ISSN 2007-9737

Training and Transfer of a PID Balance Controller for Quadruped Robots Using Artificial Neuronal Networks

Francisco José López-Cortés[1,*], Carolina Jiménez-Martínez[2], Rigoberto Cerino-Jiménez[1], Fernando Perez-Tellez[3], David Pinto[1]

[1] Benemérita Universidad Autónoma de Puebla, Facultad de Ciencias de la Computación, Puebla, Mexico

[2] Instituto Politécnico Nacional, Escuela Superior de Ingeniería Mecánica y Eléctrica, Mexico City, Mexico

[3] Technological University Dublin, School of Enterprise Computing and Digital Transformation, Dublin, Ireland

{franciscojlc21, carojimar63}@gmail.com, cerino_rigoberto@hotmail.com, fernando.pereztellez@tudublin.ie, david.pinto@correo.buap.mx

Abstract. Maintaining balance in quadruped robots requires precise coordination of joint movements, which varies depending on the unique physical characteristics of each robot, such as dimensions, mass distribution and centre of gravity. There are several control methods for quadruped robots balance, a commonly used is the Proportional Integral Derivative (PID) controllers, provide robust stability, but typically require individualised tuning for each robot due to these varying physical parameters. To address this limitation, this paper explores an intelligent control strategy that leverages neural networks to generalise balance control across different quadruped platforms. Initially, a PID controller was implemented to create a large dataset by controlling the equilibrium of a commercial 12 joints quadruped robot. This data is used to train a several perceptron neural networks to learn the complex mapping of body orientation to joint movements. Through a parameter search, it was determined that a simple single-layer neural network with 18 neurons effectively mimicking the behaviour of the PID controller. This neural network is then applied to a secondary quadruped robot with different dimensions and mass, demonstrating that single-layer networks, despite their simplicity, can effectively capture essential control dynamics, reducing model complexity and enabling rapid deployment on different quadruped robots. Furthermore, this work opens the way to scalable and adaptive control methods in robotic systems where neural networks trained on one platform can be effectively transferred to others with minimal modifications.

Keywords. Quadruped robot, balance controller, ANN controller, knowledge transfer.

1 Introduction

Quadruped robots have proven to be a versatile solution in the field of mobile robotics, excelling in applications where mobility over rough and difficult terrain is essential. Unlike wheeled or bipedal robots, quadrupeds offer greater stability and adaptability to complex environments, such as search and rescue areas [1, 2], planetary exploration [3, 4], industrial inspection in rough terrain [5, 6], and operations in human-hazardous environments [7].

Their ability to maintain stability while negotiating obstacles or uneven surfaces, together with the redundancy of their support points, make them the preferred choice for tasks requiring robustness in difficult terrain.

However, quadruped robots are mechanisms that present a high complexity in their control,

whether for walking, avoiding obstacles, adapting to the terrain and maintaining balance, the latter being a very important property since, not having a base or support fixed to the plane where they move, they must control their balance to avoid falling.

In addition, the complexity of the control of these robots lies in the fact that they have many degrees of freedom and are redundant, each leg of the robot must be precisely controlled to ensure smooth locomotion, avoiding falls or imbalances that could compromise performance.

Technically, this requires sophisticated algorithms that manage not only the positioning of the legs, but also the orientation of the body. The high dimensionality of the control makes the task computationally intensive and challenging in terms of controller design.

There are several known methods for controlling quadruped robots by merging balance control with gait pattern control, each with its strengths and weaknesses. One of these traditional methods is model-based dynamic control [8, 9], which describe complete dynamics of the robot using mathematical models, generating optimal gait patterns [10] in terms of energy efficiency and stability.

Although accurate, its main drawback is its high computational complexity, which may limit its application in real time, and its poor performance in unpredictable environments.

To manage real-time operation using a mathematical model, approaches such as Model-based Predictive Control (MPC) have been developed [11]. MPC uses detailed models of robot dynamics to anticipate future movements and calculate the necessary adjustments in real time to maintain balance and stable posture as seen in [12, 13].

This approach adjusts both leg movement and body posture based on predictions of future states, considering possible perturbations and optimizing actions to minimize loss of balance.

Furthermore, to minimize complexity and reduce control dimensionality some approaches focus on the analysis of center of mass (COM) and controlling its position by means of its legs. An example is Zero Moment Point (ZMP) control, balance and posture are regulated by constantly monitoring the position of the zero moment point

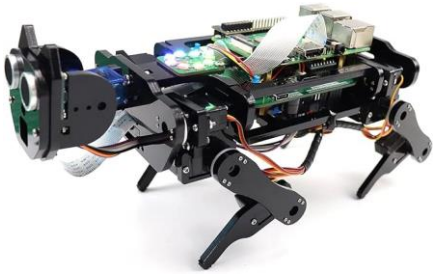

Fig. 1. Commercial quadruped robot "Robot dog kit" manufactured by the company Freenove

and adjusting the posture of the legs so that the ZMP remains within the support polygon as in [14, 15].

This approach allows for stable posture and effective balance control in static or low-speed tasks, as the robot ensures its balance by maintaining the center of mass within the support polygon. However, in dynamic movements or changing terrains, ZMP control may not be sufficiently fast or adaptive, as it does not allow for anticipation of major disturbances.

Moreover, the Inverted Pendulum Model (IPM) managed the balance by simulating the centre of mass of the robot as a control point to be aligned on the support polygon, acting as the "arm" of the pendulum around which the robot swings as shown in [16, 17]. By simplifying the complex dynamics to a general equilibrium model, IPM allows rapid response to COM movement in moderately variable terrain.

The accuracy of posture control depends on the accuracy of the model and simplifying assumptions such as rigid contact limits its applicability in environments with irregular surfaces and constantly changing contact points.

In contrast to the use of mathematical models, model-free quadruped robot controllers have emerged that focus on replicating biomechanical or biological processes, such as patterned, rhythmic neural outputs that drive rhythmic behaviours. The Central Pattern Generators (CPGs) [18] mimics the rhythmic gait patterns observed in biological organisms. CPGs can generate cyclic sequences of joint activation that perform periodic movement of each leg.

The system combining with Reinforcement Learning automatically adjusts the phase of movement based on basic sensory feedback,

adapting to small perturbations without the need for complex modelling such in [19, 20]. CPGs are limited in their ability to respond to large disturbances or highly irregular environments, which has an impact on the robot's posture accuracy and overall stability.

The computational efficiency of CPGs makes them suitable for real-time operations, but because they do not incorporate an explicit dynamic equilibrium model.

As seen in the different methods of controlling quadruped robots, balance control is an important goal in the locomotion task. The robot must compensate for the imbalance with specific movements of each leg. Mathematical modeling is necessary to know how to move every joint, but it is difficult to obtain and is unique for each robot.

Given this scenario, there is a need to develop a model-free controller that would be capable of control only the balance to reduce complexity and generalizing the balance control to quadruped robot architectures with different physical configurations.

In this work we present the develop of a compact neural network model that extract a balance PID controller behavior from an existing one, capturing basic balance dynamics and then the learned model can be transferred to another quadruped robot of different size, that was developed from scratch.

The search for optimal parameters and the analysis of the appropriate neural network to learn the balance control was fundamental to maintain computational simplicity and improving control flexibility without the need to manually adjust each parameter of the new robot.

2 Methodology

For this research work we used a commercial quadruped robot called Robot dog kit manufactured by the company Freenove, shown in figure 1, which is a quadruped robot with acrylic structure, it has (MG90) servomotors which can be controlled by position using PWM. It also has the necessary electronics for the robot to operate with lithium-ion batteries and a raspberry pi 4 to control the motors through a user interface.

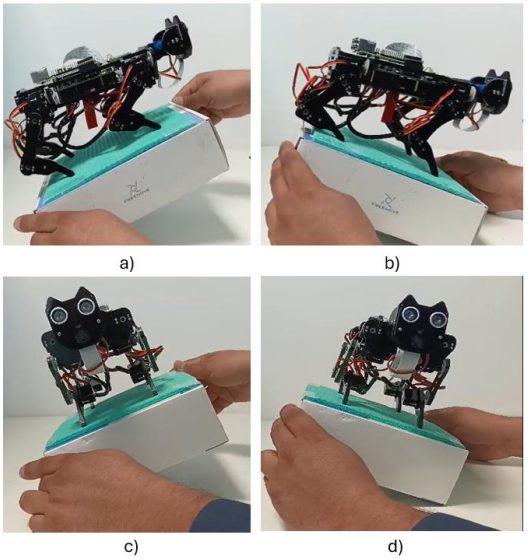

Fig. 2. Movements made to extract the database, a) and b) show the movement in pitch and c) and d) show the movement in Roll

This robot has predefined functions such as forward, backward, sideways walk and turning in both directions. This is achieved by using the inverse kinematics of the mechanism to subsequently calculate the trajectory to be followed by each of its limbs. We know that this task involves the analysis of the mechanism, its dimensions and type of morphology.

The robot has an inertial measurement unit (IMU) that combines a three-axis accelerometer and gyroscope. The accelerometer measures linear acceleration in the X, Y and Z directions, including acceleration due to gravity, allowing it to determine the tilt or angle of orientation of the device. The gyroscope, on the other hand, measures angular velocity in the same directions, providing crucial information about rotations and angular movements.

Being a commercial robot, it has mathematical equations that detail the orientation of the system by means of the information acquired by its inertial measurement unit, as well as Kalman filters and techniques that allow obtaining the Euler angles (Roll, Pitch, Yaw). An important function of this system is the balance control of the quadruped robot, which gives the robot the ability to react when it is on an inclined plane, compensating this

ISSN 2007-9737

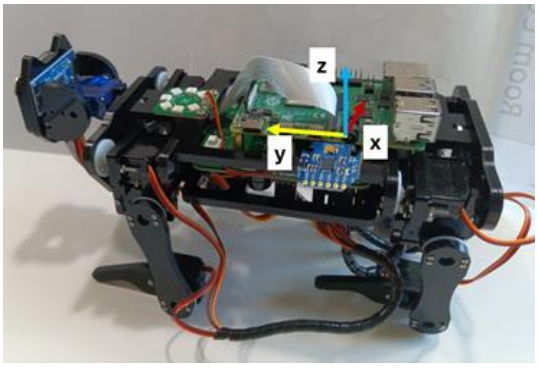

Fig. 3. Inertial reference axis to generate the artificial neural network

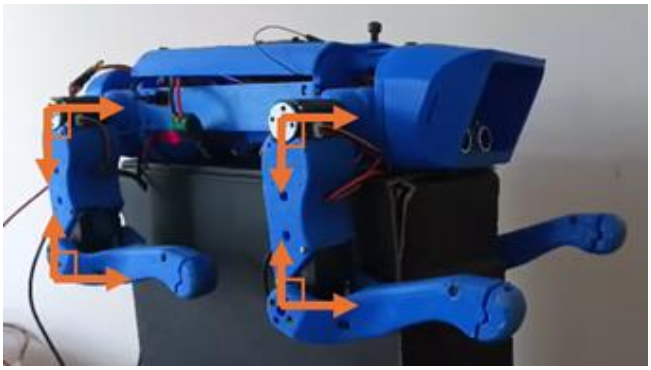

Fig. 4. Robot assembly with motors at 90

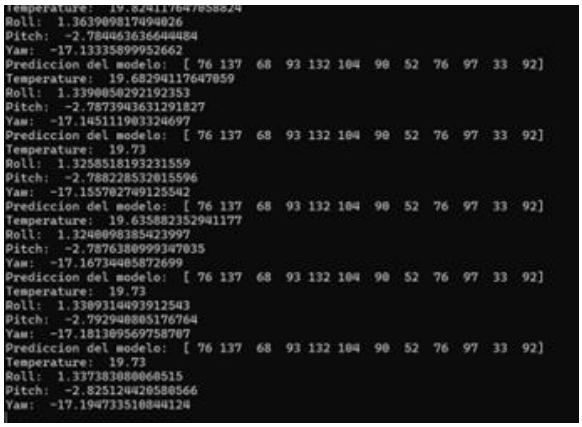

Fig. 5. Roll, Pitch and Yaw values with filters applied and the neural network inference for these values

inclination with the movement of its legs so that the robot's torso is always horizontal, thus balancing the system.

To achieve this task, a PID controller is used which aims to manipulate the variable Euler angles mentioned above, and thus, with the help of rotation matrices, Kalman filters and the morphology of the robot can achieve control of the orientation of the system, it should be noted that the robot also makes use of inverse kinematics to know the values that must be provided to the motors in angle format.

Having all this information contained in the source code of the commercial robot, it is proposed to extract this knowledge by means of a neural network, which will aim to learn the behaviour of the PID controller on the variables of the Euler angles provided by the imu.

2.1 Dataset

To achieve this task, the first step is to acquire the database of the PID controller's behaviour on the motors, i.e. it is necessary to extract the behaviour of the PID through all possible measurements and the actions that the motors generate to compensate for the tilt movements registered by the sensor.

The robot needed to be on a platform where tilt movements were performed to obtain the sensor measurements and the action of the controller in inclined planes. Thus, the robot was placed on a platform and through the movement of the platform, as shown in figure 2, the database was generated, which consists of 49,876 data, and a database for validation of 4,067 data was also extracted.

Table 1. Comparative table of MSE and MAE metrics of 1-layer neural networks with different number of neurons

N° Neurons	MSE	MAE
18	0.26062	0.25241
17	0.31753	0.30081
14	0.32096	0.30351
19	0.32676	0.29505
10	0.34511	0.30179
13	0.34582	0.31588
12	0.41526	0.37572
11	0.53823	0.39626
5	0.54264	0.40549
4	0.54407	0.41224
16	0.55667	0.43297
15	0.55823	0.40960
8	0.56823	0.43116
3	0.56898	0.43218
9	0.56915	0.42769
6	0.56921	0.43048
7	0.86466	0.61360
20	1.18497	0.76728
2	10.06871	2.16316
1	46.45846	4.66416

Table 2. Comparative table of MSE and MAE metrics of 2-layer neural networks with different number of neurons

Neurons per layer	MSE	MAE
14 -14	0.23634	0.23299
8 -8	0.25719	0.25164
16 -16	0.27813	0.27039
17 -17	0.27996	0.27148
10 -10	0.28447	0.27684
12 -12	0.28803	0.28222
4 - 4	0.32811	0.29782
18 - 18	0.34210	0.32788
11 – 11	0.35039	0.33671
13 – 13	0.35077	0.31601
6 – 6	0.35464	0.32212
19 – 19	0.37177	0.34225
5 – 5	0.38674	0.35156
9 – 9	0.38730	0.36499
15 – 15	0.40518	0.37782
20 – 20	0.46303	0.42260
3 – 3	0.53846	0.42620
7 – 7	0.80416	0.60000
2 – 2	47.11017	4.66455
1 – 1	2278.72757	40.30010

The variables that were acquired consist of the action of the 12 motors plus 3 inputs from the IMU sensor with which the Euler angles are calculated, these are the data that we use to train our robot. Once the database to train our neural network was obtained, we proceeded to use the tensorflow framework to create our model.

2.2 ANN Architecture

An exploration was carried out to find out what type of neural network architecture is suitable to perform the balancing control task on the robot and to show optimal learning of the PID controller, so we started by using a multilayer perceptron network and experiments were carried out to determine the best architecture to successfully perform the balancing task.

The experiments consisted of training with the same database a series of neural networks ranging from one neuron to 20 neurons, as well as experimenting with one and two layers in each architecture. The activation function "ReLu" was used and as a loss function, we used the "mean squared error" which is used for regression problems, all training was performed with 50 epochs.

Once the experiments are done, we select the best architecture by calculating the mean absolute error and mean squared error metrics. With these metrics we can select how many neurons and how many layers are the optimal architecture to be able to perform the learning work.

2.3 Transfer Knowledge

Once the neural network had been designed and tested, we proceeded to transfer the knowledge acquired by the model to our robotic platform. However, we must take care of certain details so that the transfer can work, and the model can perform the balancing actions on a robot whose mathematical model is unknown and where none of the conventional methods such as the calculation of trajectories for the limbs are applied.

To achieve this task, we first need to use a sensor that provides us with information about the change of state in the Euler angles, i.e. a gyroscope and accelerometer, since with the appropriate software this information can be

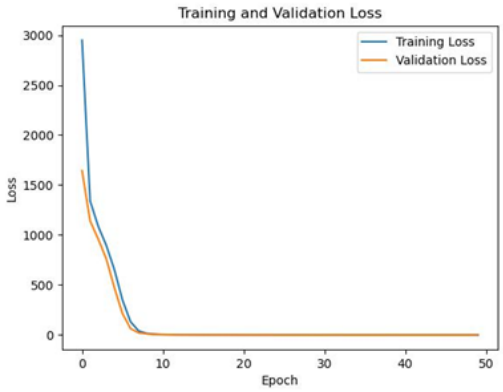

Fig. 6. Plot of the mean squared error loss function during training for the neural network with 1 layer and 18 neurons

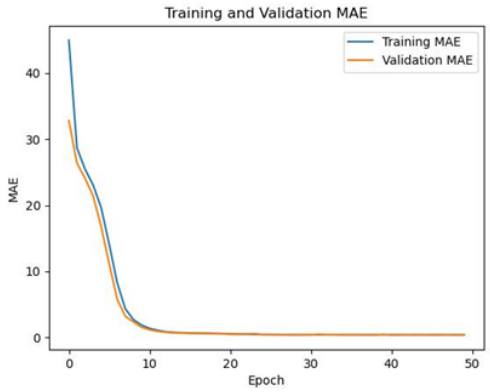

Fig. 7. Plot of the mean absolute error metric for the neura network with 1 layer and 18 neurons

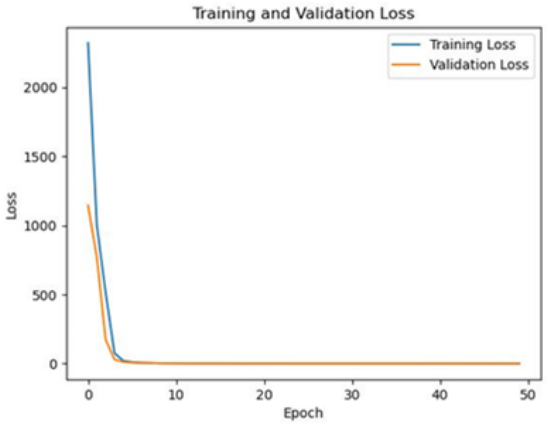

Fig. 8. Plot of the mean squared error loss function during training for the neural network with 2 layers and 14 neurons each layer

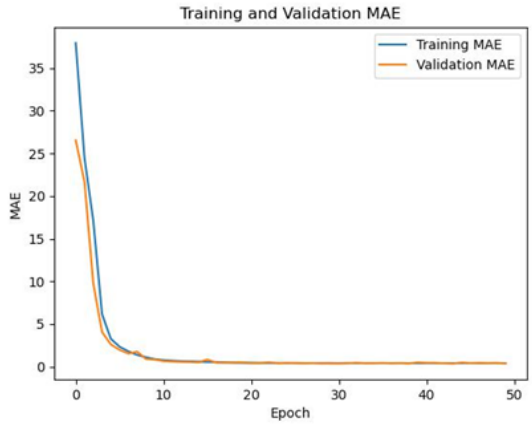

Fig. 9. Plot of the mean absolute error metric for the neura network with 2 layers and 14 neurons each layer

generated. In addition, we must consider the directions of the axes of this sensor, since the variables must have the same reference point as the sensor used to generate the database, as shown in figure 3.

We must also clarify that the IMU sensors of the new robot are not located in the same part of the robot torso from which the knowledge was extracted. However, they have a similar positioning, this impacts on the way the robot is balanced in the horizontal plane, but this problem can be solved by compensating the motors manually, it should be noted that the calibration process could be automated.

An important aspect is the assembly process of the new robot, as this must be such that the motors are assembled when they are at a predefined value of 90º, this allows the robot to be assembled as shown in figure 4, so it would be assembled in the same way as the Freenove robotic platform.

In addition, the neural network output gives values in degrees (°), so the motors of our robot should work in the same way. Our robotic platform has embedded systems to be able to run an

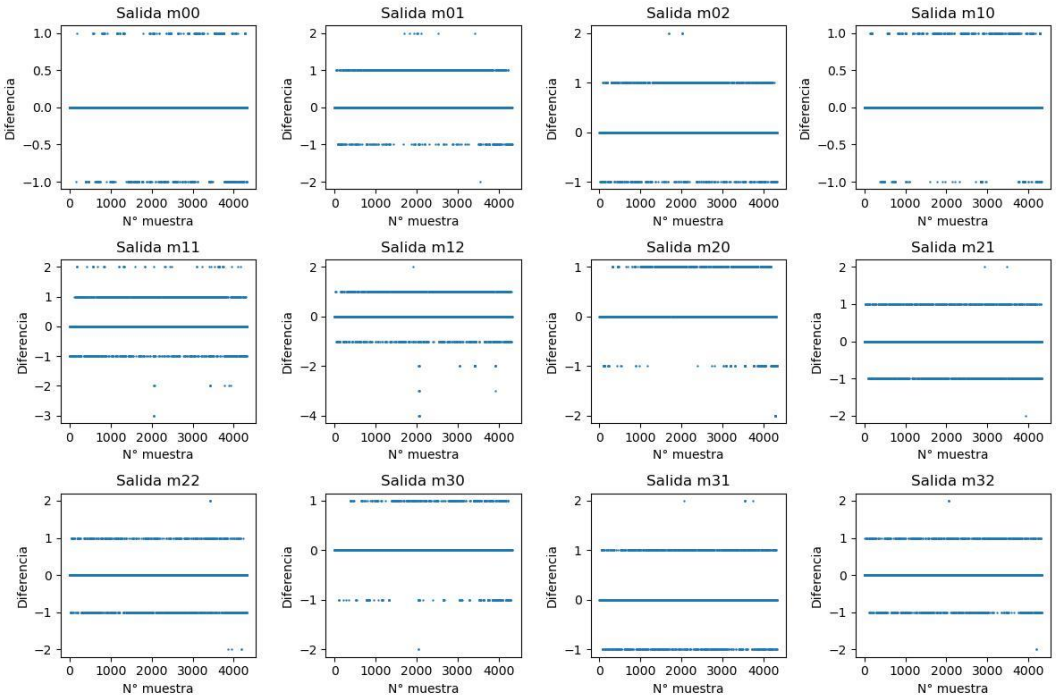

Fig. 10. Difference plot between the actual value expressed in degrees (°) and the prediction of the 1-layer model with 18 neurons

Ubuntu operating system, the problem we faced is that they do not have much support for Tensorflow so we had to convert the previously generated model to a version that occupies less computational resources such as Tensorflow Lite.

Once we installed all the dependencies in our robot, we proceeded to generate the code that extracts the information from the sensor and applies Kalman filters to obtain an accurate measurement of the angles (Roll, Pitch, Yaw). Figure 5 shows the data acquired from the sensor, as well as the response of the neural network to these readings in the new robot. Once the system had been tested, the balance control was run on the new platform.

3 Results

In this section we present the results of the search for the best neural network architecture, as well as its comparison and implementation, in addition we present graphs of the training for the best architectures and, finally, the model implemented on the new quadruped platform.

Table 1 shows the results of the metrics calculated with the validation data for the exploration of a 1-layer neural network with different number of neurons. A neural network with 18 neurons is observed to have better performance, with an MSE value of 0.26062 and a MAE of 0.2524.

Table 2 shows the results of the metrics calculated with the validation data for the exploration of a 2-layer neural network with different number of neurons. It is observed that the architecture with 14 neurons has better performance, with an MSE value of 0.2363 and an MAE of 0.2399.

Figure 6 shows the evolution of the loss function during the training of the 1-layer neural network with 18 neurons, in this case it is the average of the quadratic error. We can observe that the neural network converges approximately to the values of 10 epochs.

ISSN 2007-9737

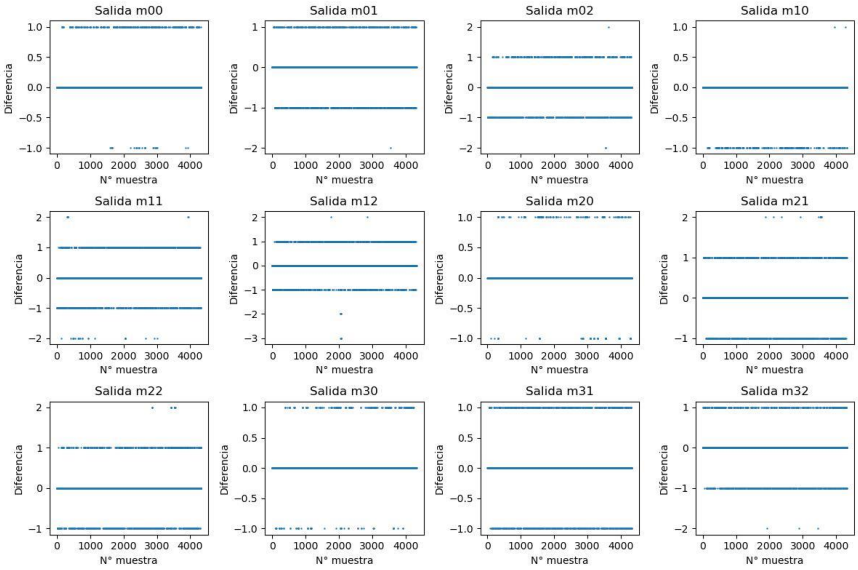

Fig. 11. Difference graph between the actual value expressed in degrees (°) and the prediction of the 2-layer model with 14 neurons each

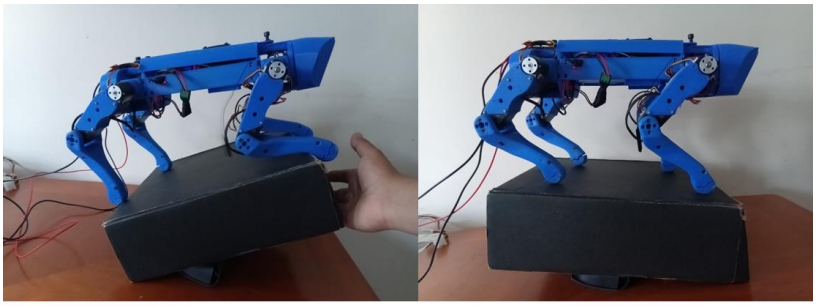

Fig. 12. Balance test on the new platform using the developed neural network

Furthermore, the graph was plotted with an additional metric "mean squared error". In figure 7 we can see how the model converges approximately at epoch 12.

Figure 8 shows the evolution of the loss function during the training of the 2-layer neural network with 14 neurons per layer, in this case it is the average of the quadratic error. We can observe that the neural network converges approximately at the values of 10 epochs.

On the other hand, the graph was also plotted with an additional metric "mean squared error". In figure 9 we can see how the model converges approximately at 20 epochs.

Figure 10 shows the error graph in the validation of the single-layer neural network with 18 neurons, where we can observe 12 sub graphs corresponding to each of the motors of the quadruped robot.

These graphs represent the error of the neural network when making an inference and it is compared with the real value that is in the dataset, and it is observed that the error of inference for the validation data is in a range of +-2° degrees.

Figure 11 shows the error graph in the validation of the 2-layer neural network with 14 neurons, where we can observe 12 sub graphs corresponding to each of the motors of the quadruped robot.

These graphs represent the error of the neural network when making an inference and it is compared with the real value that is in the dataset, and it is observed that the error of inference for the validation data is in a range of +-2° degrees.

A single-layer architecture with 18 neurons was selected to keep the model simple, as it will be used on low-resource hardware. In addition, validation and commissioning was performed on the robot, using the Tensorflow 2.0 library to load the model that we previously designed and trained.

Once the system was tested, we proceeded to run the balance control on the new platform. In figure 12 we can see the response of the front limbs to the tilt of the platform in such a way that only the Pitch angle is affected. The reaction of the model is fast and there are no jumps in the values of the inferences, so that a controlled and light movement can be appreciated.

4 Conclusion

This work showed the methodology for transferring existing knowledge from one commercial quadruped robot to another quadruped robot designed from scratch. The first robot was designed by the company Freenove, it can walk forwards, backwards and sideways, by means of classical control techniques such as calculating a trajectory that the robot limbs must follow and, as a result, it manages to generate walking patterns.

This robot has a function called balancing, which has the purpose to always keep the robot torso horizontal. In this work it was possible to obtain a consistent database of the controller behaviour used for balance task and the measurements of the sensors (gyroscope and accelerometer) that the robot has.

At the same time, a comparison of various neural network architectures was made by training with the acquired database. Hence, the optimum number of neurons and layers for the neural network was identified, this achievement gives to neural network better performance, with a very low error rate.

With the above described, it was possible to transfer this knowledge to another quadruped robot platform taking some considerations such as the inertial reference axes and the reference axes of the motors, with these considerations the quadruped robot, without knowledge of the dynamic model or its morphology, managed to imitate the balance behaviour of the Freenove robot, having a good performance when executing the task, although the motors had to be calibrated manually, but it is a step that could later be automated.

Acknowledgments

The authors of this work are grateful to the Benemérita Universidad Autónoma de Puebla and the Technological University Dublin for all the support provided, especially to the Facultad de Ciencias de la Computación and School of Enterprise Computing and Digital Transformation, as well as to the VIEP, for funding this project. We also thank the Consejo Nacional de Humanidades, Ciencia y Tecnologías (CONAHCYT) for their support through a national grant.

References

1. **Cruz-Ulloa, C., del-Cerro, J., Barrientos, A. (2023).** Mixed-reality for quadruped-robotic guidance in SAR tasks. Journal of Computational Design and Engineering, Vol. 10, No. 4, pp. 1479–1489. DOI: 10.1093/jcde/qwad061.
2. **Li, N., Cao, J., Huang, Y. (2023).** Fabrication and testing of the rescue quadruped robot for post-disaster search and rescue operations. 2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information pp. 723–729. DOI: 10.1109/ICETCI57876.2023.10176723.
3. **Chen, G., Qiao, L., Wang, B., Richter, L., Ji, A. (2022).** Bionic design of multi-toe quadruped robot for planetary surface exploration. Machines, Vol. 10, No. 10, p. 827. DOI: 10.3390/machines10100827.
4. **Arm, P., Zenkl, R., Barton, P., Beglinger, L., Dietsche, A., Ferrazzini, L., Hutter, M. (2019).** Spacebok: A dynamic legged robot for space exploration. 2019 international conference on robotics and automation, pp. 6288–6294. DOI: 10.1109/ICRA.2019.8794136.

5. **Hu, X., He, F., Xiao, P., Wang, T., Zhang, D., Zhou, X., Fan, Y. (2021).** Design of a quadruped inspection robot used in substation. 2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference, Vol. 4, pp. 766–769. DOI: 10.1109/IMCEC51613.2021.9482003.
6. **Gehring, C., Fankhauser, P., Isler, L., Diethelm, R., Bachmann, S., Potz, M., Hutter, M. (2021).** ANYmal in the field: Solving industrial inspection of an offshore HVDC platform with a quadrupedal robot. Field and Service Robotics: Results of the 12th International Conference, pp. 247–260. DOI: 10.1007/978-981-15-9460-1_18.
7. **Islam, R., Anavatti, S., Kasmarik, K., Garratt, M. (2024).** Robust and scalable swarming of quadruped robots for chemical, biological, radiological, and nuclear source localisation. 2024 10th International Conference on Mechatronics and Robotics Engineering, pp. 56–62. DOI: 10.1109/ICMRE60776.2024.10532170.
8. **Ding, X., Chen, H. (2016).** Dynamic modeling and locomotion control for quadruped robots based on center of inertia on SE (3). Journal of Dynamic Systems, Measurement, and Control, Vol. 138, No. 1, pp. 1–9. DOI: 10.1115/1.4031728.
9. **Parra Ricaurte, E. A., Pareja, J., Dominguez, S., Rossi, C. (2022).** Comparison of leg dynamic models for quadrupedal robots with compliant backbone. Scientific Reports, Vol. 12, No. 1, pp. 14579. DOI: 10.1038/s41598-022-18536-7.
10. **Kimura, H., Shimoyama, I., Miura, H. (1989).** Dynamics in the dynamic walk of a quadruped robot. Advanced Robotics, Vol. 4, No. 3, pp. 283–301. DOI: 10.1163/156855390X00305.
11. **Katayama, S., Murooka, M., Tazaki, Y. (2023).** Model predictive control of legged and humanoid robots: models and algorithms. Advanced Robotics, Vol. 37, No. 5, pp. 298–315. DOI: 10.1080/01691864.2023.2168134.
12. **Shi, Y., Wang, P., Li, M., Wang, X., Jiang, Z., Li, Z. (2019).** Model predictive control for motion planning of quadrupedal locomotion. 2019 IEEE 4th international conference on advanced robotics and mechatronics, pp. 87–92. DOI: 10.1109/ICARM.2019.8834241.
13. **Ding, Y., Pandala, A., Park, H. W. (2019).** Real-time model predictive control for versatile dynamic motions in quadrupedal robots. 2019 International Conference on Robotics and Automation pp. 8484–8490. DOI: 10.1109/ICRA.2019.8793669.
14. **Akbas, T., Eskimez, S. E., Ozel, S., Adak, O. K., Fidan, K. C., Erbatur, K. (2012).** Zero moment point based pace reference generation for quadruped robots via preview control. 2012 12th IEEE International Workshop on Advanced Motion Control, pp. 1–7. DOI: 10.1109/AMC.2012.6197116.
15. **Bellicoso, C. D., Jenelten, F., Fankhauser, P., Gehring, C., Hwangbo, J., Hutter, M. (2017).** Dynamic locomotion and whole-body control for quadrupedal robots. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3359–3365. DOI: 10.1109/IROS.2017.8206174.
16. **Gehring, C., Coros, S., Hutter, M., Bloesch, M., Hoepflinger, M. A., Siegwart, R. (2013).** Control of dynamic gaits for a quadrupedal robot. 2013 IEEE international conference on Robotics and automation. pp. 3287–3292. DOI: 10.1109/ICRA.2013.6631035.
17. **Han, K. C., Kim, J. Y. (2023).** Posture stabilizing control of quadruped robot based on cart-inverted pendulum model. Intelligent Service Robotics, Vol. 16, No. 5, pp. 521–536. DOI: 10.1007/s11370-023-00480-8.
18. **Bucher, D., Haspel, G., Golowasch, J., Nadim, F. (2015).** Central pattern generators. Encyclopedia of Neuroscience, Wiley, pp. 691–700. DOI: 10.1002/9780470015902.a0000032.pub2.
19. **Bellegarda, G., Ijspeert, A. (2022).** CPG-RL: Learning central pattern generators for quadruped locomotion. IEEE Robotics and Automation Letters, Vol. 7, No. 4, pp. 12547–12554. DOI: 10.1109/LRA.2022.3218167.
20. **Wang, J., Hu, C., Zhu, Y. (2021).** CPG-based hierarchical locomotion control for modular quadrupedal robots using deep reinforcement learning. IEEE Robotics and Automation

Letters, Vol. 6, No. 4, pp. 7193–7200. DOI: 10.1109/LRA.2021.3092647.

21. **Bahçeci, B., Erbatur, K. (2023).** Balance and posture control of legged robots: A survey. Journal of Intelligent & Robotic Systems, Vol. 108, No. 2, p. 27. DOI: 10.1007/s10846-023-01882-7.

Article received on 23/04/2024; accepted on 23/06/2024.
**Corresponding author is Francisco José López-Cortés.*