DOI: https://doi.org/10.53052/9788366249974.35

Vladimír STENCHLÁK[1], Miroslav CÍSAR[2], Ivan KURIC

Opiekun naukowy: Ivan KURIC[3]

KONTROLER STABILNOŚCI ROBOTA MICRO SPOT OPARTEGO O SZTUCZNE SIECI NEURONOWE

Streszczenie: Artykuł dotyczy projektowania kontrolera stabilności opartego na sztucznych sieciach neuronowych. Badania skupiają się na zaproponowaniu metodyki wykorzystania sztucznych sieci neuronowych w tego typu zadaniach. Eksperyment wykorzystuje MATLAB i specjalne zestawy narzędzi do programowania wszystkich niezbędnych części.

Słowa kluczowe: Robot, Spot Micro, Sztuczne sieci neuronowe

INCLINATION STABILITY CONTROLLER FOR SPOT MICRO ROBOT BASED ON ARTIFICIAL NEURAL NETWORKS

Summary: This article deals with the design of a stability controller based on artificial neural networks. This research is focused on the design of a methodology for using artificial neural networks in such tasks. This experiment is using MATLAB and specific toolboxes for programming all necessary parts.

Keywords: Robot, Spot Micro, Artificial Neural Networks

1. Introduction

Nowadays, the neural networks are often used in various applications such as predictive maintenance tasks [1], machine vision tasks [2] and others. Basically these models are solving regression or classification tasks in general. According to [3], universal approximation theorem says, that there is always existing a neural network model with specific size, which is able to approximate custom continuous function.
Inverse kinematics is using kinematic equations for calculating the final pose of the kinematic structure. For example, in the case of industrial applications, the inverse

[1] dr. inž., Department of Automation and Production Systems, Faculty of Mechanical Engineering, University in Žilina, e-mail: vladimir.stenchlak@fstroj
[2] dr. doc. inž., Department of Automation and Production Systems, Faculty of Mechanical Engineering, University in Žilina, e-mail: miroslav.cisar@fstroj.uniza.sk, email
[3] prof. dr. hab. inž. University of Bielsko-Biala, Faculty of Mechanical Engineering and Computer Science, Department of Industrial Engineering: kuric.ivan@gmail.com

kinematics is calculating the rotations of the joints of the kinematic structure, based on the final coordinates of the effector of an industrial robot. The kinematic equations are dependent on the physical properties of the structure such as lengths or angles and thus they can be complex.
That is why, this article is aiming on the possibilities of using artificial neural networks for replacing the kinematic equations as it was done in many other researches [4, 5]. In this research, the kinematic structure of an open-source project Spot Micro is used for further research.

2. Open-source project Spot Micro

This experiment is evaluated on an existing open-source project called SpotMicroAI. This project is using a designed construction of quadruped robot which was originally designed by Deok-yeon Kim [6]. The biggest advantage of this robot is that it can be built for a reasonable price and there are various options how the robot can be controlled. The robot can be controlled in a fully simulated way which is using platforms as pyBullet or ROS. This method is far more complex than using a controlling board Arduino Mega2560 with accelerometer, bluetooth module and a console joystick controller.

2.1. Leg servo motors used in Spot Micro

This robot is using 12 servo motors MG996R, which range of movement is 180° and the powering voltage is 6V. The original version uses a step down voltage regulator, but in this research we used a laboratory power supply, because for the experiment purposes, the robot was mainly stationary. The servo motors are controlled by Arduino Board MEGA2560. Placement of all the servomotors is given in the figure below.

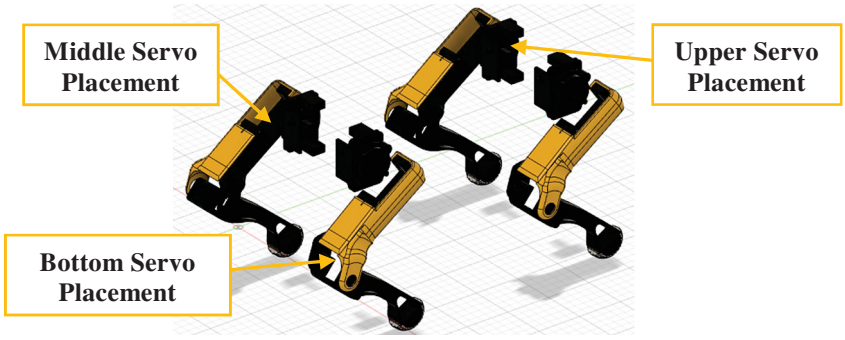

Figure 1. Placement of the controlled servo motors MG996R [source: github.com, edited]

The mentioned motors are controlled by a pulse width modulation (PWM). Used servomotors have stall torque 11kg.cm when they are powered by 6V. Operating speed of the motors at the 6V powering voltage is 0.14s/60°. In this experiment a MATLAB library Servo was used for controlling each motors.

2.2. IMU unit for obtaining accelerometer and gyroscopic data

Other crucial part of this robot is an inertial measuring unit by which we can obtain accelerations in X, Y and Z axes and also gyro data such as roll, pitch and yaw. The placement of this IMU device and all the necessary parameters gathered by this device is illustrated in the figure below.

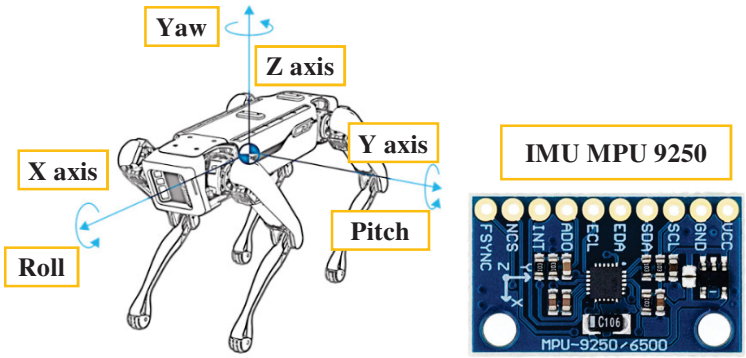

Figure 2. Illustration of pitch, roll and yaw of the quadruped robotic structure [source: bostondynamics.com, edited]

The kinematic analysis of various devices can be a problem, especially in the case of robotics area. There is an inverse kinematic study [7] dedicated to this open-source project. This method was presented at the International Journal of Scientific & Technology Research. Each leg is having 3 degrees of freedom. From the mathematical model and also geometrical model we can get kinematic equations. These equations are only applicable on this kind of robot with specific length of all parts. All the necessary angles of all motors are calculated by obtained kinematic equations. These kinematic equations are then hardcoded in the controlling board of the robot as it was done in the research [7].

The kinematical equations are consisting of transformation matrices. For this matrices it is necessary to obtain rotational matrices R_x, R_y, R_z. Rotation around X axis is calculated by the following equation:

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) & 0 \\ 0 & \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The overall rotation R_{xyz} of the robot is given in the following equation:

$$R_{xyz} = R_x\, R_y\, R_z \tag{2}$$

For the final transformational matrix:

$$T_M = R_{xyz} \begin{bmatrix} 1 & 0 & 0 & x_m \\ 0 & 1 & 0 & y_m \\ 0 & 0 & 1 & z_m \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

where the x_m, y_m, z_m are the coordinates of the center of the body of the Spot Micro quadruped robot. In the figure 3, there are the specific angles around each axes shown. We will calculate each inclination (in radians) by the following equation [8]:

$$\vartheta = \tan^{-1}\left(\frac{\mathrm{AccX}}{\sqrt{\mathrm{AccY}^2+\mathrm{AccZ}^2}}\right), \tag{4}$$

where *AccX*, *AccY*, *AccZ* are the observed accelerations by the IMU MPU 9250.

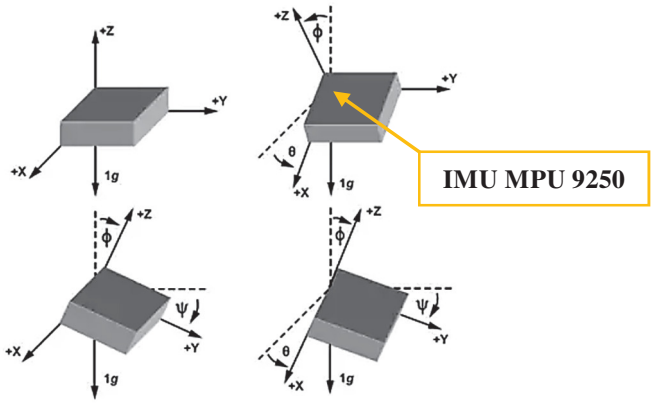

Figure 3. Inclination angles of the IMU device [8, edited]

After those calculations it is necessary to calculate transformational matrices for each leg separately, forward kinematic matrix (also with the forward kinematics elements) and also others. At the end of those calculations we can finally calculate the rotation of every joint of the robot. This inverse kinematic analysis is not the object of this research and it is very well described in [7].

The goal of this research is to design an alternate method for calculating the rotation of joints based on the input data measured by the IMU device. The obtained data such as acceleration in X, Y and Z axis are used for calculating the inclination as it was shown in the previous equation. These inclination angles will be used for calculating the position of each servomotor of the robot, so that in the end, the plane XY of the IMU device will be horizontal with respect to the ground.

3. Programming and controlling the robot in MATLAB

In MATLAB we created a script for testing the designed model of the given neural network. Firstly, we need to access the digital inputs and outputs for controlling all necessary peripherals. We created an object *robotMcu* which allows to control all necessary peripherals of the Arduino Mega 2560 board.

```
robotMcu = arduino('COM4','Mega2560','Libraries','Servo');
```

The COM Port needs to be provided, as well as the type of the used micro controller board, when it is necessary to establish a connection over the USB Port. But there is also a Wi-Fi connection or a Bluetooth connection available for this kind of communication. After running this script, it is possible to control all servo motors by

PWM pins of the used controller in real-time. You can see another input argument for this function and it is a Servo Library, so we can use objects *servo* and function *writePosition()* for positioning the motors in Matlab. For this experiment purposes an USB communication was used. For controlling one leg (right front leg) of the quadruped robot a following script can be used:

```
% Definition of front right leg
servoPPN1 = servo(robotMcu, "D2");
servoPPN2 = servo(robotMcu, "D3");
servoPPN3 = servo(robotMcu, "D4");
```

The servos of the right front leg are attached to the digital outputs pins D2, D3 and D4, which are supporting pulse width modulation. After defining the servomotors, a function *writePosition()* can be used as it is shown in the script bellow:

```
% Positioning the servos
writePosition(servoPPN1, 0.32); %max 0.95, min 0.32
writePosition(servoPPN2, 0.69); %max 0.99, min 0.21
writePosition(servoPPN3, 0.5); %max 0.88, min 0.5
```

There are two input arguments for this function a servo object and the value of the position, in range from 0 to 1 (0°-180°). We can examine limit values by using slider interactive live script components. The positioning is then executed in real-time. This created MATLAB script allows to the operator to set up all servomotor positions of the robot, so it will have a desired pose.

4. Neural network model for calculating the joints rotation the robot

This article presents a possible way how to avoid using an inverse kinematics model by using a deep neural network model. This experiment was done in MATLAB environment by the use of Deep Learning Toolbox which allows you to design, train and deploy designed models for a prediction or classification tasks.
The designed model is consisting of an input layer, which is filled by the inclination angle values measured by the IMU device of the robot. Input data is after that processed in the hidden layers of the deep neural network. In the output of this layer, there is and regression layer which is representing the rotation of each servomotor of the quadruped robot.

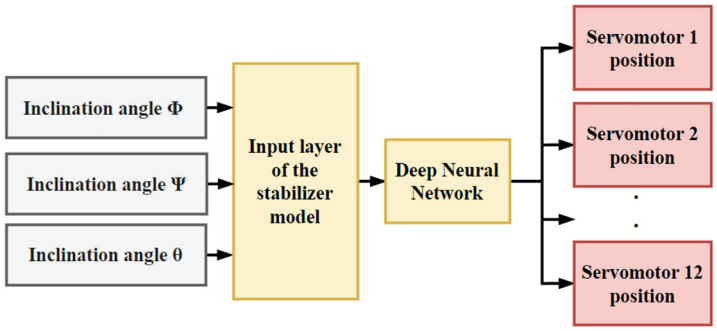

Figure 4. Scheme of the neural network model

4.1. Creating a deep neural network model in MATLAB

For the model which is described above, we can use following script, which is creating all the necessary layers of the model. In this case, we decided to use hyperbolic tangent activation function in all 4 hidden layers. There is also a possibility to rescale the input data by many ways, for example, the input data can be rescaled from 0 to 1.

```
layers = [
            featureInputLayer(3)
            fullyConnectedLayer(50)
            tanhLayer
            fullyConnectedLayer(150)
            tanhLayer
            fullyConnectedLayer(200)
            tanhLayer
            fullyConnectedLayer(25)
            tanhLayer
            fullyConnectedLayer(12)
            regressionLayer...
        ];
```

4.2. Training a deep neural network model in MATLAB

When all the training data are prepared and pre-processed and the model is created. The model can be trained. There are several training algorithms, which are also compatible for using CUDA cores for all the calculations during the model training. It is necessary to define option parameters for the training algorithm. The example script how to define the training parameters is given in the following script:

```
options = trainingOptions('adam','MaxEpochs',100, ...
    'MiniBatchSize',5, ...
    'Shuffle','every-epoch', ...
    'Plots','training-progress', ...
'Verbose',true,'ValidationData',{validationInputs,validation
Responses});
```

In this article, there is used an adaptive-momentum estimation method [9]. It is possible to put a validation dataset into the training, so the performance of the model will be checked during that training. Validation data set is often used to prevent an overfitting problem of the network. In a case of overfitting, the model wouldn't be able to estimate the positions of each servomotor correctly.

4.3. Creating training data set for the inclination stabilizer neural network

Creating the training data set for this experiment is the most crucial and time consuming part. This task of creating a training dataset will be done in the next phase of research. The process of generating training samples consists of several steps which are shown in the figure below.

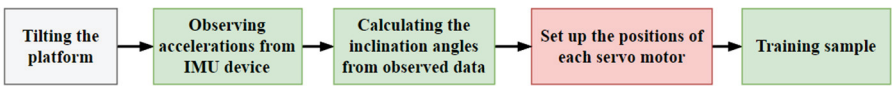

Figure 5. Specific steps for obtaining a simple training sample

In the first step it is necessary to manually tilt the platform on which the quadruped robot will be standing. On this platform another IMU device is mounted so it will be possible to obtain the actual acceleration values on the tilted platform. The schematic illustration of the platform is given in the figure below.

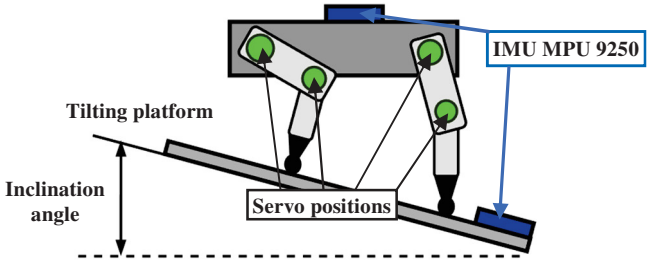

Figure 6. Tilting platform for obtaining the training samples

The mentioned IMU device is using I2C library which is using two digital pins for communication (SDA and SCL). The script which is defining the IMU device is given below:

```
IMUdevice =
mpu9250(robotMcu,"SampleRate",fs,"SamplesPerRead",pr,"TimeFo
rmat","datetime","OutputFormat","matrix");
```

The sample rate of the device is specified in the parameter *fs*, the amount of samples which are read by a single read is given in the parameter *pr*. In the script given above, there is also specified the output format of the received data.
After obtaining the accelerations, the inclination angles are calculated by using the equation (4). After this step, the servo motors positions are set up, so the inclination angles are 0. It means that the IMU device on the robot is horizontal with respect to the ground. After this step is complete a training sample can be saved to the training dataset.

5. Conclusion

This article describes in more detail the possibilities how to design a neural network for avoiding the kinematic equations for calculating specific servo positions for establishing desired pose of the given kinematic structure. This paper presents a way how to create a training dataset for this machine learning task.
In this article there are also provided MATLAB scripts for controlling the Spot Micro by using the MATLAB framework. In this research MATLAB Deep Learning toolbox was used for the designing the model. This tool allows to use GPU device with support of CUDA for the training of the model. This approach significantly decreases the amount of training time of the artificial neural network.
In the next phase of this research, the platform for obtaining the relevant training samples will be constructed. After this step, the training dataset will be created as well as the model will be trained and verified.

ACKNOWLEDGMENT

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-16-0283. Project title: Research and development of multi-criteria diagnostics of production machines and devices based on the implementation of artificial intelligence methods.

REFERENCES

1. KURIC I., KLAČKOVÁ I., DOMNINA K., STENCHLÁK V., SÁGA M. Jr. Implementation of Predictive Models in Industrial Machines with Proposed Automatic Adaptation Algorithm. Appl. Sci. 2022, 12, 1853.
2. GUOSHU Z.,QIUYU Z. Unsupervised Image Feature Extraction Based on Scattering Transform and Self-supervised Learning with Highly Training Efficiency. 2019 J. Phys.: Conf. Ser. 1237 032044.
3. NIELSEN M.: A visual proof that neural nets can compute any function: *http://neuralnetworksanddeeplearning.com/chap4.html.*
4. ARAVINDDHAKSHAN S., et al. Neural Network Based Inverse Kinematic Solution of a 5 DOF Manipulator for Industrial Application. 2021 J. Phys.: Conf. Ser. 1969 012010.
5. BINGUL Z., ERTUNC H. M., OYSU C. Applying Neural Network to Inverse Kinematic Problem for 6R Robot Manipulator with Offset Wrist. 2005. In: Ribeiro, B., Albrecht, R.F., Dobnikar, A., Pearson, D.W., Steele, N.C. (eds) Adaptive and Natural Computing Algorithms. Springer, Vienna. https://doi.org/10.1007/3-211-27389-1_27.
6. DEOK-YEON K. Spot Micro – robot dog. 2019. *https://www.thingiverse.com/thing:3445283.*
7. ŞEN M.A., BAKIRCIOĞLU V., KALYONCU M., Inverse Kinematic Analysis Of A Quadruped Robot, Int. J. Sci. Technol. Res., 2017, 6 (09) 285–289.
8. FISHER C. J., Using An Accelerometer for Inclination Sensing. 2011. *https://www.digikey.com/en/articles/using-an-accelerometer-for-inclination-sensing.*
9. KINGMA D. P., BA J., Adam: A Method for Stochastic Optimization, 2017, https://doi.org/10.48550/arXiv.1412.6980.