Human Knee Simulation Using CMAC ANN

R. A. Lima[1], V. R. F. S. Marães[1], J. P. Martins [1] and L. M. Brasil[1]

[1] University of Brasília-UnB at Gama-FGA, Brasília, Brazil

Abstract— **This paper aims to show the use of a CMAC (Cerebellar Model Articulation Control), a kind of ANN (Artificial Neural Network). The CMAC is based on cerebellum of mammals, but despite this characteristic, actually, what promotes its use, is its very fast operation, which makes it suitable for adaptive control in real time. This type of control is needed, for example, to control an active transfemoral prosthesis. Simulation of knee angular velocities, based on collected data from the contralateral knee, is presented. The simulation is available as open source software.**

Keywords— **CMAC, machine learning, knee, simulation.**

I. INTRODUCTION

The quest to improve the life quality is constituted as one of the prerogatives of Biomedical Engineering. Rehabilitation discipline is supported by biomedical engineering, for example, in prostheses constructions. Prostheses can be passives or actives [1]. Passive form use intrinsically passive actuators and active form have automatic actuators.

The construction of an active transfemoral prosthesis is not a trivial problem solution. To build them is required before to develop or to use a biomechanical model. From this, one must create a control method using control engineering techniques and / or intelligent systems.

The rise of intelligent systems brought a new landscape for Biomedical Engineering. This kind of solution allows previously unimagined system classes. It is possible to build intelligent systems which are not solvable easily by traditional PID (Proportional-Integral-Derivative), with many variables, many parameters and nonlinears. Intelligent systems are everywhere, in digital cameras, Internet search engines, speech recognition systems, cell phones, car brake systems and countless other devices.

Among the techniques for building intelligent systems, there are the ANNs (Artificial Neural Networks). This kind of system is too classified as machine learning based system [2]. They are systems which learn to figure out complex problems during the learning phase, through historical collected data, without the necessity of complex physical models. The CMAC (Cerebellar Model Articulation Control) [3] is a ANN inspired on cerebellum of mammals [4], which compared with another kind of ANN called MLP (Multi-Layer Perceptron)[5], with at least one hidden layer, has the advantage of needing much less calculations to update their weights during training [6].

Lin shows in [7], the effectiveness of CMAC model, by preliminary studies of kinematic control and gait synthesis. After training an ANN based on CMAC, to learn multivariable and nonlinear relationships kinematics of quadruped gait, the ANN was used to control straight and uphill walk of quadruped robots.

The [8] depicts the robustness of a CMAC in a biped robot running in conditions presenting disorders. The [9] presents strategies for using the CMAC in same type of robot.

Before one can build a prosthesis controller, is advisable first to simulate their behavior via software. This idea can facilitate prototypes production. In this context, the paper illustrates a human knee angular velocities simulation, using machine learning based CMAC model for possible knee control in a transfemoral active prosthesis.

II. MATERIALS AND METHODS

The simulation process is summarized in Figure 1.

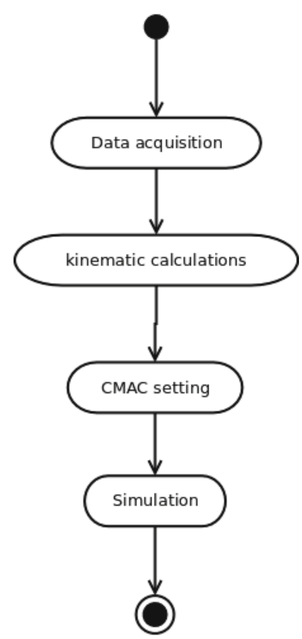

Fig. 1: Simulation Process

D.A. Jaffray (ed.), *World Congress on Medical Physics and Biomedical Engineering, June 7-12, 2015, Toronto, Canada*,
IFMBE Proceedings 51, DOI: 10.1007/978-3-319-19387-8_281

A. Data Acquisition

Knee kinematic data was utilized for CMAC training. This data was captured through motion capture techniques, using 12 cameras Qualisys Oqus MRI, passive markers and software package Qualisys QTM 3.2. Data were converted to the appropriate format supported by Octave 3.8.1 language (MATLAB format). Data were captured from a subject in Human Performance Laboratory at Faculty UnB Ceilândia. A healthy male subject was selected. He repeated a walk of approximately 5 seconds for 5 times. This process generated spatial variables, regarding the position of the markers. The markers were distributed along 34 positions at inferior limbs. As only knee flexion and extension were necessary for this work, only the markers of tibias, knees and trochanters were utilized. It was possible with these data, to calculate knees angles, angular velocities and angular accelerations. The initial and final data of each sample had to be eliminated, because they constitute the comfortable gait cycle beginning and end.

The data acquisition was approved by the UnB Health Faculty Ethics Committee, protocol N11911/12.

B. Kinematic Calculations

Kinematic calculations were made using the Octave programming language which is compatible with MATLAB. This choice was made, because Octave is open source and because the QTM export data to MATLAB format.

Angles were obtained using the equation (1) from [10].

$$\theta = acos\left(\frac{u.v}{\| u \| \; \| v \|}\right) \tag{1}$$

The u variable is equivalent to the vector at the trochanter point; v is the vector at the tibia point. The knee point must be the origin of u *and* v, so these must be translated to the new origin [11].

The angular velocities were obtained using adjacent points of collected data following equation (2):

$$\omega = \frac{\theta 2 - \theta 1}{t} \tag{2}$$

The variable t is the time between adjacent calculated angles $\theta 1$ and $\theta 2$. For this paper t is equal $1/315$ seconds.

The source code to extract markers position and to generate knee angles, angles velocities and angles accelerations, is available in http://github.com/rob-nn/gait_data_loader.

C. CMAC setting

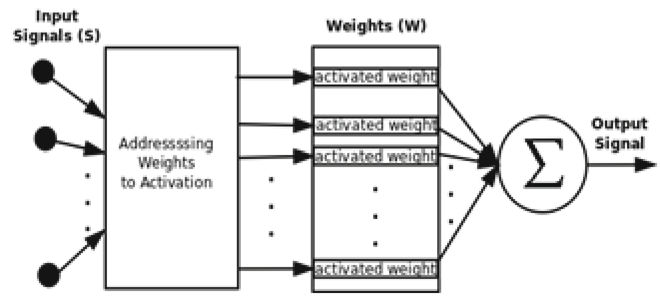

Fig. 2: Simulation Process

The CMAC model proposed by [3] is depicted in Figure 2. This type of ANN receives a signal vector *S,* and then uses this vector to calculate addresses. These addresses correspond to positions in weight vector *W*. Only activated weights will be summed to compose the CMAC output signal.

The input signals for a CMAC must have their enter space known. For example, for a vector *S,* with signals *s1*, *s2* and *s3*, the signal *s1* could have a value between 0 and 1, the signal *s2* a value between -500 and 100 and *s3* a value between -0.1 and 0.1. Furthermore, each signal must be converted to an exact quantity of discrete values. For example, a signal *sx*, which is between 0 and 2, has five discrete values, such that their possible values are 0, 0.5, 1, 1.5 and 2. If for example, it has the value 1.6, then *sx* takes the value 2, because there is not the value 1.6 in the list of discrete values. The quantity of discrete values by signal defines the signal resolution. More discrete values more resolution. Less discrete values less resolution. See Figure 3. The calculation of addresses has been implemented using the same algorithm proposed by [3].

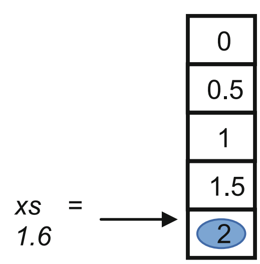

Fig. 3: Input value discretization for a signal *sx*

An essential parameter in CMAC is the number of activations. As the name says, it defines the number of weights which will compose the output. How much higher this parameter, higher will be the generalization encompassed by CMAC.

ANNs as machine learning algorithms must be trained. Learning can be supervised or not supervised. CMAC's case is supervised learning. During this phase input signals and the desired output signal must be given for CMAC's learning algorithm. The last parameter to inform is the iterations number of the CMAC. This parameter is the number of times the training algorithm will update the weights after calculating the output for all input signals previously collected.

D. Simulation and implementation details

For CMAC's implementation and knee's simulation was adopted the Python 2.7 programming language, the software packages NumPy and Matplotlib were used too. The operating system is Mac OS X Yosemite version 10.10.1. The hardware is a Mac Pro (Mid 2010), processor 2.8 GHz Quad-Core Intel Xeon, RAM 3GB. In addition, the source code to the created simulations is available in http://github.com/rob-nn/motus.

As input signals for the simulation, velocities vectors in a 3D plane (x, y, z) of the left knee were chosen. As output, was chosen to predict the signal corresponding to the right knee angular velocities. These choices were made based on work [12].

Before running the simulation, the RNA CMAC must be trained. For accomplish this task only 50% of the data collected in a single walk were used. The other 50% were used to make the prediction of the output signal and compare them with the real signal captured.

III. RESULTSS

Figure 4 shows the simulation performed. For this simulation was informed: 50 iterations, 30 activations and 200 discrete values per input signal. The input signals are left knee angular velocities in a 3D plane (x, y, z). The data are 50% of a 5 seconds walk. Parts of the first and last moments were discarded. The output is the angular velocities of the right knee.

In addition to the generated simulation, an open source project, called Motus was created. See Figure 5 This project aims to create a tool for basic gait analysis and signal simulation of human lower limbs. Figure 5 shows the version 0.1 of this tool, which is available on GitHub site at http://github.com/rob-nn/motus.py address. The available version is capable, a while, to work with the following signals:

1. Knee angles (flexion, extension);
2. Knee angular velocities (flexion, extension);
3. Knee angular accelerations (flexion, extension);
4. Knee velocities in a 3D plane.

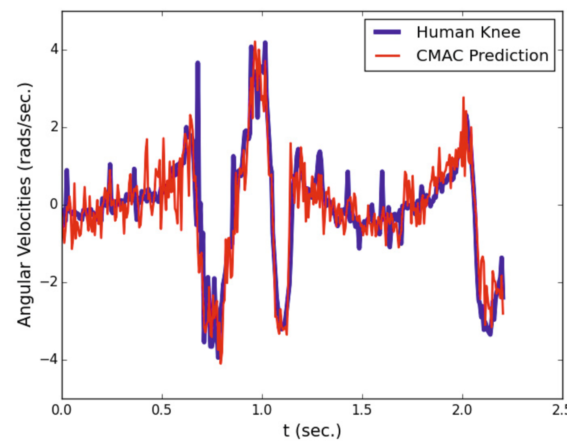

Fig. 4: Human knee angular velocity prediction

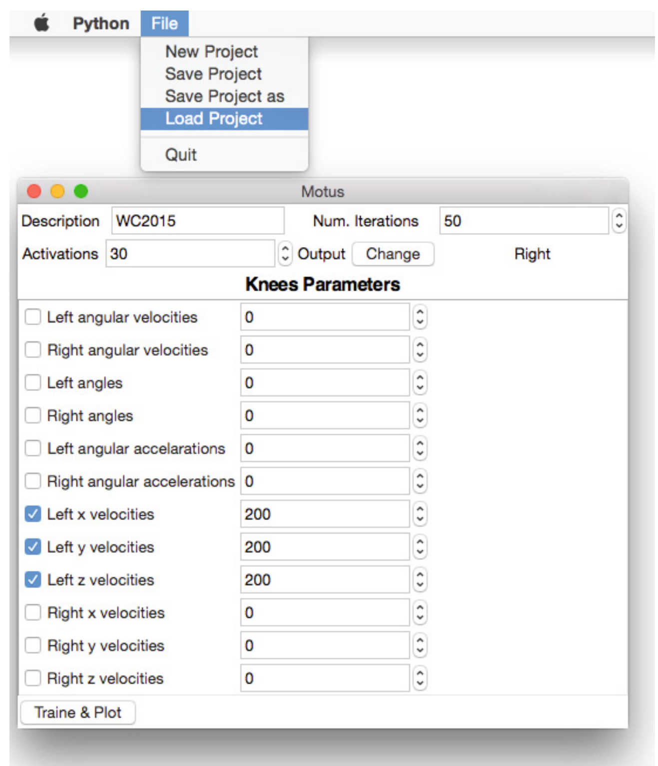

Fig. 5: Motus Version 0.1

Motus development is powered by Healthy Informatics Lab (*Laboratório de Informática em Saúde* – LIS).

IV. Discussion

The trained CMAC can now be used for testing in a real prototype for a transfemoral active prosthesis. The advantage is that it is easily implemented in embedded systems. It requires only simple tables and calculations. Maybe more data is required, but a tool for training the CMAC is already done.

Others joints (hip, anckles) and order degree of freedoms can be simulated too. It is necessary however, to collect data from the other joints and calculate new angles, velocities and accelerations (using the same implemented methods), which isn't done yet. Motus is a working in progress.

V. Conclusions

The development of a transfemoral active prosthesis requires building control systems. These systems should be implemented as embedded systems, which typically have limited computing power. ANNs can be used for building control systems, but ANNs require a process called training. This usually requires a high computational power and is best accomplished in desktop systems or computers clusters.

This paper has shown the simulation approximation of a signal relating to a knee angular velocity, based on contralateral knee signals. All this simulation was performed using a CMAC, a type of ANN, which showed very good results graphically.

Besides the simulation, all necessary code to build and train the CMAC and also to simulate signals is available as open source. This project is on GitHub site and is accessible to anyone. The same is under continuous development.

VI. Acknowledgments

The authors thank the resources provided by the LIS and the Movement Lab of UNB Ceilândia, also CAPES for the financial support given to the first author.

VII. Conflict of Interests

The authors declare that they have no conflict of interest.

VIII. References

1. Pickle N T, Wilken J M et al. (2014) Whole-body angular momentum during stair walking using passive and powered lower-limb prostheses. J Biomechanics 47:3380–3389
2. Bishop C N (2006) Pattern recognition and machine learning. Springer, Singapore
3. Albus J S (1975) A new approach to manipulator control: The cerebellar model articulation controller (CMAC). J of Dynamic Systems, Measurement, and Control 220-227
4. Albus J S (1975) A theory of cerebellar function. Mathematical Biosciences J 10:25-61
5. Haykin S (2008) Neural networks and learning machines. Pearson Prentice Hall, New York
6. Smith R L (1998) Intelligent motion control with an artificial cerebellum. University of Auckland, Auckland
7. Lyn Y (1992) A CMAC neural-network-based algorithm for kinematic control of a walking machine. Engn. Applic. Artif. Intel. J 6:539-551
8. Sabourin C, Bruneau O (2005) Robustness of dynamic walk of a biped robot subjected to disturbing external forces by using CMAC neural networks. Robotics and Autonomous Systems J 51:81-99
9. Sabourin C, Bruneau et al. (2006) Control strategy for the robust dynamic walk of a biped robot. The International J of Robotics Research 25:843-860
10. Edwards L (2010) Calculus. Books Cole, Belmont
11. Poole D (2011) Linear algebra a modern introduction. Brooks Cole, Boston
12. Vallery H., Burgkart R et al. (2011) Complementary limb motion estimation for the control of active knee prostheses

Author: Roberto Aguiar Lima
Institute: UnB Gama / Mestrado em engenharia biomédica
Street:
City: Brasília
Country: Brazil
Email: aguiarlima.roberto@gmail.com