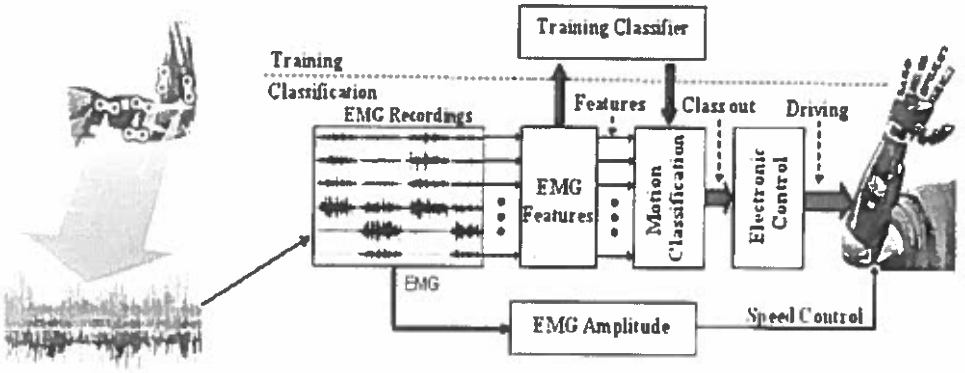

Fig. 2 The myoelectric based controller

onset) or at steady states only (i.e. during maintained contraction); Englehart [7] showed that steady state data are classified more accurately than transient data.
2. *Feature representation* This phase involves two stages: feature extraction and feature reduction. Generally, features belong to three main domains: time, frequency, and time-frequency [13]. The second stage is employed to reduce the dimensionality of the initial feature space, attempting to preserve the classification accuracy, while reducing the computational costs of the classifier [14]. Dimensionality reduction strategies may be either feature selection or feature projection, where a new feature subset is created combining the original features through a linear or non-linear mapping. Previous studies have shown that the projection approach leads to a higher discrimination ability compared to feature selection [12].
3. *Classification* The classifier receives as input the reduced feature set and it has to match the different patterns with the correct movement class [6].

Considering again Fig. 1, the vision system has to detect the object and compute the grasping point. It has to look at the scene, recognize the object the user is looking at, and compute the approach position to grasp it. The first choice was the hardware: given the fundamental need for a depth map, the only options were a stereo camera or a device for measuring the distance of every point. In 2010, Microsoft released an inexpensive sensor for the Xbox 360 gaming console, called Kinect, able to generate a point cloud combining a RGB camera and two IR depth sensors (using structured light scanning). Later on, PrimeSense introduced open source drivers, allowing new developers to use it for non-gaming contexts. In light of this, the Microsoft Kinect [11] was adopted. Moreover, several libraries concerning image analysis and 3D point clouds exist: while OpenCV is more based on two dimensional images, a novel open source project named Point Cloud Library (PCL) suited best our case. PCL is written in C++ and released under the BSD license. Thus, with little alterations, various algorithms needed for our goal were available in it. Our vision system makes use of the acquired points cloud, without the need of image analysis. After training on a number of objects, the system is able to compute the approach point to grasp objects [21, 24] of similar geometry and functionality.

After those points a trajectory for elbow and wrist can be computed and sent to the motor control. It is foreseen that a miniaturised kinect device will be mounted on special glasses able to compute the focus of attention of the user when looking at the object to take.

The entire MyArm system has been developed in the V_REP[1] frame. This open source environment provides specific algorithms for trajectory planning and obstacle avoidance, and can send the output both to a graphic simulator window and to the real robot. This framework will be valuable also for training the patient in using the prosthesis.

[1] v-rep by Coppelia Robotics. http://www.coppeliarobotics.com.

3 The EMG Module

Aim of this study was to experimentally verify whether a pattern recognition approach could be useful to classify a quite large number of shoulder movements. To this purpose we have implemented a number of classification algorithms and have compared the discrimination capabilities of different feature extraction methods. Then we have checked the possibility to reduce the number of muscles to be analyzed without losing classification accuracy.

In the last decades, the myoelectric prostheses control underwent a significant improvement after the introduction of sEMG pattern recognition strategies [13]. The pattern recognition control approach is founded on the assumption that patterns of sEMG signals from several muscles include information about the intentional movement of the amputated limb. The aim of this method is to match each sEMG pattern to one motion class among a multiplicity of preselected movements. Then the chosen movement is automatically performed in a pre-programmed mode by the prosthetic device.

Our EMG classifier has to detect shoulder movements. Unlike the conventional myoelectric control, our method exploits the contraction of several synergistic muscles related to the movement the amputee wishes to perform, and does not require that a single degree of freedom is independently controlled. In this way, a more intuitive and rapid control can be obtained.

Since there is no literature about classification of shoulder movements, we devised an experiment to collect and analyze sEMG data to verify the viability of our classifier.

The experimental data were collected from eight healthy subjects (four males and four females) aged 25.0 ± 1.8 years, informed about the experimental procedures. They performed a series of eight shoulder movements, chosen among the most common, that were repeated ten times each. Every movement included four sequential phases: resting, elevation, isometric holding, and return to the rest condition. All the movements have been executed with the arm fully extended. The planes considered are the usual frontal and sagittal planes, and the vertical plane rotated of 45° from the sagittal plane, as in Fig. 3. The eight movements are defined starting from the orthostatic position—with parallel feet at 12 cm distance, arms along the body with palms inside:

Fig. 3 Two of the movements: shoulder abduction in the *frontal plane*, and *top view* of shoulder elevation in a plane rotated 45° from the *sagittal plane*

- 4 flex/extensions in the sagittal plane to 45°, 90°, 110°, and −30°;
- 2 ab/adductions in the frontal plane to 45° and 90°;
- 2 elevations in the rotated plane to 45° and 90°.

We focused on these movements because usual prosthetic solutions for shoulder disarticulation can reasonably control only 2 degrees of freedom. For this reason humeral internal/external rotation was not considered.

Eight pairs of electrodes were placed over the following trunk muscles: clavicular and sternal heads of the pectoralis major, serratus anterior, trapezius descendent, trapezius transversalis, trapezius ascendent, infraspinatus, and latissimus dorsi. These muscles are indeed synergistic in the analyzed movements and they are superficial muscles preserved also after upper limb amputation. SEMG data were recorded by bipolar disposable electrodes (26 mm in diameter) connected to miniaturized, wireless probes for digital data collection. Each pair of electrodes was placed according to the guidelines provided by [2] with an interelectrode distance of 26 mm. SEMG signals were recorded at a sampling rate of 1.0 kHz and processed in Matlab.

Data analysis considered the only steady state phase (isometric hold) that is about 3 s. Six signal segmentation settings were tested, by combining different window sizes (L) and increments (I): (a) L = 500 ms, I = 250 ms, (b) L = 500 ms, I = 125 ms, (c) L = 500 ms, I = 62 ms, (d) L = 250 ms, I = 250 ms, (e) L = 250 ms, I = 125 ms, and (f) L = 250 ms, I = 62 ms. After a random shuffling of all acquired trials from the 8 subjects, data were split 60 % into a training dataset, and 40 % in a test dataset. The training set was used to train the classifier; the test set was employed to estimate the classification accuracy.

For each of the six segmentations tested, different feature sets were extracted and reduced using principal component analysis (PCA) to produce a new feature set by a linear projection of the original feature vector onto the eingenvector of the covariance matrix.

Classification was performed using linear discriminant analysis (LDA). This classifier was chosen due to its high performance in EMG signal classification, low computational cost, ease of implementation, high speed training and robustness, as reported in [12].

The performance of the classification system was defined in terms of classification error and was evaluated by considering three different sets of motion classes.

1. Nine motions as described above;
2. Five motions: shoulder extensions in the sagittal plane to 90° and −30°, shoulder abduction at 90°, shoulder elevation at 90° along the rotated plane, and rest;
3. Four motion classes: as before after removing shoulder elevation.

The feature set that gave the best results is the one proposed in [20], and includes sample entropy (SampEn), cepstral coefficients (CC) of the 4th order, root mean square (RMS), and WL. Its results on the test set are in Table 1. This best performance was obtained with L = 500 ms, I = 62 ms.

Table 1 The classification error for 4, 5, 9 classes

	Error (%)
4 classes	0.00
5 classes	2.61
9 classes	7.74

Table 2 The classification accuracy using 6 or 8 acquisition channels

	4 classes (%)	5 classes (%)	9 classes (%)
6 channels	99.98	96.11	88.31
8 channels	100	97.39	92.56

After observing that the classifier has very good accuracy, we investigated the effect of reducing the number of sEMG channels. With 6 channels, a small degradation in accuracy [10] is illustrated in Table 2.

To make the classifier answer available with a reduced number of electrodes, we can observe that a 4 classes classifier with 6 acquisition channels has still a very high accuracy. Moreover, to reduce the initial delay, we can consider using $L = 250$ ms, $I = 62$ ms, that we have found to produce a classification accuracy of 99.11 % for 4 classes. For more data and more details please refer to [22, 23].

4 The Vision Module

Picking up an item using an autonomous robot is a challenging and fairly novel task. It has been studied from a psychological, biological and engineering focus, without reaching satisfying results. Therefore, the ultimate aim is to provide the user (or the application) with the most suitable grasping points for the given item(s). It uses a training method were objects are shown to the system.

For each object we intend to generate only two grasping points—related to the centre of mass of the given cloud. These points are two because most of the artificial hands act like a gripper. The whole system presents a modern approach to a task that was never completely solved, adopting some state of the art algorithms and an inexpensive, effective and brand new hardware device.

The scenario where the system was designed and tested is that items are placed on a surface and they are not moving. At the same time, the camera, attached to the user head, is almost static too, and it faces them. The training and testing stages are performed without changing the pose of the camera with respect to the plane. Lighting conditions do not affect the outcome at all. Our vision module is based on a classifier to recognize the kind of object to grasp based purely on data collected from Kinect.

The main steps in analysing Kinect data both in training and testing are five: generate the point cloud, remove the table points, compute the nearest neighbours, cluster the cloud, and compute the barycentre and the grasping points. All the steps

make use of an efficient representation based on k-d trees [3], a space-partitioning data structure for organizing points in a k-dimensional space. The classifier we built makes wide use of this structure that allows search, insert, and delete operations to have on average O (n log n) time complexity.

1. *Generate the point cloud* A point cloud is a set of unorganized, irregular points in 3D. To generate point cloud from depth image captured by the Kinect one may convert from (u, v, d)—the first two coordinates are the x/y position, while d is the distance provided by the IR sensor—to the point cloud, which represents points using the standard (x, y, z) coordinates.
2. *Remove the table* In order to identify the main objects placed in front of the Kinect, we have to cluster the scene, under the assumption that there is a plane (i.e. a table) where the items are placed on. Thus this table has to be removed from the scene, using the RANSAC (RANdom SAmple Consensus) method [27].
3. *Find neighbours* 3D feature estimation methodologies are needed. In general, PCL uses approximate methods to compute the nearest neighbours of a query point, using fast k-d tree queries. Once determined, the neighbouring points of a query point can be used to estimate a local feature representation that captures the geometry of the underlying sampled surface around the query point.
4. *Clustering* If we assume that the planar structure has been removed, clustering the remaining items is a necessary step to separate the individual object point clusters lying on the plane. We used the k-nearest neighbour algorithm (k-NN) [1], a method for classifying objects based on closest training examples in the feature space. k-NN is a type of instance-based learning, where the function is only approximated locally and all computation is deferred until classification. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common amongst its k nearest neighbours (k is a positive integer). If k equals 1, then the object is simply assigned to the class of its nearest neighbour. The FLANN library[2] has been used for clustering.
5. *Compute the grasping position* The final task is to estimate two points where the hand could firmly grasp and hold the item. Since the prosthetic hand we expect to use is a simple gripper, the approach of [19] suits our case. It requires to compute the item's centroid, which is the intersection of all straight lines that divide the object into two parts; then to compute the two grasping points P1 and P2, as in Fig. 4. Since we require the object to be held firmly, the centre of mass must lie on the straight line that connects P1 with P2 and the first of the two is the closest boundary point to the centroid. Issues arise when dealing with P2: since it lies on the line connecting P1 with the barycentre, it isn't part of the cloud and it wasn't visible to the camera, therefore it has to be estimated. Since we know the type and the orientation of the object, we define a parameter p that means "how far" P2 is with respect to the other two points. For most of the objects, given their symmetries, p should be equal to 1; however, the Kinect

[2]http://www.cs.ubc.ca/research/flann/.

Fig. 4 Computing grasp in theory and on examples of the point cloud

usually sees a small portion of the object and the centroid is biased towards the visible points. As a consequence, P2 is a little farther than one would expect. In other cases such as when dealing with a cup, some points of its inner side are visible, therefore p = 1 works flawlessly.

The training phase consists of the following: place the object in front of the camera, compute its pose, rotate it and collect many poses, store in a k-d tree. All the computations are the same for either the training or the recognition phases. Given a set of training data, the nearest neighbour search returns a set of potential candidates with sorted distances to the query object. During testing we allow in the scene more than one object. Once they have been extracted as clusters, their nearest neighbours are computed. Lastly, the two grasping points are computed.

Classifiers have to be tested on a different test set to estimate the ability to generalize; we assess the goodness using accuracy [10]. An attempt to classify a new object placed in front of the Kinect can either yield a correct or a wrong prediction. The latter case can be subdivided into completely wrong classifications and those where the item was correctly identified, but not the pose.

A number of factors affect the outcome of the test, as the role of position and size of the objects, the number of items in the scene, possible overlapping, distance from the camera, number of different poses used in training, number of different items known to the system, and difference between the item used in training with the one used during testing. Consider that the Kinect sensor has a practical ranging limit of 0.6–3.5 m: the camera doesn't sense items closer, while farther ones are rendered with a poor number of points. Thin portions of objects are barely caught by the infrared grid and make the pose estimation extremely hard; for instance, the handle of a cup isn't recognized most of the times.

Ten items with 82 poses were used to train and test the system. The objects, in Fig. 5, include a cup with a handle, a deodorant spray can, a book, an alarm clock, a cardboard tube, an eyeglass case, a cube, an apple, a shoe, and a toilet paper roll. The parameter p has been computed using several empirical tests for every item. Results are in Table 3 for 4 distances and 15 tests per object per distance. Other trials considered also low overlapping and different number of objects in the scene without finding significant downgrade of the performance.

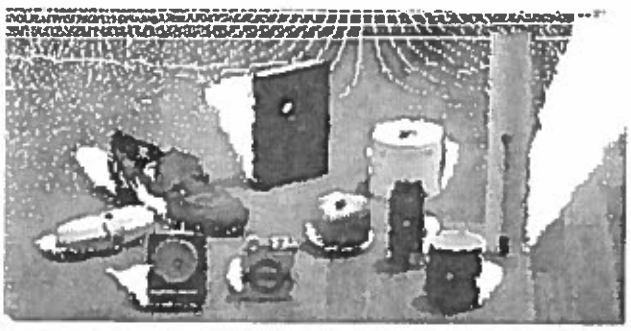

Fig. 5 The objects used for training

Table 3 Recognition statistics of the testing phase

Distance (m)	Recognition object (%)	Recognition pose (%)
0.6	100.0	100.0
1.1	100.0	93.3
1.6	100.0	86.6

5 Trajectory Generation Module

The trajectory generation considers the recognized shoulder movement and the grasp point to reach. Since there is a plane and possibly other point clouds recognized in the system, it has to compute a collision free trajectory to avoid the table and other objects.

The model of the prosthesis has been defined in the D-H notation, considering the mechanical system on the left of Fig. 5 that represents the prosthesis developed at Department of Mechanics in Politecnico di Milano [4]. There are 6 axes in the prosthesis: A for shoulder flexo-extension; B for the abduction of the shoulder; C for the axial rotation of the arm, obtained with a passive joint to adapt to external forces; D for the flexo extension of the lower arm; E for wrist prono-supination; F for wrist flexo-extension.

The direct kinematics is simply obtained after multiplication of the joints matrices and premultiplication of the matrix expressing the shoulder position with respect to Kinect. The inverse kinematics is computed in closed form using a geometric approach.

Trajectory computation requires two inputs: the EMG signal indicating the movement to do, and the point to reach computed from Kinect data. The trajectory computation generates the via points (at least approach and grasping) and sends them sequentially to the controller while monitoring the EMG input, that can change since the user may decide to terminate a task before the object is reached.

Considering the limitations in the wrist angles, 3 approaches to the grasp point are computed frontal, from top, from side. For each of them after setting the values for the wrist angles, we compute the wrist position and then the other joints values with the inverse kinematics.

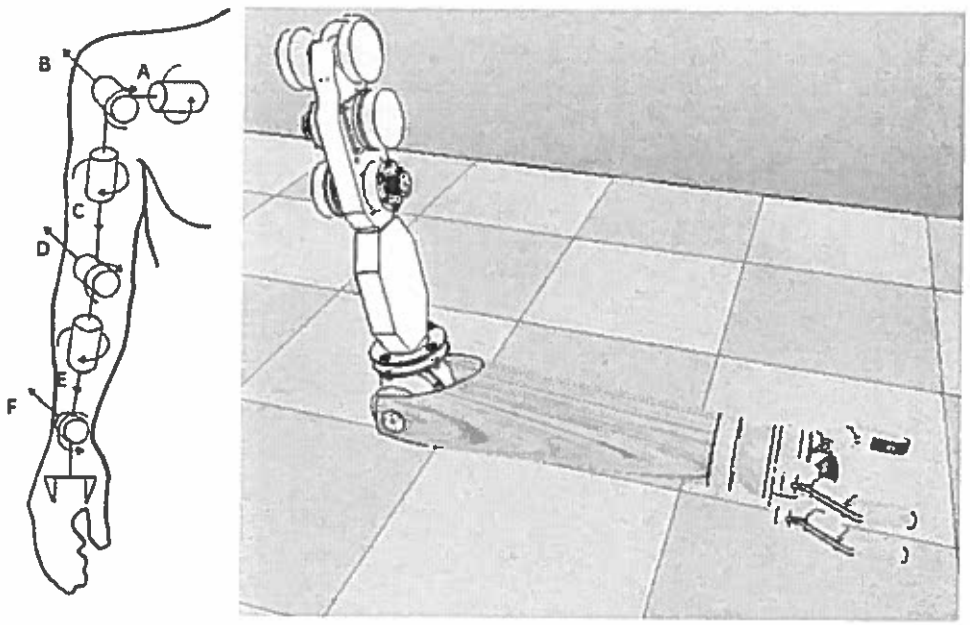

Fig. 6 The structure of the arm prosthesis including the hand, and its V-REP representation

All the computations are done in v-rep using the remote API (Application Programming Interface) approach that allows the user to control the simulation from remote hardware or software. In V-REP the model of the prosthesis is the one illustrated in Fig. 6 on the right. Since V-REP offers controllers and algorithms the implementation is straightforward.

6 Results and Conclusions

The system has been conceptually developed on the basis of a prototype prosthesis whose construction is under way at the Mechanical Department of Politecnico di Milano.

The overall system integration in v-rep has demonstrated the solution to be valuable in terms of precision of reaching. The error in Cartesian space is about 6 mm. Instead the time needed by the vision module is about 7 s, too much to integrate directly our vision solution in the controller.

As an important finding, the independence from the prosthesis is very high. The V-REP simulator solution offers an efficient way for the patient training without using the real prosthesis.

However implementing the real prosthesis has still some open issues: miniaturize the hardware, and better software solutions. Our system is a proof of concept

that still needs engineering to become fully usable. Further activities are under way to adapt the architecture to the case of exoskeletons of the arm, where the role of the vision system can be reduced or even eliminated.

References

1. Altman NS (1992) An introduction to kernel and nearest-neighbor nonparametric regression. Am Stat 46(3):175–185
2. Barbero M, Merletti R, Rainoldi A (2012) Atlas of muscle innervation zones. Springer, Milan, Italy, pp 103–120
3. Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509
4. Casolo F (2010) Elbow prosthesis for partial or total limb replacements. In: Casolo F (ed) Motion Control, Chapter 16. InTech (2010)
5. Corbett EA, Perreault EJ, Kuiken TA (2011) Comparison of electromyography and force as interfaces for prosthetic. J Rehabil Res Dev 48:629–642
6. Englehart K, Hudgins B (2003) A robust, real–time control scheme for multifunction myoelectric control. IEEE Trans Biomed Eng 50(7):848–854
7. Englehart KB, Hudgins B, Parker PA (2001) A wavelet-based continuous classification scheme for multifunction myoelectric control. IEEE Trans Biomed Eng 48(3):302–310
8. Gini G, Arvetti M, Somlai I, Folgheraiter M (2012) Acquisition and analysis of EMG signals to recognize multiple hand movements for prosthetic applications. Appl Bion Biomech 9:145–155
9. Gini G, Cavazzana L, Mutti F, Belluco P, Mauri A (2014) New results on classifying EMG signals for interfacing patients and mechanical devices. In: Rodic A, Pisla D, Bleuler H (eds) New trends in medical and service robots. Springer, vol 20 Mechanisms and Machines Sciences series (2014)
10. Han J, Kamber M, Pei J (2005) Data mining: concepts and techniques, Second Edition. Morgan Kaufmann, San Francisco
11. Han J, Shao L, Xu D, Shotton J (2013) Enhanced computer vision with microsoft kinect sensor: a review. IEEE Trans Cybernetics 43(5):1318–1334
12. Hargrove L, Losier Y, Lock BA, Englehart K, Hudgin B (2007) A real-time pattern recognition based myoelectric control usability study implemented in a virtual environment. In: Proceedings 29th annual international conference on IEEE EMBS. Lyon, France, pp 4842–4845 (2007)
13. Hudgins B, Parker P, Scott R (1993) A new strategy for multifunction myoelectric control. IEEE Trans Biomed Eng 40(1):82–94
14. Jain AJ, Duin RPW, Mao J (2000) Statistical pattern recognition: a review. IEEE Trans PAMI 22(1):4–37
15. Kuiken A, Li G, Lock BA, Lipschutz RD, Miller LA, Stubblefield KA, Englehart KB (2009) Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms. JAMA 301(6):619–628
16. Lai JCK, Schoen MP, Perez Gracia A, Naidu DS, Leung SW (2007) Prosthetic devices: challenges and implications of robotic implants and biological interfaces. In: Proceedings of IMechE, vol 221, pp 173–183 (2007)
17. Lisi G, Cattaneo D, Belluco P, Gini G (2011) From the classification of EMG signals to the development of a new lower arm prosthesis. In: Proceedings of IFAC 18 world congress. Milan, pp. 6493–6498 (2011)
18. Miguelez JM, Miguelez, MD, Alley RD Amputations about the shoulder: prosthetic management, Chapter 21. American Academy of Orthopaedic Surgeons (online)

19. Parker P, Englehart K, Hudgins B (2006) Myoelectric signal processing for control of powered limb prostheses. J Electromyogr Kinesiol 16(6):541–548
20. Phinyomark A, Quaine F, Charbonnier S, Serviere C, Tarpin-Bernard F, Laurillau Y (2013) EMG feature evaluation for improving myoelectric pattern recognition robustness. Expert Syst Appl 40(12):4832–4840
21. Richtsfeld M, Vincze M (2008) Grasping of unknown objects from a table top. workshop on vision in action: efficient strategies for cognitive agents in complex environments. Marseille, France
22. Rivela D, Scannella A (2013) Classificazione del segnale sEMG tramite pattern recognition per il controllo del giunto della spalla di una protesi attiva di arto superiore, Master Thesis, Politecnico di Milano
23. Rusu RB, Cousins S (2011) 3D is here: point cloud library (PCL). In: Proceedings of IEEE ICRA
24. Saxena A, Driemeyer J, Ng AY (2008) Robotic grasping of novel objects using vision. Int J Robot Res 27(2):157–173
25. Shenoy PK, Miller J, Crawford B, Rao RPN (2008) Electromyographic control. IEEE Trans BME 55(3):1128–1135
26. Smith LH, Hargrove LJ, Lock BA, Kuiken TA (2011) Determining the optimal window length for pattern recognition-based myoelectric control: balancing the competing effects of classification error and controller delay. IEEE Trans Neural Syst Rehabil Eng 19(2):186–192
27. Yang MY, Forstner W (2010) Plane detection in point cloud data. University of Bonn