Three Machine Learning Techniques for Automatic Determination of Rules to Control Locomotion

Slavica Jonić, *Student Member, IEEE*, Tamara Janković, Vladimir Gajić, and Dejan Popović,* *Member, IEEE*

***Abstract*—Automatic prediction of gait events (e.g., heel contact, flat foot, initiation of the swing, etc.) and corresponding profiles of the activations of muscles is important for real-time control of locomotion. This paper presents three supervised machine learning (ML) techniques for prediction of the activation patterns of muscles and sensory data, based on the history of sensory data, for walking assisted by a functional electrical stimulation (FES). Those ML's are: 1) a multilayer perceptron with Levenberg–Marquardt modification of backpropagation learning algorithm; 2) an adaptive-network-based fuzzy inference system (ANFIS); and 3) a combination of an entropy minimization type of inductive learning (IL) technique and a radial basis function (RBF) type of artificial neural network with orthogonal least squares learning algorithm. Here we show the prediction of the activation of the knee flexor muscles and the knee joint angle for seven consecutive strides based on the history of the knee joint angle and the ground reaction forces. The data used for training and testing of ML's was obtained from a simulation of walking assisted with an FES system [39]. The ability of generating rules for an FES controller was selected as the most important criterion when comparing the ML's. Other criteria such as generalization of results, computational complexity, and learning rate were also considered. The minimal number of rules and the most explicit and comprehensible rules were obtained by ANFIS. The best generalization was obtained by the IL and RBF network.**

***Index Terms*—Adaptive-network-based fuzzy inference system (ANFIS), artificial neural networks (ANN's), functional electrical stimulation (FES), inductive learning (IL), multilayer perceptron, radial basis function (RBF) ANN, walking.**

I. Introduction

CONTEMPORARY functional electrical stimulation (FES) systems use volitionally controlled four or six channels of stimulation [25], or a preprogrammed sequence of stimulation patterns applied to as many as 48 muscles [19]. Tuning the stimulation patterns in the later case is handcrafted for each user [19]. Several more sophisticated control strategies were presented in the literature, involving open-loop [8], [19], [34] and closed-loop control [16], [17]. When considering the closed-loop control it is important to speak of dynamic characteristics of muscles. They operate as low-pass filters with respect to neural inputs. Dynamics of muscles can be characterized by rise time of approximately 50–100 ms, depending on the muscle type. Muscle activity is delayed after the neural signal for about 30–50 ms. Those features of a muscular system impose that a command signal precedes the required muscle activity when a real-time control is to be implemented, and are the most important arguments against implementing closed-loop control.

An alternative for control is using nonanalytical control to clone the biological control [36], [38]. The basic mechanism for implementation of such algorithms is a rule-based system [37]. Rule-based controllers for locomotion were originally hand crafted [1], [2], [4], [12], [44]–[46], and lately automatically tuned [11], [15], [18], [20]–[24], [31]–[33] but none was sufficiently practical to be widely used. A rule-based control is an implementation of "if–then" relations. "If" part of a rule describes the sensory states, while a "then" part of a rule defines the corresponding motor, that is muscle activity. In other words, when a characteristic sensory pattern is recognized, a muscle activity must occur.

In order to design a rule-based system a knowledge base has to be generated. It was hypothesized that machine learning (ML) can help in acquiring the needed knowledge. Learning in general can be described as capturing and memorizing of a connectivism between facts. ML is a computerized capturing and memorizing process. ML's were applied successfully to automate the walking [11], [18], [20], [33], [37], but the walking pattern was not improved.

We propose to use simulation results of a fully customized biomechanical model of a human with disability for inputs and outputs required for ML, therefore to clone a desired pattern of walking. The simulation described in [14] and [39] generates a set of inputs and outputs which is suitable for the said cloning.

ML's used in this study are: 1) a multilayer perceptron (MLP) with the Levenberg–Marquardt improvement of back-propagation (BP) algorithm [41]; 2) an adaptive-network-based fuzzy inference system (ANFIS) [13]; and 3) a combination of an entropy minimization-type of inductive learning (IL) technique [11], [32], [33], [35], [47] and a radial basis function (RBF)-type of artificial neural network (ANN) [6], [32] with orthogonal least squares (OLS) learning algorithm [6].

A comparison of IL with adaptive logic networks (ALN) using restriction rules [22], [24] shows that ALN's have some advantages over IL. The application of ALN's is intentionally omitted since it was published earlier in great details [20]–[24]. Heller *et al.* [11] compared an IL method based on an algorithm called "hierarchical mutual information classifier" [42] with a MLP with BP algorithm in reconstructing muscle activation from kinematic data during normal walking. The conclusion was that both techniques show comparable performance, although each technique has some advantages over the other one. A comparison of IL method based on

Manuscript received November 12, 1997; revised July 22, 1998. This work was supported in part by a grant from the Ministry of Science and Technology of Serbia, Belgrade, Yugoslavia. *Asterisk indicates corresponding author.*

S. Jonić, T. Janković, and V. Gajić are with the Faculty of Electrical Engineering, University of Belgrade, 11000 Belgrade, Yugoslavia.

*D. Popović is with the Faculty of Electrical Engineering, University of Belgrade, 11000 Belgrade, Yugoslavia (e-mail: dbp@el.etf.bg.ac.yu).

Publisher Item Identifier S 0018-9294(99)01846-7.

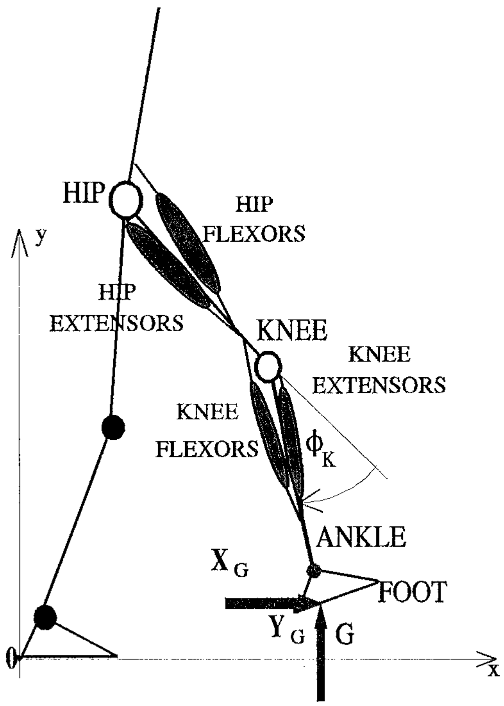


Fig. 1. Model of a human body showing the knee joint angle, ground reaction forces, and the equivalent hip and knee flexors and extensors. The model is used for calculating data used as inputs and outputs for pattern matching.

minimization of entropy and the RBF network in predicting muscle activation and sensory data from the history of sensory data for a human with spinal cord injury (SCI) [15] shows that the best generalization comes from combining both. The benefits of merging fuzzy logic and an ANN were explored extensively in the literature [13], [27].

The aim of the study was to analyze which technique is the best when considering the number of rules, the simplicity and comprehensibility of rules, and the generalization of the results.

II. Task Formulation

The muscle activation patterns and sensory data used for pattern matching were obtained from biomechanical analysis of a human walking with FES [39]. The leg is modeled as a planar, two-segment linkage of rigid bodies (Fig. 1). It is assumed that the leg was driven by two pairs of equivalent monoarticular flexor and extensor muscles acting around the hip and knee joints (Fig. 1). The term equivalent muscle relates to the simplification that all of the muscles contributing to the activity at a joint are represented with a single muscle. The inputs for simulation are: the ground reaction forces, hip acceleration, the knee and hip joint angles, and the angle between the trunk and the horizontal. This data was recorded during a self-paced, level walking of an able-bodied human. The outputs are the muscle activation patterns at the knee and hip joints, as well as the corresponding knee and hip joint angles. Dynamic programming was used to determine muscle activation patterns by optimizing the tracking of the trajectories of the joints and minimizing the overlap of agonist and antagonist activities as showed in details in Popović *et al.* [39].

The set of inputs and outputs for the supervised ML was selected from the collection of results obtained by simulation. Fig. 2 shows inputs borrowed with permission from the authors of [39] and used in this pattern matching study. The top panel is the angle at the knee joint, the middle panel shows the ground reaction forces, and the bottom panel shows the activation pattern of the equivalent knee flexor muscle. The maximum activation was assumed to be one, while the resting muscle is described as zero. The sampling rate for all data was 100 per s.

The design of rules has the following two elements: 1) the pattern matching, that is the prediction of activation patterns of the equivalent knee flexor muscle; and 2) the pattern matching, that is the prediction of the knee joint angle. The estimation of muscle activation patterns with respect to the sensory signals provides the timing for the onset and offset of the change of activity, but as said this command must precede the actual activity. Thus, the pattern matching of a sensory data is required to predict a sensory pattern which will follow after 100 ms. The knee joint angle and the ground reaction forces preceded for 50 ms were used as inputs when predicting the muscle activation pattern. The knee joint angles preceding for 50, 100, and 150 ms were used as inputs when predicting the sensory pattern. The training set for this study included intentionally only the four consecutive strides (Fig. 2, HC3 to HC7). The testing set included the sequence of seven consecutive strides (Fig. 2). The first two strides and the last stride were not used for the training. The learning algorithms were implemented using MatLab 4.2c.1 on a PC platform (Pentium 133-MHz, 16-Mb RAM).

The evaluation of results for all three techniques was done by comparing the following:

1) the errors in the timing of muscle activations t_{er}, that is periods when a muscle is active given in the form

$$t_{er}(i) = h(a(i) - a^*) - h(a_{\text{des}}(i) - a^*)$$

where $a(i)$ and $a_{\text{des}}(i)$ are the ith sample of estimated and simulated (desired) muscle activations, respectively, a^* is a threshold defined for the desired signal allowing to say that a muscle is active if the activation is bigger than a^*

$$h(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

2) cross correlation between simulated (desired) and estimated signals (muscle activation and joint angle) in the form

$$\text{cor} = \frac{\sum_{i=1}^{r} s(i) s_{\text{des}}(i)}{\sqrt{\sum_{i=1}^{r} s^2(i)} \sqrt{\sum_{i=1}^{r} s_{\text{des}}^2(i)}}$$

where $s(i)$ and $s_{\text{des}}(i)$ are the ith sample of the estimated and the desired signals, respectively, and r is the number of signal samples;

3) elapsed time for the training and testing.

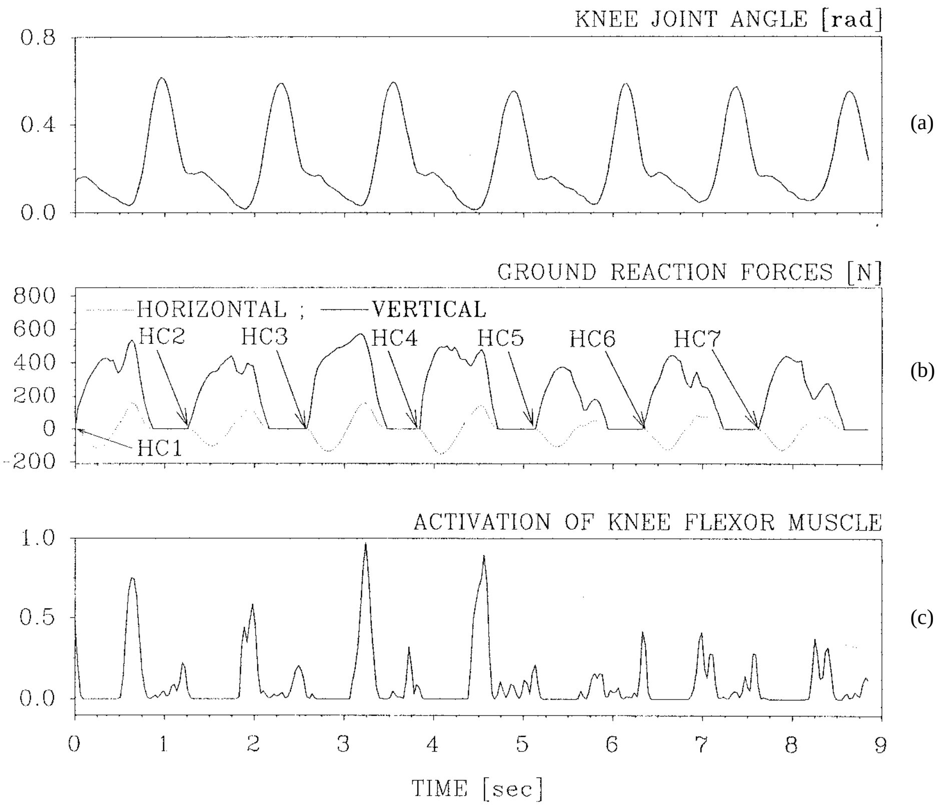


Fig. 2. Data obtained from the simulation and used as input and output sets for pattern matching. (a) Shows the angle at the knee joint, (b) shows the ground reaction forces, and (c) shows the activation patterns of the equivalent knee flexor muscle. The activation is normalized at one. The arrows in middle panel show heel contact. Time interval between two consecutive arrows is one gait stride.

III. Use of MLP

A. MLP and BP Learning Algorithm

Investigations of MLP's have been intensified since the formulation of the BP learning algorithm [41]. An MLP is a feed-forward network, typically consisting of several layers of nonlinear processing nodes called hidden layers with a linear output layer. Processing nodes take as input only the outputs of the previous layer, which are combined as a weighted sum and then passed through a nonlinear processing function known as the "activation function." This activation function is typically sigmoidal in shape. An MLP with three hidden layers can form arbitrarily complex decision regions and can separate classes that are meshed together. It can form regions as complex as those formed using mixture distributions and nearest neighbor classifiers [28]. The MLP used here has a single linear output node and a single hidden layer. The network function can be described by

$$f(\omega) = \xi_0 + \sum_{q=1}^{N} \xi_q \psi\left(d_q^T v + \theta_q\right)$$

where v is the input vector, N is the number of nodes in the hidden layer, and $\psi(\cdot)$ is activation function selected here as sigmoidal. The input vector is connected to the qth hidden node by weight d_q, and θ_q is the scalar bias for this node. The output from qth hidden node is connected to the output node by weight ξ_q, and ξ_0 is the scalar bias for the output node. The weight vector contains all the weights and biases called parameters of the network.

The BP learning algorithm is a generalization of a gradient descent algorithm. It uses a gradient search technique to minimize a cost function equal to the sum of the squares differences between desired and estimated net outputs. Derivatives of error (called delta vectors) are calculated for the network's output layer, and then backpropagated through the network until delta vectors are available for each hidden layer of the network. The BP algorithm may lead to a local, rather than a global error minimum. If the local minimum found is not satisfactory, use of several different sets of initial conditions or a network with more neurons can be tried.

B. Improvements of BP Learning Algorithm

A simple BP algorithm is very slow because it must use small learning rates for stable learning. There are ways to improve the speed and general performance of a BP algorithm. It can be improved in two different ways: by heuristics and by using more powerful methods of optimization. Speed and reliability of BP can be increased by techniques called momentum and adaptive learning rates. The momentum technique helps the network to get out, if stacked in shallow minimum. By the use of adaptive learning rates it is possible to decrease the learning time. In this paper we used a Levenberg–Marquardt modification of BP algorithm. By using Levenberg–Marquardt optimization the training time can be shortened. Its update

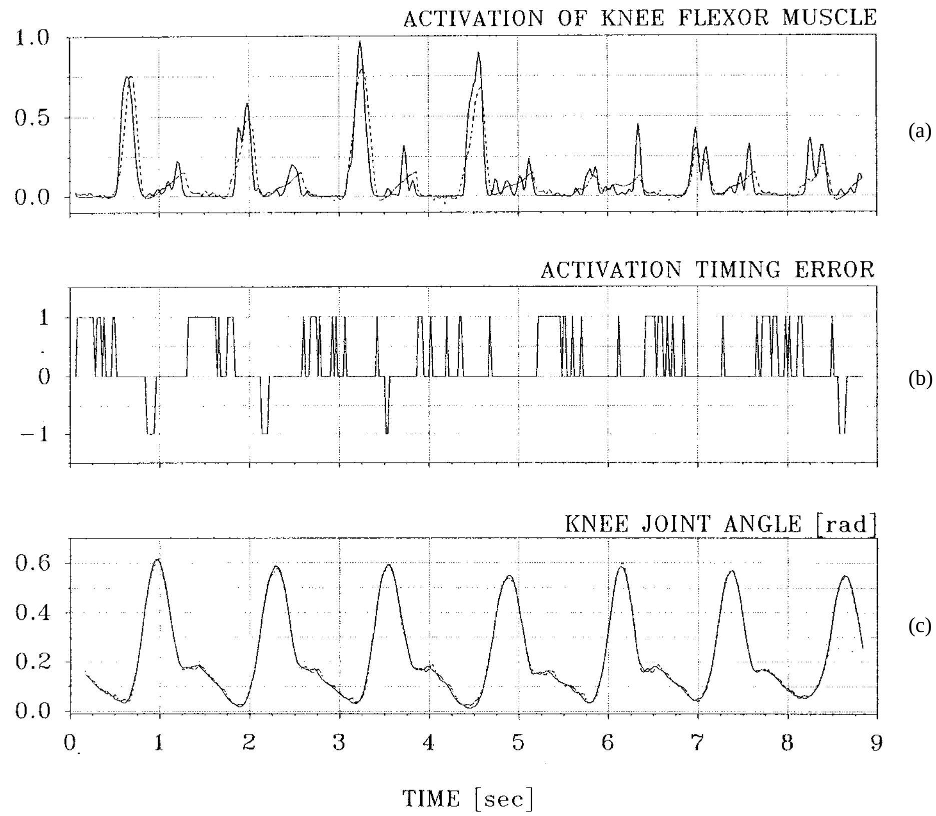


Fig. 3. Results of applying the MLP with Levenberg–Marquardt modification of BP learning algorithm. (a) Shows the simulated activation patterns of the equivalent knee flexor muscle shown at Fig. 2 (full line) and the predicted activation patterns (dashed line). The inputs used for this pattern matching were the simulated knee joint angle and the ground reaction forces shifted back for 50 ms. (b) Shows the errors in the muscle activation timing. (c) Shows the simulated knee joint angle (full line) and the predicted angle (dashed line). The three inputs for this matching were the simulated knee joint angle shifted back for 50, 100, and 150 ms.

rule is

$$\Delta\omega = \left(J^T J + \mu I\right)^{-1} J^T e$$

where $\Delta\omega$ is column matrix whose number of rows matches the number of network parameters, J is the Jacobian matrix of derivatives of network function error, that is the difference between desired and estimated net outputs, for each training pattern to each network parameters. The number of rows in J matches the number of training patterns, and the number of columns matches the number of network parameters. e is a column matrix of errors for each training pattern. The number of rows in e matches the number of training patterns. I is the identity matrix whose number of rows as well as columns matches the number of network parameters. μ is a scalar. If μ is very large, the above expression approximates gradient descent, while if μ is small this expression becomes the Gauss-Newton method. The coefficient μ is changed in such a way as to join good features of both algorithms: gradient descent algorithm (it does not require that initial values of parameters are well chosen), and Gauss–Newton algorithm (it has quadratic convergence near an error minimum). As long as the error gets smaller, μ is made bigger, but, once the error starts increasing, μ is made smaller. The Levenberg–Marquardt is much faster than the gradient descent algorithm, on which the standard BP algorithm is based. However, it requires more memory than the gradient descent algorithm.

C. Results of Using Multilayer Perceptron with Levenberg–Marquardt Modification of Backpropagation Learning Algorithm

Fig. 3 shows results of application of the MLP with Levenberg–Marquardt modification of BP learning algorithm. Fig. 3(a) shows the predicted (dashed line) and simulated (full line) activation patterns of the equivalent knee flexor muscle. The inputs for this pattern matching were the knee joint angle and the ground reaction forces from the preceding 50 ms, while the output was the muscle activation pattern. Fig. 3(b) shows the errors in the muscle activation timing. Fig. 3(c) shows the simulated (full line) and the predicted (dashed line) knee joint angle. The inputs for this pattern matching were the knee joint angle preceding for 50, 100, and 150 ms, while the output was the knee joint angle. Two different networks were used for matching of muscle activation and joint angle.

The number of nodes in a hidden layer of the network estimating muscle activity was chosen to be six, and the number of nodes in a hidden layer of the network estimating the joint angle was chosen to be four. The multiplier factor for increasing μ was selected to be 1.2, and the multiplier factor for decreasing of μ was selected to be 1/1.2 in both case of pattern matching. The number of training epochs in both case was chosen to be 25. All of the numbers given above were selected after numerous and tedious trials to get the best pattern matching results. The cross correlation between the desired and estimated value of muscle activity was 0.94, and cross correlation between the desired and estimated value of joint angle was 0.999. The muscle activation pattern matching lasted approximately 22 s. The joint angle pattern matching lasted approximately 18 s.

IV. Use of Adaptive-Network-Based Fuzzy Inference System

A. Fuzzy Sets and Fuzzy Models

Fuzzy sets are a generalization of conventional set theory. They were introduced by Zadeh [50] as a mathematical way to represent vagueness in everyday life. A formal definition of fuzzy sets that has been presented by many researchers is following: a fuzzy set A is a subset of the universe of discourse X that admits partial membership. The fuzzy set A is defined as the ordered pair $A = \{x, m_A(x)\}$, where $x \in X$ and $0 \leq m_A(x) \leq 1$. The membership function $m_A(x)$ describes the degree to which the object x belongs to the set A, where $m_A(x) = 0$ represents no membership, and $m_A(x) = 1$ represents full membership.

One of the biggest differences between conventional (crisp) and fuzzy sets is that every crisp set always has a unique membership function, whereas every fuzzy set has an infinite number of membership functions that may represent it. This is at once both a weakness and a strength; uniqueness is sacrificed, but this gives a concomitant gain in terms of flexibility, enabling fuzzy models to be "adjusted" for maximum utility in a given situation.

The typical steps of a "fuzzy reasoning" consist of: 1) Fuzzification, that is comparing the input variables with the membership functions of the premise (IF) parts in order to obtain the membership values between zero and one; 2) Weighing, that is applying specific fuzzy logic operators (e.g., AND operator, OR operator, etc.) on the premise parts membership values to get a single number between zero and one; 3) Generation, that is creating the consequent (THEN) part of the rule; and 4) Defuzzification, that is aggregating the consequent to produce the output.

There are several kinds of fuzzy rules used to construct fuzzy models. These fuzzy rules can be classified into the following three types according to their consequent form [26]:

1) Fuzzy rules with a crisply defined constant in the consequent

$$R_i\text{: IF } x_1 \text{ is } A_{i1} \text{ and} \cdots \text{and } x_m \text{ is } A_{im}, \text{ THEN } y \text{ is } c_i$$

.

2) Fuzzy rules with a linear combination of the systems input variables in the consequent

$$R_i\text{: IF } x_1 \text{ is } A_{i1} \text{ and} \cdots \text{and } x_m \text{ is } A_{im},$$

$$\text{THEN } y \text{ is } g_i(x_1, \cdots, x_m) = b_0 + b_1 x_1 + \cdots + b_m x_m$$

.

3) Fuzzy rules with fuzzy set in the consequent

$$R_i\text{: IF } x_1 \text{ is } A_{i1} \text{ and} \cdots \text{and } x_m \text{ is } A_{im}, \text{ THEN } y \text{ is } B_i$$

where R_i is the ith rule of the fuzzy system, x_j $(j = 1, 2, \cdots m)$ are the inputs to the fuzzy system, and y is the output from the fuzzy system. The linguistic terms A_{ij} and B_i are fuzzy sets, c_i and b_j denote crisp constants.

The so-called zero-order Sugeno, or Takagi–Sugeno–Kang fuzzy model [43] has rules of the first type, whereas the first-order Sugeno fuzzy model has rules of the second type. The easiest way to visualize the first-order Sugeno fuzzy model is to think of each rule as defining the location of a "moving singleton" (a single spike from the consequent) depending on what the input is. Sugeno models are similar to the Mamdani model [29] which has rules of the third type, and which is more intuitive, but computationally less efficient. Fuzzification and weighing, are exactly the same, but generation and defuzzification are different [26].

For the type of fuzzy rules used in Mamdani model various methods are available for defuzzification: the centroid of area, bisector of area, middle of maximum, largest of maximum etc. [10], but all of these methods are based on the calculation of the two-dimensional-shape surface, that is on the integration. The Sugeno-style enhances the efficiency of the defuzzification process because it greatly simplifies the computation; i.e., it has to find just the weighted average of a few data points. The implication method (generation) is simply multiplication, and the aggregation operator just includes all of the singletons. For the first-order Sugeno fuzzy model, that is used in this paper, defuzzified value y_0 is

$$y_0 = \frac{\sum_i g_i(a_1, a_2, \cdots, a_m) \prod_j \mu_{A_{ij}}(a_j)}{\sum_i \prod_j \mu_{A_{ij}}(a_j)}$$

where $\mu_{A_{ij}}(a_j)$ is the membership degree of input a_j $(j = 1, 2, \cdots m)$ to the antecedent linguistic term A_{ij} for the ith rule of the fuzzy system.

B. Subtractive Clustering Method and Adaptive-Network-Based Fuzzy Inference System

Membership functions are subjective and context-dependent, so there is no general method to determine them. Currently, when fuzzy set theory is applied in control systems, the system designers are given enough freedom to choose membership functions and operators, usually in a trial and error way. After a hand-tuning process, the system can function effectively. However, the same methodology is hardly applicable when the system is a general purpose one, or when the context changes dynamically. This suggests an explanation why the most successful applications of fuzzy logic happen in control systems, rather than in natural language processing, knowledge base management, general purpose reasoning.

The method used for identification of a first-order Sugeno-type fuzzy interference system (FIS) is a two step procedure: the first step used subtractive clustering method [7] for initial identification of FIS, the second step used an adaptive-network with BP and least squares (LS) algorithms for tuning of initial identified FIS [13]. The approach used here to extract the initial rules is based on replacing identification of membership

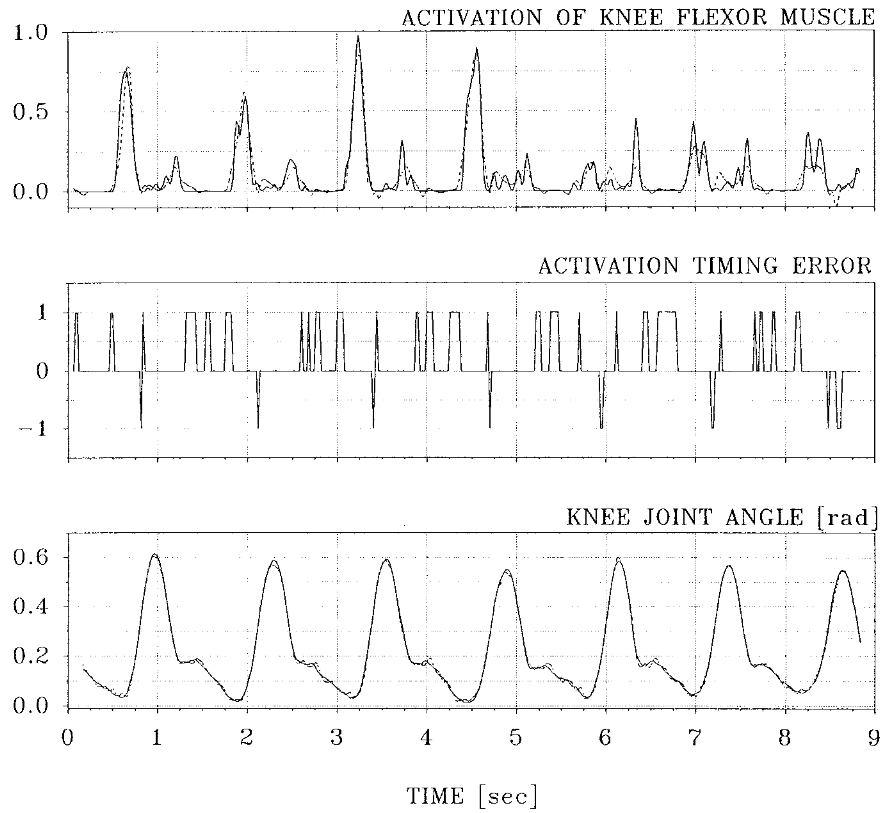


Fig. 4. Results of applying the subtractive clustering method for initial identification of a first- order Sugeno-type FIS and the adaptive-network with BP and LS learning algorithms for tuning of initial identified FIS. For details see the caption of Fig. 3.

functions of input variables with identification of the centers of cluster-like regions. Fuzzy c-means the technique introduced by Bezdek [5] as an improvement on earlier data clustering methods and requires that the number of clusters is known. If there is not a clear idea how many clusters there should be for a given set of data, subtractive clustering method [7] can be used for estimating the number of clusters and the cluster centers in a set of data. This method is used here and it is an extension of the Mountain clustering method proposed by Yager [48]. It assumes each data point is a potential cluster center and calculates a measure of the potential for each data point based on the density of surrounding data points. The algorithm selects the data point with the highest potential as the first cluster center and then destroys the potential of data points near the first cluster center. The algorithm then selects the data point with the highest remaining potential as the next cluster center and destroys the potential of data points near this new cluster center. This process of acquiring a new cluster center and destroying the potential of surrounding data points repeats until the potential of all data points falls bellow a threshold. The range of influence of a cluster center in each of the data dimensions is called cluster radius. A small cluster radius will lead to finding many small clusters in the data (resulting in many rules) and vice versa. The cluster information obtained by this method are used for determining the initial number of rules and antecedent membership functions, that is for identifying the FIS. Then, the linear least-squares estimation is using to determine the consequent for each rule. The result is the initial fuzzy rule base. However, for different initial values, this method may give different results, because the identification algorithm depends on an optimization procedure.

An adaptive-network with a single output node and a single hidden layer was used here for tuning the initial identified FIS. The LS method and the BP gradient descent method were used for tuning linear and nonlinear parameters of first-order Sugeno-type FIS, respectively [13]. A network structure facilitates the computation of the gradient for parameters in a FIS. The adaptive-network improves the rules determined from initial identification of FIS. The result is an FIS which corresponds to the minimum training error.

C. Results of Using Subtractive Clustering Method and ANFIS

Fig. 4 shows results of application of the subtractive clustering method for initial identification of a first-order Sugeno-type FIS and the adaptive-network with BP and LS learning algorithms for tuning of initial identified FIS. The top panel shows the predicted (dashed line) and simulated (full line) activation patterns of the equivalent knee flexor muscle. The inputs for this pattern matching were the knee joint angle and the ground reaction forces from the preceding 50 ms, while the output was the muscle activation pattern. The middle panel shows the errors in the muscle activation timing. The bottom panel shows the simulated (full line) and the predicted (dashed line) knee joint angle. The inputs for this pattern matching were the knee joint angle preceding for 50, 100, and 150 ms, while the output was the knee joint angle. Two different ANFIS's were used for matching of muscle activation and joint angle.

The cluster radius of each input was chosen to be 0.5. 46 nodes were obtained in a hidden layer of the adaptive-network for FIS estimating muscle activity, and five fuzzy rules were obtained. 30 nodes were obtained in a hidden layer of the adaptive-network for FIS estimating the joint angle, and only three fuzzy rules. The number of training epochs in both case was chosen to be ten. The cluster radius and the number of epochs were selected based on the experience gained through numerous trials while trying to get the best matching. The cross correlation between the desired and estimated value of muscle activity was 0.95, and cross correlation between the desired and estimated value of joint angle was 0.999. The muscle activation pattern matching lasted approximately 53 s. The joint angle pattern matching lasted approximately 46 s.

V. Use of Entropy Minimization Type of IL and RBF Type of ANN

A. Entropy Minimization Type of Inductive Learning Technique

The IL is a symbolic technique which uses supervised learning and generates a set of "if–then–else" decision rules. A method [32], [33] used here is based on an algorithm called "hierarchical mutual information classifier" [42]. A program Empiric described in [11] implements this algorithm. This algorithm produces a decision tree by maximizing the average mutual information at each partitioning step. It uses Shannon's entropy as a measure of information.

Mutual information is a measure of the amount of information that one random variable contains about another random variable. It is a reduction of the uncertainty of one random variable due to the knowledge of the other. Consider two random variables X and Y with a joint probability density function $p(x,y)$ and marginal probability density functions $p(x)$ and $p(y)$. Mutual information $I(X,Y)$ is the relative entropy between the joint distribution and the product distribution $p(x)p(y)$

$$I(X,Y) = \sum_x \sum_y p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}.$$

The mutual information can also be written as

$$I(X,Y) = S(X) - S(X/Y)$$

where $S(X)$ is Shannon's entropy

$$S(X) = -\sum_x p(x) \log_2 p(x)$$

and $S(X/Y)$ is conditional entropy

$$S(X/Y) = -\sum_x \sum_y p(x,y) \log_2 p(x/y)$$

$p(x/y)$ is the conditional probability density function.

An effective method of integrating results of a mutual information algorithm into a production rule formalism, following the original work of Pitas [35] and Watanabe [47] is shown in [32] and [33]. While generating the decision tree, the algorithm performs a hierarchical partitioning of the domain multidimensional space. Each new node of the decision tree contains a rule based on a threshold of one of the input signals. Each new rule further subdivides the example set. The training is finished when each terminal node contains members of only one class. An excellent feature of this algorithm is that it determines threshold automatically based on the minimum entropy [32], [33], [35]. This minimum entropy method is equivalent to determination of the maximum probability of recognizing a desired event (output) based on the information from an input.

B. RBF Type of ANN

RBF network [6], [32] is a feed-forward network. The RBF used here has a single output node and a single hidden layer which contains as many neurons as are required to fit the function within the specifications of error goal. The transformation from the input space to the hidden-unit space is nonlinear, whereas the transformation from the hidden-unit space to the output space is linear. Its function is given by

$$f(\omega) = \xi_0 + \sum_{q=1}^{N} \xi_q \zeta(||v - c_q||)$$

where v is the input vector, N is the number of the hidden nodes, and $\zeta(\cdot)$ is the activation function (known as the radial basis function for a RBF network). Theoretical investigations and practical results show that the type of nonlinearity $\zeta(\cdot)$ is not crucial to the performance of RBF network [40], and it is usually taken to be bell-shaped function as in this case. The $||\cdot||$ denotes a norm that is usually taken to be Euclidean. The c_q are known as vectors of RBF centers, ξ_q and ξ_0 are the qth weight and the bias for output linear node.

A common learning algorithm for RBF networks is based on first choosing randomly some data points as RBF centers and then using singular value decomposition to solve for the weights of the network. An arbitrary selection of centers may not satisfy the requirement that centers should suitably sample the input domain. Furthermore, in order to achieve a given performance, an unnecessarily large RBF network may be required. Since performance of an RBF network critically depends upon the chosen centers, we used an alternative learning procedure based on the OLS learning algorithm [6]. By providing a set of inputs and corresponding outputs, the values of weights ξ_q, bias ξ_0, and RBF centers (parameters for RBF network) can be determined using the OLS algorithm in one pass of the learning data so that a network of an adequate size can be constructed.

When an input vector v is presented to such a network, each neuron in the hidden layer will output a value according to how close the input vector is to the centers vector of each neuron. The result is that neurons with centers vector very different from the input vector will have outputs near zero. These small outputs will have a negligible effect on the linear output neurons. In contrast, any neuron whose centers vector is very close to the input vector will output a value near one. If a neuron has an output of one, its output weights in the second layer pass their values to the neuron in the second layer. The width of an area in the input space to which each radial basis neuron responds can be set by defining a spread

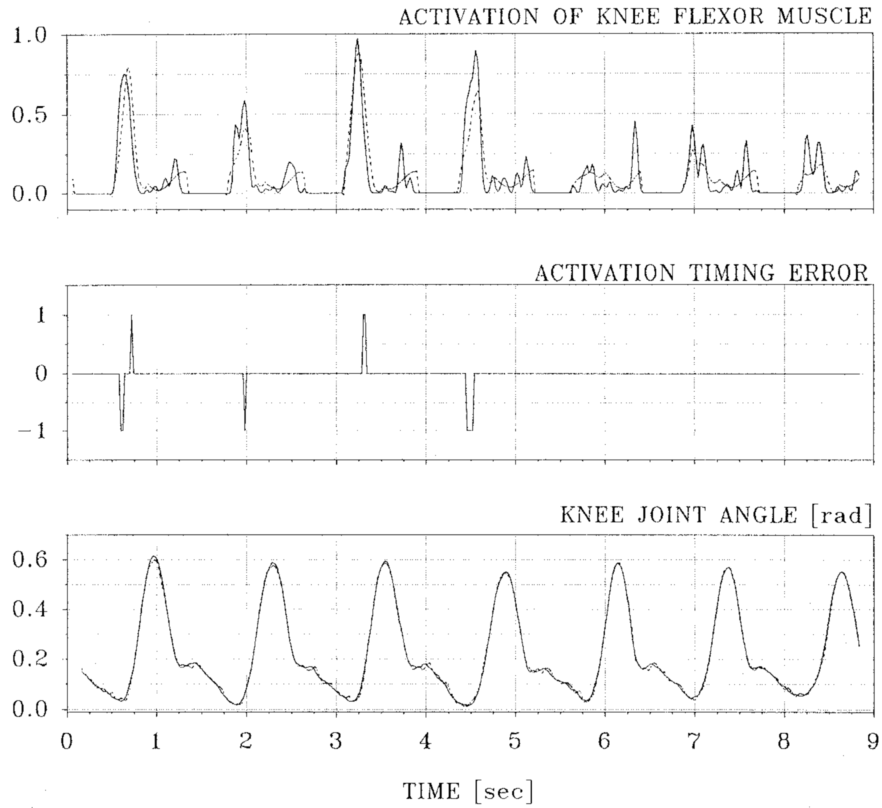


Fig. 5. Results of applying the minimum entropy IL and RBF network with OLS learning algorithm. For details see the caption of Fig. 3.

constant for each neuron. This constant should be big enough to enable neurons to respond strongly to overlapping regions of the input space. The same spread constant is usually selected for each neuron.

C. Results of Using Entropy Minimization Type of IL and RBF Type of ANN

The muscle timing was estimated using minimum entropy IL, and the level of muscle activation was estimated using RBF network with OLS learning algorithm. The RBF network estimates the level very well because it gives a "continuous" output, therefore it does not work so well for the muscle timing estimation [15]. Thus, the rules designed using IL method correct the muscle timing estimated using RBF [15]. The parameters for RBF network were calculated based only on elements from the input and output sets that fall in the interval in which a muscle was estimated to be active. The joint angle was estimated using only RBF network.

Fig. 5 shows results of applying the minimum entropy IL and RBF network with OLS learning algorithm. The top panel shows the predicted (dashed line) and simulated (full line) activation patterns of the equivalent knee flexor muscle. The inputs for this pattern matching were the knee joint angle and the ground reaction forces from the preceding 50 ms, while the output was the muscle activation pattern. The middle panel shows the errors in the muscle activation timing. The bottom panel shows the simulated (full line) and the predicted (dashed line) knee joint angle. The inputs for this pattern matching were the knee joint angle preceding for the 50, 100, and 150 ms, while the output was the knee joint angle. Two different networks were used for matching of muscle activation and joint angle.

The spread constant for the RBF network estimating muscle activity was selected at 80 000, and the obtained network had 290 nodes in the hidden layer. The spread constant for the RBF network estimating the joint angle was selected at three, and the obtained network had 455 nodes in the hidden layer. Each input signal was divided into 30 fixed levels (chosen potential thresholds) for the application of minimum entropy IL. The spread constant and the number of potential thresholds were selected to get as good as possible pattern matching. The number of training epochs in both cases was one. The cross correlation between the desired and estimated muscle activity was 0.90, and cross correlation between the desired and estimated value of joint angle was 0.999. The muscle activation pattern matching lasted approximately 10 s, while the joint angle pattern matching lasted approximately 31 s.

VI. Discussion

In this paper three supervised ML's for prediction of muscle activation patterns (knee flexor muscles) and sensory data (knee joint angle) were compared. The data for training and testing was prepared using the results of the simulation of a walking with an FES system. The ML's are: 1) a MLP with Levenberg–Marquardt modification of BP learning algorithm; 2) an ANFIS where a subtractive clustering method was used for initial identification of a first-order Sugeno-type FIS and an adaptive-network with BP and LS algorithms was used

TABLE I
THREE CRITERIA FOR EVALUATION OF MLS: 1) THE CROSS CORRELATION BETWEEN SIMULATED (DESIRED) AND ESTIMATED SIGNALS (KNEE FLEXOR MUSCLE ACTIVATION AND KNEE JOINT ANGLE); 2) THE ERRORS IN THE TIMING OF MUSCLE ACTIVATIONS; AND 3) THE ELAPSED TIME FOR THE TRAINING AND TESTING

	maximum of t_{er} [s]	*cor*		elapsed time for mapping and test [s]	
		joint angle	muscle activity	joint angle	muscle activity
MLP	0.30	0.999	0.94	18	22
ANFIS	0.22	0.999	0.95	46	53
IL with RBF network	0.08	0.999	0.90	31	10

for tuning of initial identified FIS; and 3) combination of an entropy minimization type of IL technique and a RBF type of ANN with OLS learning algorithm.

The presentation includes only seven strides. Four strides were used for the training, while the whole sequence for the testing. The small number of strides was intentionally used to demonstrate that even though the repeatability is very low from stride to stride, and the limited number of strides is used, all ML's employed are capable of predicting the outputs within acceptable margins of error. The aim of the study was to analyze which technique is the best for design of the rules for an FES controller, and not to get the perfect map of muscle activations. The simulation used to get the inputs and the outputs for pattern matching provided only one plausible sensory-motor map, and its accuracy is questionable because it utilizes a very simplified biomechanical model of a human walking. Therefore, the obtained map of muscle activations only represents a starting point for fitting an FES controller to a person with disability. We tested all three ML's using longer sequence of level walking with up to 50 successive strides, up and down the stairs walking and the like, and the results obtained are very similar to the one presented here, but they are not included because they could not help in explaining the differences between the different pattern matchings.

It is noteworthy that the ML's tested require ad-hoc selection of a number of elements such as a number of nodes in a hidden layer, a clustering radius, etc. We listed the values that were chosen after thorough testing and inspecting the cross correlation, timing error, and complexity aiming to get the best pattern matching.

Table I summarizes some of the elements which can be used for assessing the efficacy and quality of ML's. Those elements are: cross correlation between simulated (desired) and estimated signals, errors in the timing of muscle activations, and elapsed time for the training and testing.

The advantage of a rule-based learning method (e.g., minimum entropy IL method and fuzzy logic) is that the rules determined are both explicit and comprehensible, while the rules used by the ANN's (e.g., MLP and RBF network) are implicit within their structure and not easily comprehensible. There are methods which extract approximate classification rules from a trained ANN, and they help in evaluating the learned knowledge [30]. Furthermore, ANN's are computationally intensive. In view of the versatility of ANN and rule-based learning methods, their combination can be expected to exhibit many advantageous features such as: 1) the parameters of the system have clear physical meanings, which they do not have in general ANN; and 2) human linguistic descriptions or prior expert knowledge can be directly incorporated, for example, into fuzzy neural network structure. On the other hand, the disadvantage is that the network structure requires a large number of term nodes and there is no efficient process for reducing the complexity of combined neural network with rule-based method.

RBF networks and MLP's are examples of nonlinear layered feed-forward networks. These networks are both universal approximators. The RBF network with supervised learning has some advantages over the MLP: 1) the RBF network with supervised learning of cluster centers as well as network weights has characteristic fast training; often it can be designed in a fraction of the time it takes to train the MLP with a BP learning algorithm, even the RBF network may require more nodes than the MLP; 2) this RBF network is able to exceed the generalization performance of the MLP with a BP algorithm substantially [9]; and 3) the spread constant is the only element which has to be selected for this RBF network. The idea employed by a RBF network is very similar to a fuzzy logic method. The output from a radial basis neuron can be interpreted as a membership of input vector to that neuron.

Better prediction of the knee joint angle was obtained compared to the prediction of the activation pattern of the knee flexor muscle for all three used ML's when using the same input and output sets for training and testing. The knee joint angle was predicted with sufficient accuracy by each applied technique (cross correlation was 0.999). The level of knee flexor muscle activity was predicted in a similar way by each applied technique (cross correlation was 0.94 for the MLP, 0.95 for the ANFIS, and 0.90 for the combination of minimum entropy IL and RBF network), but not the muscle timing. The best results for prediction of muscle timing were obtained by using a minimum entropy IL with RBF network, and the worst by MLP (maximum of muscle activation timing error was 300 ms for the MLP, 220 ms for the ANFIS, and 80 ms for the combination of entropy minimization IL and RBF network).

In order to assess the acceptability of the muscle timing errors, they should be related to the dynamic characteristics of the musculoskeletal system: the dynamics of muscles can be characterized by rise time of approximately 50–100 ms, depending on the muscle type. The resonance frequency of the skeletal system depends on the phase of the walking cycle (stance or swing), but can be estimated to be lower then 1 Hz. Therefore, timing errors on the order of 50 ms seem to be acceptable. Because simulated studies [49] show that the gait pattern is more sensitive to muscle stimulus on/off timing than to stimulus amplitude, muscle activity timing errors obtained by MLP and ANFIS are less acceptable than other.

The small number of rules being explicit and comprehensible was obtained by using the first order Sugeno-type FIS network. The tuning procedure is not too complicated because it requires only the cluster radius and the number of epochs to be assumed, while the remaining elements are determined automatically. The other two methods give a large number being implicit and very difficult to comprehend. The use of the RBF network is by far the easiest to implement because it requires only the spread constant to be assumed, and everything else is automatic. The most complicated is the MLP pattern matching because it requires a large number of parameters to be adopted.

The overall goal of the study was to develop a procedure of acquiring knowledge needed for the better control of bipedal walking when using a functional electrical stimulation system. It was postulated that control of bipedal locomotion relies on prestored synergistic actions. The specific questions that were answered relate to the plausible use of machine learning techniques for acquiring nonexisting knowledge needed for real-time rule-based control with an assistive system. The kinematics of an able-bodied subject walking was used as a desired trajectory which has to be cloned. The kinematics and dynamics of the walking were used also for two other purposes: 1) calculating the patterns of muscle activities that are required to achieve the tracking of the desired kinematics [39]; and 2) determining a mapping between the kinematics and patterns of muscle activities by using a machine learning technique. The biomechanical model used for simulation was fully customized using geometrical, inertial and other dynamic parameters recorded from a subject with disability who will eventually use the assistive system [39].

The result of the study will be included into a clinical procedure. Determining individual skeletal and neuromuscular properties of an eventual user of the system is needed in order to simulate the locomotion and determine the pattern which has to be cloned. The simulation results would be used for the training of a machine learning technique, and the results would be the rules for rule-based control of locomotion. This rule-based control would be used as the initial protocol for functional electrical stimulation. The same procedure is directly applicable for control of an externally controlled artificial leg.

ACKNOWLEDGMENT

The authors would like to acknowledge the valuable suggestions and help in editing the manuscript to Dr. R. B. Stein from the Division of Neuroscience, University of Alberta, Edmonton, Alt, Canada, and to Dr. M. Popovic from ETH Zürich, Zürich, Switzerland.

REFERENCES

[1] B. Aeyels, L. Peeraer, J. Van der Sloten, and G. Van der Perre, "Development of an above-knee prosthesis equipped with a microprocessor-controlled knee joint: First test results," *J. Biomed. Eng.*, vol. 14, pp. 199–202, 1992.

[2] B. J. Andrews, R. W. Barnett, G. F. Phillips, C. A. Kirkwood, N. Donaldson, D. Rushton, and T. A. Perkins, "Rule-Based control of a hybrid FES orthosis for assisting paraplegic locomotion," *Automedica*, vol. 11, pp. 175–199, 1989.

[3] M. A. Arbib, *Brain, Machines and Mathematics*. New York: Springer-Verlag, 1986.

[4] A. Bar, P. Ishai, P. Meretsky, and Y. Koren, "Adaptive microcomputer control of an artificial knee in level walking," *J. Biomechanical Eng.*, vol. 5, pp. 145–150, 1983.

[5] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Functional Algorithms*. New York: Plenum Press, 1981.

[6] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Networks*, vol. 2, pp. 302–309, 1991.

[7] S. Chiu, "Fuzzy model identification based on cluster estimation," *J. Intell. Fuzzy Syst.*, vol. 2, 1994.

[8] H. J. Chizeck, R. Kobetic, E. B. Marsolais, J. J. Abbas, I. H. Donner, and E. Simon, "Control of functional neuromuscular stimulation system for standing and locomotion in paraplegics," *Proc. IEEE*, vol. 76, pp. 1155–1165, 1988.

[9] S. Haykin, *Neural Networks, A Comprehensive Foundation*. New York: Macmillan College, 1993.

[10] H. Hellendoorn and C. Thomas, "Defuzzification in fuzzy controllers," *J. Intell. Fuzzy Syst.*, vol. 1, pp. 109–123, 1993.

[11] B. Heller, P. H. Veltink, N. J. M. Rijkhoff, W. L. C. Rutten, and B. J. Andrews, "Reconstructing muscle activation during normal walking: A comparison of symbolic and connectionist machine learning techniques," *Biolog. Cybern.*, vol. 69, pp. 327–335, 1993.

[12] K. B. James, R. B. Stein, R. Rolf, and D. Tepavac, "Active suspension above-knee prosthesis," in *Proc. 6th Int. Conf. Biomech. Eng.*, J. C. Goh and A. Nathan, Eds., 1990, pp. 317–320.

[13] J. S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference systems," *IEEE Trans. Syst. Man, Cybern.*, vol. 23, pp. 665–685, 1993.

[14] S. Jonić and D. Popović, "Rule-based controller for locomotion—Use of radial basis function ANN," in *Proc. Neurel'97*, Belgrade, Yugoslavia, 1997, pp. 49–52.

[15] ——, "Machine learning for prediction of muscle activations for a rule-based controller," in *Proc. IEEE Ann. Conf. on EMBS*, Chicago, Oct. 1997, pp. 1781–1784.

[16] G. Khang and F. E. Zajac, "Paraplegic standing controlled by functional electrical stimulation: Part I—Computer model and control-system design," *IEEE Trans. Biomed. Eng.*, vol. 36, pp. 873–884, 1989a.

[17] ——, "Paraplegic standing controlled by functional electrical stimulation: Part II—Computer simulation studies," *IEEE Trans. Biomed. Eng.*, vol. 36, pp. 885–893, 1989b.

[18] C. A. Kirkwood, B. J. Andrews, and P. Mowforth, "Automatic detection of gait events: A case study using inductive learning techniques," *J. Biomed. Eng.*, vol. 11, pp. 511–516, 1989.

[19] R. Kobetic and B. Marsolais, "Synthesis of paraplegic gait with multichannel functional neuromuscular stimulation," *IEEE Trans. Rehab. Eng.*, vol. 2, pp. 66–78, 1994.

[20] A. Kostov, D.Popović, R. B. Stein, and W. W. Armstrong, "Learning of EMG patterns by adaptive logic networks," in *Proc. Ann. IEEE Conf. EMBS*, San Diego, CA, Oct. 1993, pp. 1135–1136.

[21] A. Kostov, R. B. Stein, D. B. Popović, and W. W. Armstrong, "Improved methods for control of FES for locomotion," in *Proc. IFAC Symp. Biomed. Model.*, Galveston, TX, 1994, pp. 422–428.

[22] A. Kostov, B. Andrews, D. Popović, R. B. Stein, and W. W. Armstrong, "Machine learning in control of locomotion," presented at *IEEE Conf. EMBS*, Baltimore, MD, 1994.

[23] ——, "Machine learning in control of functional electrical stimulation (FES) for locomotion," *IEEE Trans. Biomed. Eng.*, vol. 42, pp. 541–552, 1995.

[24] A. Kostov, Machine learning techniques for the control of FES-assisted locomotion after spinal cord injury, Ph.D. dissertation, Univ. Alberta, Edmonton, Alt, Canada, 1995.

[25] A. Kralj and T. Bajd, *Functional Electrical Stimulation, Standing and Walking after Spinal Cord Injury*. Boca Raton, FL: CRC, 1989.

[26] K. M. Lee, D. H. Kwak, and H. Lee-Kwang, "Fuzzy inference neural network for fuzzy model tuning," *IEEE Trans. Syst. Man, Cybern.*, vol. 26, pp. 637–645, 1996.

[27] C. T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. Comput.*, vol. 40, pp. 1320–1336, 1991.

[28] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE Acoust, Sppech, Signal Processing Mag.*, vol. 3, pp. 4–22, 1987.

[29] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *Int. J. Man-Machine Studies*, vol. 7, pp. 1–13, 1975.

[30] H. Narazaki, T. Watanabe, and M. Yamamoto, "Reorganizing knowledge in neural networks: An explanatory mechanism for neural networks in data classification problems," *IEEE Trans. Syst., Man, Cybern.*, vol. 26, pp. 107–117, 1996.

[31] Z. Nikolić and D. Popović, "Automatic rule determination for finite state model of locomotion," presented at *IEEE Annu. Conf. EMBS*, Baltimore, MD, 1994.

[32] Z. Nikolić, Automatic rule determination for finite state model of locomotion, Ph.D. dissertation, Univ. Miami, Coral Gables, FL, 1995.

[33] Z. Nikolić and D. Popović, "Automatic detection of production rules for locomotion," *J. Automat. Contr.*, vol. 6, pp. 81–94, 1996.

[34] M. N. Oğuztöreli, D. Popović, and R. B. Stein, "Optimal control for musculo-skeletal systems," *J. Automat. Contr.*, vol. 4, pp. 1–16, 1994.

[35] I. Pitas, E. Milios, and A. N. Venetsanopoulos, "Minimum entropy approach to rule learning from examples," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 621–635, 1992.

[36] D. Popović, R. Tomović, D. Tepavac, and L. Schwirtlich, "Control aspects of active above-knee prosthesis," *Int. J. Man-Machine Studies*, vol. 35, pp. 751–767, 1991.

[37] D. Popović, "Finite state model of locomotion for functional electrical stimulation systems," *Progr. Brain Res.*, vol. 97, pp. 397–407, 1993.

[38] D. Popović, R. B. Stein, K. Jovanović, R. Dai, A. Kostov, and W. Armstrong, "Sensory nerve recording for closed-loop control to restore motor functions," *IEEE Trans. Biomed. Eng.*, vol. 40, pp. 1024–1031, 1993.

[39] D. Popović, R. B. Stein, M. N. Oğuztöreli, M. Lebiedowska, and S. Jonić, "Optimal control of walking with functional electrical stimulation: A computer simulation study," *IEEE Trans. Rehab.. Eng.*, vol. 7, Sept. 1998.

[40] M. J. Powell, "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation*, J. C. Mason and M. G. Cox, Eds. Oxford, U.K.: Oxford Univ. Press, 1987, pp. 143–167.

[41] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning interval representation by error propagation," in *Parallel Distributed Processing*. Cambridge, MA: MIT Press, 1986, ch. 8, pp. 318–361.

[42] I. K. Sethi and G. P. R. Sarvarayudu, "Hierarchical classifier design using mutual information," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 4, pp. 441–445, 1992.

[43] M. Sugeno, *Industrial Applications of Fuzzy Control*. Amsterdam, The Netherlands: Elsevier Science, 1985.

[44] R. Tomović, S. Turajlić, and D. Popović, "Active modular unit for lower limb assistive devices," in *Advances in External Control of Human Extremities (ECHE) VII*. Belgrade, Yugoslavia: ETAN, 1981, pp. 1–15.

[45] R. Tomović, "Control of assistive systems by external reflex arcs," in D. Popović, Ed., *Advances in External Control of Human Extremities VIII*. Belgrade, Yugoslavia: ETAN, 1984, pp. 7–21.

[46] R. Tomović, D. Popović, and R. B. Stein, *Nonanalytical Methods for Control of Human Movements*. Singapore, Singapore: World Scientific. Pte. Ltd., Oct. 1995.

[47] S. Watanabe, *Pattern Recognition*. New York: Wiley Interscience, 1985.

[48] R. Yager and D. Filev, "Generation of fuzzy rules by mountain clustering," *J. Intell. Fuzzy Syst.*, vol. 2, pp. 209–219, 1994.

[49] G. T. Yamaguchi and F. E. Zajac, "Restoring unassisted natural gait to paraplegics via functional neuromuscular stimulation: A computer simulation study," *IEEE Trans. Biomed. Eng.*, vol. 37, pp. 886–902, 1990.

[50] L. A. Zadeh, "Fuzzy sets," *Inform. Contr.*, vol. 8, pp. 338–352, 1965.

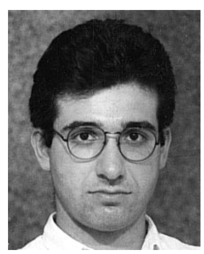
Slavica Jonić (S'97) was born in 1970. She received the B.Sc. degree from the Faculty of Electrical Engineering, Belgrade, Yugoslavia, in 1996. She is currently a graduate student at the Faculty of Electrical Engineering, Belgrade.

Her research interests include modeling of human motor behavior, biological signal processing, and modern computer techniques for application in the control of rehabilitation devices.

Ms. Jonić is a student member of EMBS.

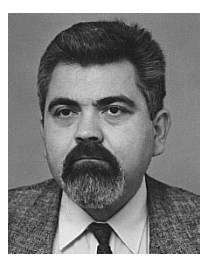
Tamara Janković was born in 1971. She received the B.Sc. degree from the Faculty of Electrical Engineering, Belgrade, Yugoslavia, in 1997.

Her research interests include software and hardware.



Vladimir Gajić was born in 1969. He received the B.Sc. degree from the Faculty of Electrical Engineering, Belgrade, Yugoslavia, in 1997.

His research interests include software and hardware development of biomedical equipment based on microcontroller units.



Dejan Popović (M'91) was born in 1950. He received the B.Sc., M.Sc., and Ph.D. degrees at the Faculty of Electrical Engineering, Belgrade, Yugoslavia, in 1974, 1977, and 1981, respectively.

He is currently Professor of Biomedical Engineering at the Faculty of Electrical Engineering. His scientific interest is in control of movements. He is involved in the design of assistive systems for humans with physical disabilities. He contributed many research papers in journals, books and conference proceedings.

Dr. Popović is a member of EMBS and Yugoslav Society for ETRAN. He is also a member of the Board of Directors of the IFESS, and Vice-President of the YUBEMP.