RESIDUAL SHOULDER MOTION VECTOR PROJECTION

Yves Losier, Kevin Englehart and Bernard Hudgins
Institute of Biomedical Engineering, University of New Brunswick, Fredericton, Canada

ABSTRACT

Providing an externally powered prosthetic solution for high-level upper extremity amputation cases requires several design factors to be considered by the clinical fitting team. The sensors used must often be finely adjusted during the prosthesis fabrication in order to provide consistent repeatable control input signals. The work presented in this paper introduces a mathematical framework capable of implementing robust residual shoulder motion driven control using only a short preliminary training protocol in an attempt to remove issues such as sensor type, alignment, and range of motion. The algorithm determines the output amplitude values based on the real time residual shoulder position to reliably drive the prosthetic limb's actuators. This algorithm provides a new control input option for designing an externally powered prosthetic solution for high-level amputation cases.

INTRODUCTION

Previous research has shown residual shoulder motion to be a useful input source for various prosthetic control strategies [1-3]. Its importance is often amplified for high-level amputation cases where the availability of robust input control sources is often limited. The most common use of residual motion is to drive cable-operated joints. This body-powered method has been in use for several decades and is the most clinically available option at this point in time. Externally powered systems do exist which can use sensors, such as force sensing resistors, joysticks, and rocker switches that are activated by the user's residual shoulder motion. Single or dual site myoelectric signal originating from the residual limb and shoulder complex can also be used as a switch or to drive a specific actuator [4,5].

The selection of sensors and the control scheme by the clinical team will heavily depend on the consideration of several design factors (patient's musculature condition, range of motion, learning ability, etc) in order to obtain an appropriate prosthetic rehabilitation plan [6]. Other design issues such as sensor orientation and output range also requires some consideration prior to the fabrication of the prosthesis. Some level of final adjustments and modifications are often required with any devised solution. Ideally, it would be beneficial to have an initialization protocol by which some of these factors would be taken into consideration and their associated implementation complexity removed from the prosthetic rehabilitation design stages. Automatic tailoring of the system for factors such as the user's range of motion, the sensor type, positioning and output range would also speed up the setup time required within a clinical and/or system retraining setting. The algorithm presented addresses such issues by adapting the actuator output calculations based upon data collected during a short training session once the prosthesis has been fabricated.

METHODOLOGY

The fundamental basis of the algorithm consists of three stages: 1) creating *class specific* vectors based on training data, 2) determining the projected interim class values by relating a real time input signals based vector to these class vectors, and 3) calculating the class outputs using these values along with algorithm parameters. The first stage is performed automatically immediately following the training session while the latter two stages are executed in real time.

<u>Training Protocol</u>

Users are instructed to complete five shoulder motions: elevation, protraction, depression, retraction, and a no movement/rest class. Each motion is held for one second and the entire set is repeated five times. The data collected during the training session provides the information necessary to determine the average position for each class, termed *class centroids*, within the input signal space. These centroids are treated as localized vectors (Figure 1) to produce the *class specific* vectors (Figure 2). These class vectors are created using the rest class as the origin where X ∈ {Elevation, Protraction, Depression, Retraction} denotes one of the four vector classes:

$$\vec{V}_{Rest\,X} = \vec{V}_{O\,X} - \vec{V}_{O\,Rest} \tag{1}$$

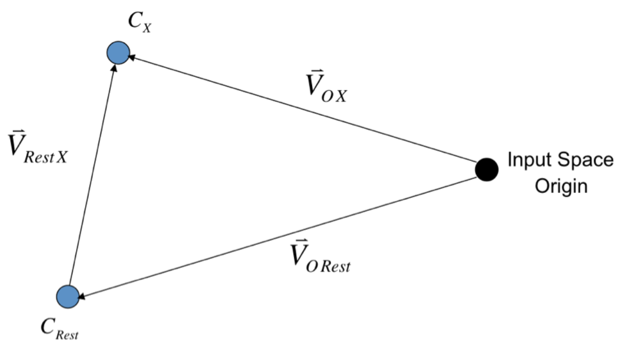

Figure 1: Class Centroids and Vector Diagram Within Input Space

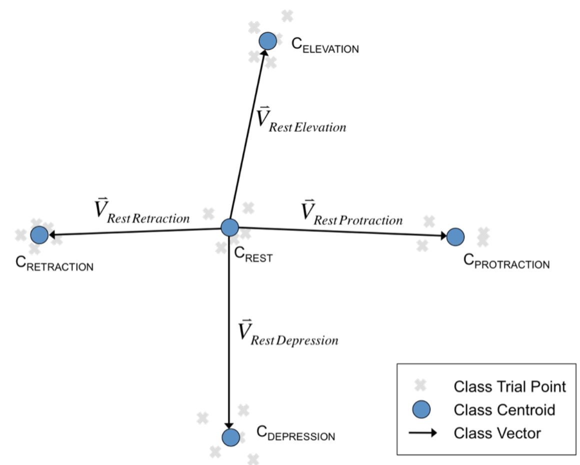

Figure 2: Complete Class Specific Vector Diagram for the Vector Projection Algorithm

Having created the class vectors with the training data, it is now possible to calculate the magnitude component for each class vector along with the angle between two adjacent vectors:

$$\left|\vec{V}_{Rest\,X}\right| = \sqrt{\sum_{i=1}^{N}\left(C_X - C_{Rest}\right)_i^2} \qquad (2)$$
$$\textit{where } N \equiv \textit{input space dimension}$$

$$\Delta\theta = \cos^{-1}\left(\frac{\vec{V}_{Rest\,X1}\cdot\vec{V}_{Rest\,X2}}{\left|\vec{V}_{Rest\,X1}\right|\left|\vec{V}_{Rest\,X2}\right|}\right) \qquad (3)$$

Input Signal Projection onto Class Vectors

The second stage of the algorithm requires the use of the current sensor signal values in order to create one final vector, termed *input vector*, using equation (1). Similarly, its magnitude and the angles between it and the adjacent class vectors can also be calculated using equations (2) and (3) respectively.

The algorithm requires that the newly computed input vector magnitude be normalized using each class vector's magnitude to compensate for the user's residual range of motion for each class:

$$d_X = \frac{\left|\vec{V}_{Rest\,Current}\right|}{\left|\vec{V}_{Rest\,X}\right|} \qquad (4)$$

The projected value, $\boldsymbol{d_X}$, then represents the normalized input vector magnitude for each of the four principal classes.

Class Strength Outputs and Tuning Parameters

The final stage of the algorithm uses the previously calculated values along with two tunable parameters (***TF*** and ***SF***) to determine the four class strength outputs:

$$\omega_X = \alpha\left(d_X, TF\right)\delta\left(\theta_X, SF\right) \qquad (5)$$

where α and δ are defined as:

$$\alpha = \begin{cases} 0 & , \quad d_X < TF \\ \dfrac{d_X - TF}{1 - TF} & , \quad TF < d_X < 1 \\ 1 & , \quad d_X > 1 \end{cases} \qquad (6)$$

$$\delta = \frac{\cos\left(\dfrac{\theta_X}{\Delta\theta}\,SF\,\pi\right) + 1}{2} \qquad (7)$$

The magnitude coefficient, α, represents the adjusted input signal's projected value, $\boldsymbol{d_X}$, based on the threshold factor, ***TF***. This coefficient is required to provide a '*deadzone*' area for the Rest class (Figure 3). This area ensures that no class outputs are active within the desired zone. It should also be noted that current implementation of the magnitude coefficient ensures that no discontinuities will occur when crossing the boundary between the rest and 'active' regions.

The offset coefficient, δ, reduces the effective output strength of a given class as the angle between the input signal and class vectors increases. The spread factor, ***SF***, dictates how quickly the value will diminish as the angle increases.

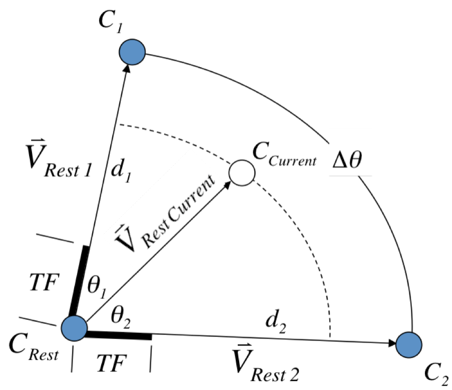

Figure 3: Illustration of Threshold Factor based *deadzone* for the Vector Projection Algorithm

Figure 5: Experimental Bypass Socket with Linear Transducer Inputs used for the Evaluation of the Vector Projection Algorithm

ALGORITHM EVALUATION

Three separate case studies were investigated in a preliminary attempt to evaluate the effectiveness, reliability and versatility of the vector projection algorithm. The first two cases used a two axis joystick as the input signal source (Figure 4) while the third case study used two linear transducers mounted on an experimental bypass socket (Figure 5).

The first study consisted of orienting the joystick such that one of its axes was vertical while the other horizontal. The subject performed the training protocol previously described in this paper and then proceeded to qualitatively assess the algorithm performance using class output feedback displayed in a Matlab graphical user interface (Figure 6). A similar setup was used for the second study with the exception of the joystick being rotated by 45 degrees. The final case study differed since the input signals originated from two linear transducers rather than a joystick.

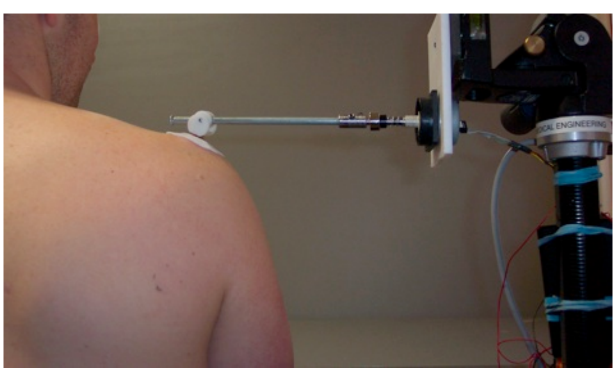

Figure 4: Experimental Joystick Apparatus used for the Evaluation of the Vector Projection Algorithm

The subject was additionally asked to remove and re-don his bypass shoulder socket during the course of the experiment. The algorithm was reassessed without any retraining once the socket was reattached.

DISCUSSION

The algorithm appeared to perform robustly in all cases evaluated. The users reported no non-elicited class activation during the course of the experiment. Additionally, users enjoyed the ability to have dual activation of adjacent classes. This effect could be amplified or reduced by tuning the spread factor, ***SF***. It was also observed that both tuning parameters can be easily and intuitively adjusted by the clinical team, following the training session, to tailor the system to the user's preferences.

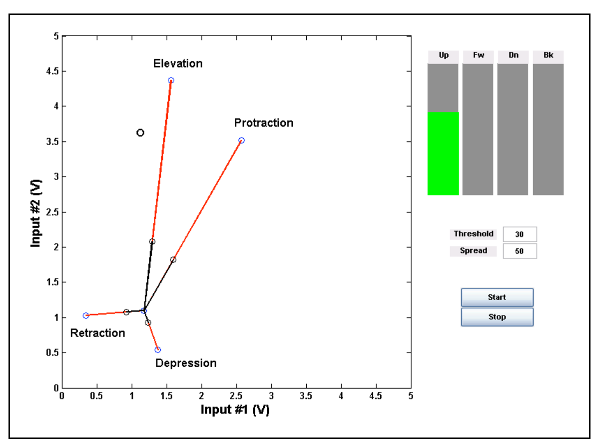

Figure 6: Matlab Graphical User Interface Feedback Diagram

CONCLUSION

A mathematical framework has been devised and experimentally implemented to provide a robust residual shoulder motion based control option for a high-level externally powered prosthesis. This algorithm removes several complexities of prosthetic fitting by using a short preliminary training session to tailor the system to both the prosthesis setup and user. The preliminary case studies presented have demonstrated that robust proportional class outputs can be achieved using this algorithm for different prosthetic design scenarios.

Further quantitative research is currently ongoing to evaluate the usability of this algorithm when combined with an endpoint control strategy. Ultimately, the usefulness of this scheme must be assessed with an actual prosthetic fitting to properly evaluate its ability to intuitively and reliably enhance the prosthetic user's ability to perform tasks of active daily living. Work is currently ongoing in conjunction with the clinical fitting team at the Institute of Biomedical Engineering at the University of New Brunswick to achieve this goal.

ACKNOWLEDGEMENTS

This work was supported by NSERC Discovery Grants 171368-03 and 217354-01, the Research Assistantship Initiative (RAI) of the New Brunswick Innovation Foundation (NBIF), and the Atlantic Innovation Fund (AIF).

REFERENCES

[1] D.C. Simpson, "The choice of control system for the multimovement prosthesis: extended physiological proprioception (EPP)," *The control of upper-extremity prostheses and orthoses,* 1974.

[2] N. Storey, "Control of an arm prosthesis," *Ph.D. thesis*, University of Southampton, UK, 1977.

[3] J.A. Doubler, D.S. Childress, "An analysis of extended physiological proprioception as a prosthesis control technique," *Journal of Rehabilitation Research and Development*, vol. 21, no. 1, pp. 5-18, 1984.

[4] C. Lake, J.M. Miguelez, "Comparative analysis of microprocessors in upper limb prosthetics," *Journal of Prosthetics and Orthotics*, vol. 15, no. 2, pp. 48-63, 2003.

[5] C. Lake, J.M. Miguelez, "Evolution of microprocessor based control systems in upper extremity prosthetics," *Technology & Disability*, vol. 15, no. 2, pp. 63-71, 2003.

[6] J.M. Miguelez, "Critical factors in Electrically powered upper-extrimity prosthetics," *Journal of Prosthetics and Orthotics*, vol. 14, no. 1, pp. 36-38, 2002.