pHealth 2020
B. Blobel et al. (Eds.)

doi:10.3233/SHTI200620

Control System of a Lower-Extremity Exoskeleton Based on the Artificial Neural Network

David Sebastian MARTINEZ LEMA, Aleksei KARAVAEV, Jan HYBL, Jan HEJDA, Petr VOLF, Patrik KUTILEK[1]
Faculty of Biomedical Engineering, Czech Technical University in Prague, Sitna sq. 3105 Kladno, Czech Republic

Abstract. A lower-extremity exoskeleton can facilitate the lower limbs' rehabilitation by providing additional structural support and strength. This article discusses the design and implementation of a functional prototype of lower extremity brace actuation and its wireless communication control system. The design provides supportive torque and increases the range of motion after complications reducing muscular strength. The control system prototype facilitates elevating a leg, gradually followed by standing and slow walking. The main control modalities are based on an Artificial Neural Network (ANN). The prototype's functionality was tested by time-angle graphs. The final prototype demonstrates the potential application of the ANN in the control system of exoskeletons for joint impairment therapy.

Keywords. Lower-extremity, exoskeleton, control system, neural network, Keras, walking.

Introduction

Contrary to a healthy human skeleton, which supports the body internally, the exoskeleton brace is a device supporting the body externally. Exoskeletons are usually designed to allow patients with mobility disorders to walk or increase strength and endurance [1]. Exoskeletons have several key components. Firstly, the frame, is usually made of lightweight materials. It must be strong enough to support the weight of the body, as well as the weight of the exoskeleton and its components [2]. Secondly, sensors are needed to capture information about how the user wants to move. A controller, then controls the exoskeleton´s actuators. The controller is an on-board computer which takes and processes the information captured by the sensors. This computer coordinates the actuators in the exoskeleton and allows the exoskeleton and its user to stand, walk, climb or descend [3]. The applied control algorithms are one of the key elements of the system which enables the user to optimize movement.

Model-based control systems for exoskeletons are nowadays commonly used. They form part of a dynamic model which can be obtained by identifying the system involved in the movement and then replicating its behavior [4]. Another approach is to drive the

[1] Corresponding Author. Patrik Kutilek, Faculty of Biomedical Engineering, Czech Technical University in Prague, Sitna sq. 3105 Kladno, Czech Republic; E-mail: kutilek@fbmi.cvut.cz.

exoskeleton by a Finite state machine. These algorithms are based on states and transitions. To reach a certain outcome, the control system must undergo several transitions. They have been used in combination with a foot pressure sensor determining whether the system should provide a supportive movement [5]. The gait phase classification method based on neural networks (NN) using sensor signals and foot force sensors to classify gait phases was designed in 2015, see [6]. The Neural Network Toolbox in MatLab 2015 was experimentally used for the NN design [6]. Exoskeletons in gait rehabilitation are often designed to assist limited movements caused by adverse conditions.

This article aims to describe the prototype's design and control system based on NN made using the Keras Python library [7]. The lower extremity brace was developed by the Faculty of Biomedical Engineering, Czech Technical University in Prague in cooperation with Prokyber Ltd. The actuator and control system are expected to support upright standing and allows slow walking by generating supportive torque in the weakened muscles caused by conditions like Arthrofibrosis. The implemented result should be a functional prototype of lower extremity brace control system based on NN.

1. Methods and Systems

1.1. Exoskeleton Structure

The mechanical structure of the smart brace mostly of aluminum, was designed to resemble the biomechanical pulley system of the human knee. It is equipped with three links and two revolving joints. The three exoskeleton links simulate the structure of the human leg consisting of a foot, shank and thigh; the two revolving joints correspond with the ankle movement serving as a pivot and a support point. The knee, actuated by this mechanism consists of gears and pulleys as shown in Figure 1.

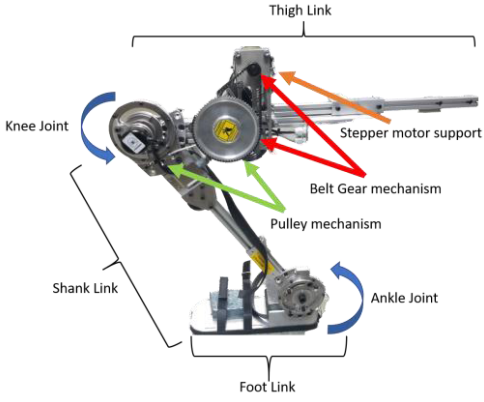

Figure 1. Two DOF lower-extremity exoskeleton

The structure of the exoskeleton is designed to reduce stress in the muscles and knee joint caused by lifting the upper body and thigh linked to the exoskeleton, including all of its components. The required torque for the smart brace was approximated using a basic free body diagram and the calculated moment applied onto the joint [8]. The resulting torque of 21.2 Nm was used for the reference to select the appropriate actuator.

Having reviewed the latest and state of the art equipment available, it was determined that - due to the high accuracy, torque provided and its open loop - a bipolar stepper motor would be most suitable for this application. Based on the structural analysis, the stepper motor LDO-60STH86-3004A, manufactured by LDO Motors was selected. The torque rating of this motor is 2.75 Nm at low rpm, however, when multiplied by the gear ratio of the actuation mechanism, a final torque of 30.25 Nm was obtained. This value is higher than the required torque of 21.1 Nm. Based on these findings, the actuation system would be able to lift the applied load and generate supportive torque. The Rotary encoder BHK 16.05A.0500-I2-5, manufactured by BHK, was selected to determine the angular position of the leg. This encoder was chosen due to the high precision given by the number of pulses per revolution (500 pulses per revolution) and the fact that it operates at 5V, which is the voltage used by most microcontrollers.

1.2. Control System

STMicro STM32 F446 is the microcontroller board used in this prototype. It exploits a customized firmware developed specifically to control smart braces. The embedded functions of this firmware operate at input and output values stored in its registers. The system is controlled by a computer using the Ubuntu PC Linux 20.04LTS operating system for making decisions, sending R/W requests and receiving responses via Modbus RTU from the STM32, operating in a master-slave mode with the device.

The control system is based on the input of the encoder's pulses from its terminals. Internally, the signals from the encoder are transformed into a degree value and assigned to a register where they can be read when needed. Some pins serve as digital inputs, and derive the binary state from the buttons, for control. As an output the STM32 sends three different pulses to the stepper driver denominated: Direction, Enable and Pulse to control the stepper motor. These rectangular pulse signals are in the range of 0V to 5V. The stepper driver DM805-AI (manufactured by Leadshine) was selected as it matches the current rating of 3A required by the stepper motor and for the diversity of control modalities it allows. This combination proved to be particularly useful in the stage of actuator prototyping of the Pulse/Direction functions. To enable access and run the main functions, the control algorithms read and write values from the STM32 registers via Modbus RTU communication, while using their respective addresses. Since the functions use a hexadecimal notation, a specific function, transforming them into floating point values and reversely, was implemented. In this form they were used throughout the Python control algorithms.

An algorithm based on an artificial neural network (ANN) was designed, trained and adapted for control purposes when the intention of movement was to be detected. The algorithm used for NN modelling was designed by using the Keras Python library [7]. To employ a NN for the detection of movement intention in the smart brace, it was also necessary to identify and analyze multiple sources of information to identify the movement desired. The sources for ANN are the angular displacement $\Delta\theta$ of the encoder and the binary states of the interrupters initially placed on the exoskeleton's foot support belt and on the base. This data is used as the input features of the model. The evaluation of leg movement led to the assumption that when the user intends to flex the leg, the angle difference is negative and smaller than some threshold value of -15. Reversely, if the user intends is to extend the leg, the angle difference is positive and bigger than some threshold value of 15. Both of these assumptions take into consideration that if a user

does not press any of the buttons but still tries to move, the angle displacement Δθ will always mean that the user has the intention of movement. If the angle difference is negative and Button 1 on the foot belt is pressed, the system should become more sensitive given that it would have more information to decide that the user is trying to flex their leg. This would be achieved by reducing the threshold value from -15 to -10. The same would occur if Button 2 was pressed while trying to extend the leg; the threshold value would get reduced from 15 to 10. In addition to these instances, the buttons may be pressed accidentally, opposite to the intended direction of movement. However, as the encoder's position considered this to be a movement, and while plausible, it can only be activated after accidentally pressing Button 2 for contraction and surpassing a threshold greater than -25 or accidentally pressing Button 1 for extension and surpassing a threshold of 25. These levels were chosen a safe margin of error and was designed to protect the user.

If a given movement is between the ranges of the positive and negative thresholds, the system cannot determine whether the user wants to move the leg at all and no movement is generated. After the intention to move the leg is detected, there are three possibilities to choose from: to flex, extend or stop the smart brace movement. These outcomes were used as different classes for the classification of the user´s intention and as a simplified estimate of the classes relevance labeled as Very high, High, Very low and Low.

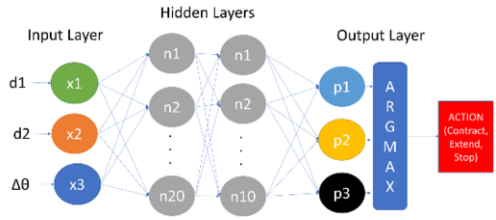

Figure 2. Designed NN for lower-extremity exoskeleton control

To train ANN for classification purposes requires a reasonable amount of testing data. Such data is, however, currently not available. To collect it, the smart brace would have to be tested on many occasions, on many users and under various circumstances. Although comprehensive data mining is beyond the scope of this paper, it is possible to test the potential of this model detecting the intention of movement by generating synthetic data based on the input features and the expected class outputs defined above. This is done with the intention to transfer learning to the live data provided by the smart brace's sensors [9]. To create the data, the thresholds were placed on a table in such a manner that at the threshold value the class with preference has a probability value close to 0.5, which represents half of the total probability. Then the probability of the class with preference is extended incrementally by a factor of 0.2. This increment is parallel to the angle displacement until the value reached is equal to the total probability of 1, from this point the probabilities of the other classes are set to zero. The class preference has the highest probability to become the outcome depending on the input. The values of the Extend and Stop classes are calculated by subtracting the total probability of 1 and the value of the class with preference, time a scaling factor.

The neural network receives three input features, see Figure 2. The decision with the highest probability of whether the user wants to contract, extend, or stop a movement is selected by means of an Argmax function.

Initially, a CSV file is uploaded with the synthetically generated data to train the network by separating the input data from the output values. The Adam gradient descent optimizer is used, where - after getting a prediction - the algorithm goes back to make the necessary adjustments to the weights based on how accurate the model is. The acquisition of new predictions continues until the model has become more accurate.

1.3. Testing of the control system

For the ANN based leg rises modality the model was initialized and weights were loaded into it. Then the initial angular position of the encoder was read followed by a delay. The purpose of the delay is to give the system some time to identify whether the position has changed. Following this step, the actual position is read and the angular displacement $\Delta\theta$ is determined by calculating the actual position minus the initial position. Next, the state of the input buttons is read and put into an array with the angular displacement $\Delta\theta$. This array is passed through the trained neural network and an Argmax function. The Argmax function returns the index of the class with the highest probability. This index value is assigned to the action variable and is read by the if statements; it determines the action to be taken by the system. If the action is equal to 0 the flexion movement is generated. If the action equals 1, the extension movement is the outcome. Finally when the action equals 2, the stop action occurs. These functions are followed by a delay to allow the system to preform them. The flowchart for this algorithm is described on Figure 3.

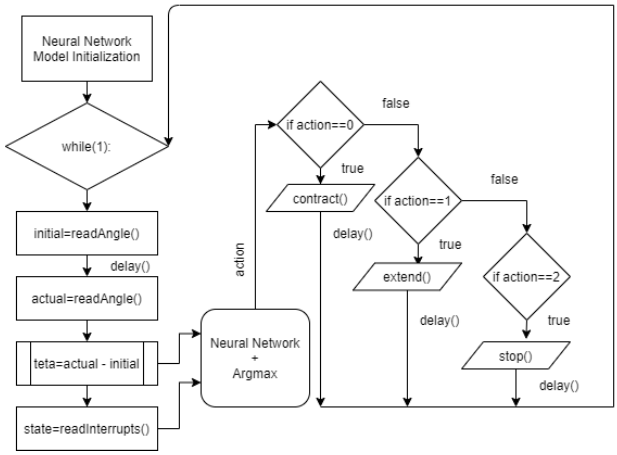

Figure 3. Simplified flowchart of the leg rises control algorithm based on an ANN.

2. Results

The algorithms were written in the Python programming language and use the functions of several libraries. The PyModbus library is used to read the encoder degrees, button signals and activate the stepper motor by Modbus RTU communication. The NumPy library for mathematical operations with arrays and Keras library to initialize the ANN and load the pre-trained weights were also used. Three control modalities were proposed

to allow leg lifting. The result of testing the design of the control system is knee angle control using neural networks, see Figure 4.

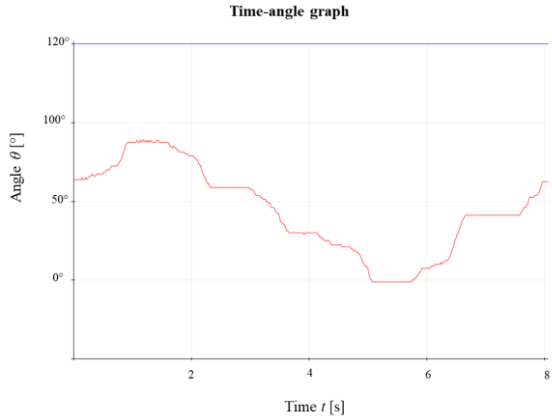

Figure 4. Example of ANN application for mediating leg lifting.

3. Conclusion

The work described above proved that the novel method of control system based on ANN made with the Keras Python library can be used for the control system of lower-extremity exoskeletons to treat joint impairments. This is the first time that an ANN made with the Keras Python library is described and utilized in a lower-extremity smart brace. The results obtained through preliminary testing need be followed by clinical tests before practical application can occur in clinical practice. However, as already stated, it is beyond the focus of the article but the authors feel the first important step has been taken.

References

[1] Viteckova S, Kutilek P, Jirina M. Wearable lower limb robotics: A review; Biocybernetics and Biomedical Engineering 2013; 33,2: 96-105.
[2] Viteckova S, Kutilek P, Boisboissel G, Krupicka R, Galajdova A, Kauler J, Lhotska L, Szabo Z. Empowering lower limbs exoskeletons: state-of-the-art. Robotica 2018; 36,11: 1743–1756.
[3] Chen B, Zi B, Wang Z, Qin L, Liao W. Knee exoskeletons for gait rehabilitation and human performance augmentation: A state-of-the-art. Mechanism and Machine Theory 2019, 134: 499–511.
[4] Anam K, Al-Jumaily A. Active Exoskeleton Control Systems: State of the Art. Procedia Engineering 2012, 41: 988–994.
[5] Shamaei, K, Cenciarini M, Adams A, Gregorczyk K, Schiffman J, Dollar A. Design and Evaluation of a Quasi-Passive Knee Exoskeleton for Investigation of Motor Adaptation in Lower Extremity Joints. IEEE Transactions on Biomedical Engineering 2014. 61,6: 1809–1821.
[6] Jung J, Heo W, Yang H, Park H. A neural network-based gait phase classification method using sensors equipped on lower limb exoskeleton robots. Sensors 2015, 15,11: 27738–27759.
[7] Bloice, MD, Holzinger A. A tutorial on machine learning and data science tools with python, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 2016, 9605: 435-480.
[8] Madeti BK, Chalamalasetti SR, Bolla Pragada SKS. Biomechanics of knee joint - a review. Front. Mech. Eng. 2015, 10: 176–186.
[9] Le TA, Baydin AG, Zinkov R, Wood F. Using synthetic data to train neural networks is model-based reasoning, 2017 International Joint Conference on Neural Networks (IJCNN), 2017, Anchorage, AK; p. 3514-3521.