Bipedal Robot Walking and Balancing using a Neuronal Network Model

Robert Hercus[1], Litt-Pang Hiew[1], Nur'Ain Saaidon[1], Kit-Yee Wong[1], and Kim-Fong Ho[1]
[1]Neuramatix Sdn. Bhd., Mid-Valley City, Kuala Lumpur, Malaysia

Abstract - *This paper presents an alternative approach for controlling the walking and balancing of a bipedal robot. The proposed method uses a neuronal network to learn the sensor events obtained via the force sensors and accelerometer and to control the motor events of Bioloid's Dynamixel motors, to walk and balance the bipedal robot. A neuron layer called the controller network links the sensor neuron events to the motor neurons. The proposed neuronal network model (NNM) has demonstrated its ability to successfully control the walking and balancing of a bipedal robot, in the absence of a dynamic model and theoretical control methods.*

Keywords: Neuronal network, bipedal robot, balancing, control

1 Introduction

Research in the field of humanoid robotics has received great attention during recent years. Apart from their potential use in the development of prosthetics and rehabilitation devices, humanoid robotics is also being studied with the intention of creating humanoid robots which are able to interact with humans and assist them in everyday tasks. One of the advantages of humanoid robots lie in their human-like structures which allow them to move in areas that are normally inaccessible to wheeled robots, such as stairs, making them suitable for assisting the sick and elderly, as well as aiding humans in dangerous tasks and exploration missions. As such, much focus has been placed on the study of bipedal walking robots.

The motion of a bipedal walking robot can be categorized into the single support phase (with one foot on the ground), double support phase (with two feet on the ground) and the transition phase. In ordinary human gait, the length of the double support phase lasts for approximately only 20% of the step cycle [1], hence there exists a major challenge in generating a stable bipedal gait to prevent the humanoid robot from falling. In general, there are two types of bipedal gaits: static and dynamic walking. Static walking assumes that all dynamic forces produced by the motion of the robot limbs are negligible compared to the gravitational forces on the robot, therefore although it is easier to implement, the resulting gait can be unacceptably slow, with individual steps taking several seconds [2]. In dynamic walking, posture control based on dynamic generalizations of the concept of center of mass (CoM), such as the zero moment point (ZMP) [3], center of pressure (CoP) and the FRI [4], are used for generating stable bipedal gaits. The ZMP, originally introduced in published literature [5], is defined as the point on the ground where the total moment generated due to gravity and inertia equals zero. The ZMP is calculated and manipulated so that it remains in the support polygon in order for the robot to be dynamically stable and not fall. For stable walking, the ZMP of the robot must follow the desired ZMP trajectory estimated based on the desired configuration of the robot. A similar concept, the CoP, defines the point on the ground where the resultant ground-reaction forces act. Likewise, it is calculated and manipulated so that it does not reach the edge of the support polygon (or foot, in the case of the single support phase) to prevent the bipedal robot from falling. ZMP and CoP points have been shown to coincide despite the difference in their core definitions [4, 6], as long as all the contact points occur on a single plane.

The ZMP is prominently used for gait planning of bipedal humanoid robots, and it is the stability control used in ASIMO [3], a 26-DOF humanoid robot developed by Honda Motor Company in 2000. In addition to [3], there are many other researchers such as Vukobratovic et al. [7], Shih et al. [8], and Dasgupta and Nakamura [9] who proposed methods for robot walking pattern synthesis based on the ZMP, and they have all successfully demonstrated walking motions either using real robots or simulations. The importance of coordination between the hips and ankles for balancing have also been highlighted in [10,11], where the hip-ankle strategy was demonstrated to exhibit better balancing performance compared to the algorithms employing only the ankle strategy (rotating the ankles while locking the knee and hip joints) for push recovery [12,13].

In terms of intelligent control methods, many methods have been proposed and published. In [14], researchers have proposed a genetic-fuzzy controller for biped robots in which the dynamic stability of two-legged robots climbing staircases is simulated. Researchers [15] have also employed fuzzy logic to determine effective walking control of biped robots where two different fuzzy controllers for the support leg and the swing leg are described. The other noteworthy published works involve the application of a neuro-fuzzy algorithm [16], adaptive neuro-fuzzy algorithm [17], supervised learning using neural network [18] and a back-propagation artificial neural network as the learning scheme in [19]. The neural network with back propagation [19], neuro-fuzzy [20], and SVM [21] methods are popular in this research discipline because they can perform regression between the input and the output (error between measured ZMP and ZMP trajectory) in the supervised learning manner. The RNN [22] and CMAC [23] methods have showed that unsupervised learning can be applied in a bipedal walking robot, but these methods are typically used as part of a hybrid controller together with a compensated torque controller (PID type), and are aimed at modeling and then compensating the system's faults, uncertainties or environmental disturbances.

2 Neuronal Network Model

The Neuronal Network Model (NNM) is a multi-node, multi-level, and multi-network neuronal network which is based on the concept of how the brain is thought to work [24]. The basic unit in a network is an elementary neuron, which represents a sensor or motor event. These neurons can be associated spatially or temporally to represent sequences of sensor or motor events. The NNM concept has been applied in the balancing control of an inverted pendulum [25,26], trajectory control of a muscle-actuated manipulator [27] as well as the navigation control of a UAV [28]. A software implementation of the NNM, the NeuraBASE toolbox, is available for download at [29].

The proposed NNM controller for the bipedal walking robot consists of three distinct and fundamental network types, namely a) sensor neurons and events - inputs to the system (CoP values for the feet, bipedal tilt direction, observed joint angles); b) motor neurons - outputs from the system (joint rotations); c) interneurons - association between two sensor events (to form a linked sensor neuron), or between a sensor event and a motor action (to form a controller neuron). Each type of event builds up an association of events in their respective network. The sensor network and motor network store sensor neurons and events, motor neurons and events, and interneurons associations respectively. A simplified data structure of the neurons used in the NNM is described in Table 1. More detailed descriptions of the sensor, motor and controller neurons are provided in Section 3.

Table 1: Data Structure of a Neuron (basic)

Field	Data Type
Head	unsigned int
Tail	unsigned int
Successor	unsigned int
Frequency/ Weight*	signed int
Next	unsigned int
Overshoot/Undershoot Flags*	unsigned short

* Denotes Fields only applicable to the Controller Neuron

3 Experimental Setup

The bipedal walking robot system hardware is made up of multiple joint structures, which imitate the motion of human legs whereby each joint is controlled by a Dynamixel servomotor. The current angular position and speed of the Dynamixel motor are accessible via the CM501 controller that comes with the Bioloid Robot Kit. Figure 1 depicts the partially assembled Bioloid robot. Each foot of the robot is attached to a force pad sensor, which measures the CoP (center-of-pressure) point. An accelerometer is also attached to the torso of the Bioloid to detect the falling state.

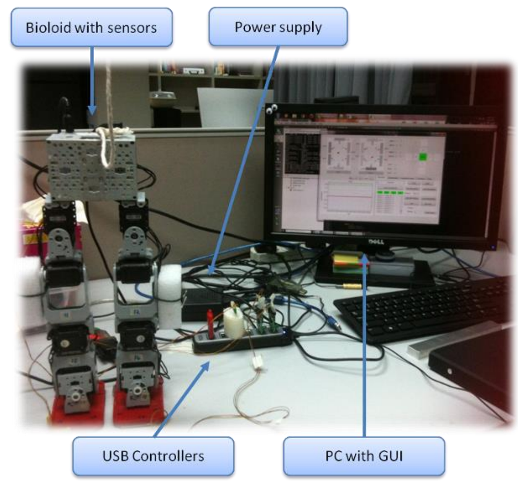


Figure 1: The partially assembled Bioloid Robot and its mechatronic components.

The walking gait cycle is a sequence of postures that describes how the bipedal robot should walk (see Table 2). The ideal posture for a stable walking gait of the bipedal robot has been developed as a reference for training. In this work, the NNM controller only controls selected motors throughout the gait cycle of the bipedal robot. The joint assignments for the Bioloid are shown in Table 3.

Table 2: Walking gait of the Bioloid

Dyna	Sub-Step						
	Starting Position	1/7	2	3	4	5	6
Front view							
Side view							
Process		weight shifting (dual support)	weight shifting (single support)	linear extension	weight shifting (dual support)	weight shifting (single support)	linear extension
Target		CoP LX	CoP LX	right leg fully extends	CoP RX	CoP RX	left leg fully extends

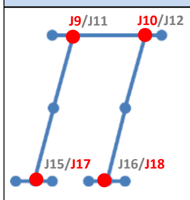
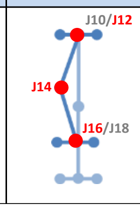
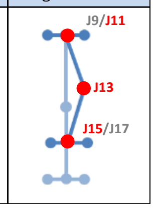
● NB controlled joint ● Joint with pre-set angular position

Walking is a repetitive motion consisting of the following basic phases that alternate on each leg:

- *Dual Support (Weight Shifting):* In this phase, both feet must be on the ground while the ankle and hip joints will move in a parallelogram configuration to shift the weight of the biped sideways until the body weight is concentrated on one foot.
- *Single support (Balancing):* Only one foot is in full contact with the ground and this foot will support the full weight of the bipedal robot. While the other leg is lifted off the ground, the supporting leg needs to ensure the stability of the structure.

- *Linear Extension:* Once the biped has successfully lifted its foot and tilted forward, its knee needs to be extended a certain amount to return the structure to a parallelogram state, thus preparing it for the start of a new walking cycle.

Table 3: Joint assignments for the Bioloid

Front view	Side view (left) - left foot lifted	Side view (right) - right foot lifted
 J9/J11 J10/J12 J15/J17 J16/J18	 J10/J12 J14 J16/J18	 J9/J11 J13 J15/J17

There are six main sub-steps/postures involved in the balancing process, and they are weight shifting processes (sub-steps 1, 2, 4 and 5) on single planes and linear extend process (sub-steps 3 and 6). During each step, the NNM controller will control specific sets of motors, and the other motors will be set with ideal pre-set angular positions.

3.1 Training Controller

The controller program for Bioloid training acquires the input value from three sensors: readings from the two force pad sensors and readings from the accelerometer. The output of the controller program is the angular increment of the Dynamixel motor, which can be retrieved, via the NNM controller's prediction or by using a pre-set value. The program also includes the graphical user interface (GUI) to monitor the input signal from the sensors throughout the training session. The readings from the force pad sensors are used to compute the CoP values and the readings from the accelerometer will be used to detect the falling state of the bipedal robot. While the controller program is executing the gait cycle, the NNM controller will save the sequence of CoP values and return the angular increment to the respective motors as the executed output. The controller program was designed using a multithreaded programming method as timing is very crucial for the control system.

Figure 2 shows the data flow in the controller program code. Each sensor has its own thread where each thread will request six sets of 2-byte data from its respective sensor device at every ~5ms interval and then segment it accordingly for future use in the program. Then, the sensor sequence refresh thread will update all segmented sensor values from the sensor thread and rearrange the sensor sequence for the NNM controller every 50ms. The main thread will also take the sensor values from the sensor thread for data monitoring and plotting purposes. The training thread will activate the NNM controller thread which will update its own sensor sequence at its chosen time interval, evaluate the sensor sequence and then give predictions in the form of serial commands to specific Dynamixel motors controlled by its network.

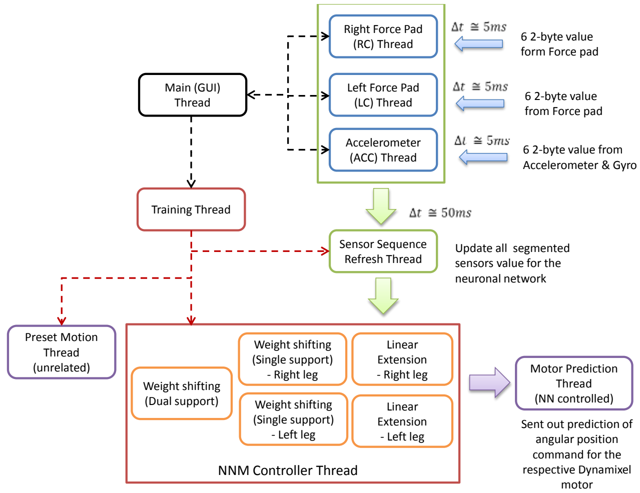


Figure 2: System data flow chart

3.2 NNM Controller

The neuronal network architecture of the robot's weight-shifting phase is depicted in Figure 3. There are three sensor networks: two for building/storing the sequence of CoP values and the bipedal robot's tilt direction, and a third network for the CoP target. The events of these sensor networks are connected in the interneuron networks, which act as the head of the controller neurons. The CoP sensor networks A and B keep the sensor events, which are the observable variables while the target CoP network keeps the sensor events that represent the target positions of the CoP value. The network that connects these sensor networks together and provides the sensor neuron to the controller network is called the CoP-state Interneuron network.

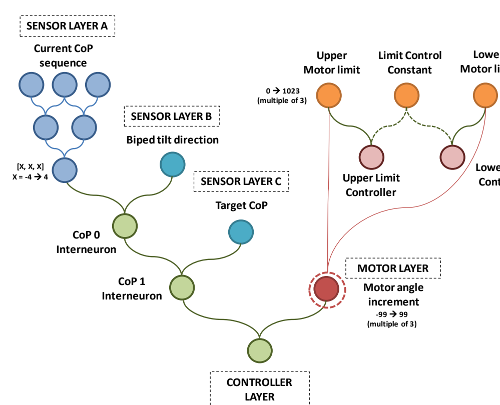


Figure 3: The neuronal network architecture for the weight shifting phase

In the motor layer, the angle increment network will keep the motor events that respond to the observed sensor events and required target positions. The motor neurons consist of angular increments ranging from -99 units to 99 degree units, and the frequency/strength of controller neurons is capped at 10 and -10. However, an additional network of angle boundary controls the selection of the motor neuron. The angle boundary network keeps the maximum and minimum angle limit of the respective group of motors that will be updated only when the biped falls. The falling event of the biped may result from the extreme angle issued to the motor that is unsuitable for the balancing process. Thus, elimination of the extreme angle in the movement's history buffer will help to keep the biped within an acceptable angle range instead of executing excessive turns. The controller layer maintains the controller events connecting the sensor events to the motor events.

Table 4: NNM Controller I/O for the Weight Shifting Controller

Input(s)	Output
CoP sequence (historical)	Angle increment for joint group 1 (joints 9,10,17,18)
Current parallelogram tilt direction	
Target CoP	

For the linear extension phase as referred to in Figure 4, the network architecture is much simpler, given that the target is to merely extend the leg fully. The controller event will connect the sensor event of current joint angle and the motor event of angle increment needed in order to extend the lifted leg. Eventually, the biped will know how much extension is required to straighten the leg given the initial position of the leg in order to touch the ground before the start of the next phase, which is the weight shifting phase.

All NNM controllers have near-identical algorithms, and the only difference between them is the choice of sensors, the complexity of the NNM network (see Tables 4 and 5), and the activation duration which solely depends on the phase of the walking cycle. Each phase will activate its own NNM controller(s) which will process their respective sensor inputs and issue their angle increment commands as outputs of the controller.

After the initialization of the neuronal networks that will be used in that particular controller thread, the program will execute a loop of prediction and feedback processes (see Figure 5). However, the loop will only proceed to the prediction process if the controller status is active, meaning it is in the correct walking phase and the prediction status is idle. When both conditions are satisfied, it will refresh its target sensor value, and get the current sensor sequence and current angular position and polarity of its respective Dynamixel motor. The current CoP sequence will then be evaluated with respect to the target CoP position to determine if the motor requires a prediction value to move. If the average of the CoP sequence is not within the target position, the prediction process will be executed, in which it searches for the longest match sensor neuron and its respective controller neuron, in order to get the controller neuron with the highest frequency and retrieve the motor neuron for the predicted angle increment value. However, the increment value will undergo a process of verification within the angle boundary network. Increments resulting in an angle exceeding the angle limits will not be issued, in which case the angle increment will be saturated. If there is no suitable controller neuron above the threshold, a new random value will be generated and issued to the respective motors.

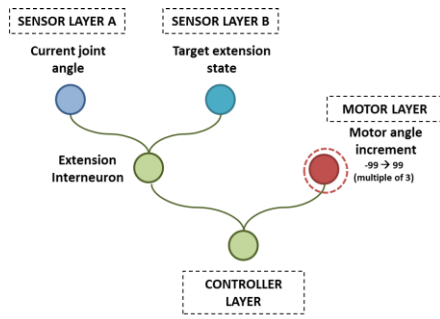


Figure 4: The neuronal network architecture for the linear extension phase

Table 5: NNM Controller I/O for the Linear Extension Controller

Input(s)	Output
Current hip joint angle	Angle increment for joint group 2 (joints 11,13,15) / joint group 3 (joints 12,14,16)
Target extension state (0 or 1)	

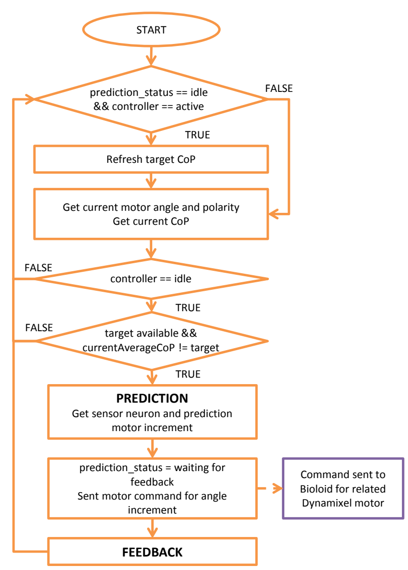


Figure 5: The general flow chart for each NNM Controller

Once the angle increment has been issued, the controller thread will awaken at an interval of 50ms to check the status of the execution. The feedback process will only be executed if the biped has stopped moving or the sensor values have stopped oscillating due to mechanical vibrations. No feedback is given if the target has not been reached but the biped does not fall. A positive feedback will be given if it reaches the target CoP region, whereas a negative feedback will be given when the biped falls.

During the feedback process, the actual motor neuron from the prediction process may not be rewarded/penalized; instead a motor neuron representing *the difference between the current motor angle and the motor angle when the prediction was first executed* will be linked to the sensor neuron. However, it is noted that since the motor's speeds will be adjusted according to its angle increment bracket, therefore by learning these motor events, the NNM controllers will eventually begin to issue smaller motor commands when the sensor values are close to the target, leading to less oscillations in the biped's movements.

Referring to Figure 6, a reward or penalty will only be given when the average CoP value has reached the target region or when the biped falls. All of the sensor neurons in the storage buffer will be connected to the motor neuron that represents the angle difference between the current angle and its angle before the prediction.

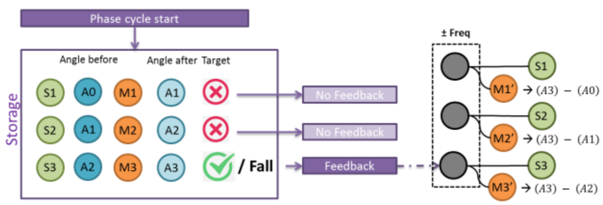


Figure 6: NNM Controller feedback reward/penalty concept

4 Experimental Results

4.1 Experimental Setup

The biped is trained to maintain its balance while walking by targeting different Center-of-Pressure points (CoPs) at different sub-steps. The forward motion (along the y-axis) is preset, whereas the process of balancing and weight shifting along the x-axis are controlled using the NNM controller. The reward/penalty system for all controller neurons is designed as shown in Table 6. The different NNM controllers activated during different points of a walking cycle (Figure 7) are shown in Table 7. Note that sub-steps 1 and 7 have identical targets, and differ only in the biped's initial pose, where the biped stands upright at the beginning of sub-step 1 (at the start of the training program) whereas it is greatly tilted towards its right side at the beginning of sub-step 7 (at the end of sub-step 6).

As all the controllers in Table 7 give predictions in the form of angle increments, this could lead to continuous increment recommendations in case of improper foot placements which may possibly affect the force sensor readings, resulting in final joint angles that may cause the structure to topple over. Therefore, additional NNM controllers have also been added for each of the three support phase controllers (Single Support (Left), Single Support (Right), and Double Support Controllers) to control the absolute joint limits, and any angle increment recommendations commanding the joints to rotate beyond these limits will then be capped accordingly. Since the biped's joint angles are acquired every ~50ms, this allows the current motor prediction to be overridden in case the target CoP has been overshot. However, as the joint servos have been programmed to slow down when the observed joint angle is close to the commanded joint angle, a mere change in direction may result in heavy oscillations or failure, thereby resulting in longer training times. Therefore, the role of the NNM controller would be to recommend smaller angle increments in order to prevent these overshoots. In order to speed up training, biased random predictions will now be given, whereby a random angle increment will be recommended according to the target CoP relative to the current CoP, as opposed to giving random predictions in either direction. Training was conducted for approximately 200 walking trials (~7 hours) and was terminated after the biped managed to take 20 steps (10 walking cycles) without falling.

Table 6: The reward and penalty system for all Controller Neurons

Type	Reward/Penalty
Success (on target)	+3
Partial success (within tolerance limit)	+1
Failure (biped falls)	-2

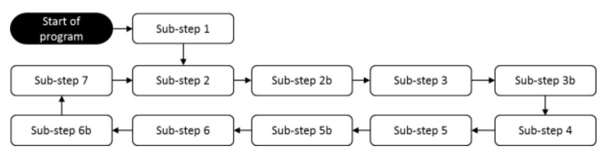


Figure 7: Flowchart showing the sub-step transitions of the training program.

4.2 Results and Discussion

Performance of the controllers is gauged by the NNM controllers' ability to learn the correct $\Delta\theta$ increments to achieve the desired CoP while learning the joint limits to prevent the biped from falling down during the process. Consequently, the success of training is determined as follows: the biped should be able to maintain its balance while moving forward for longer durations. Due to table size limitations, the maximum number of steps allowed was capped at 20 steps. Each step covers the time from when the biped needs to shift its weight in the double support phase up to the point when it puts its foot back down on the surface, e.g. sub-steps 1/7-3b and sub-steps 4-6b.

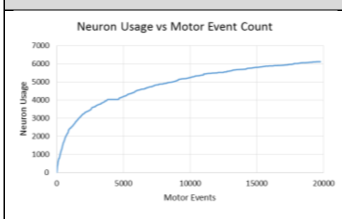
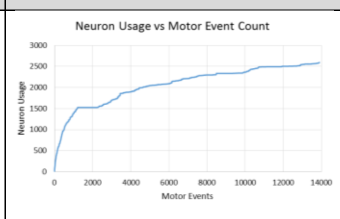
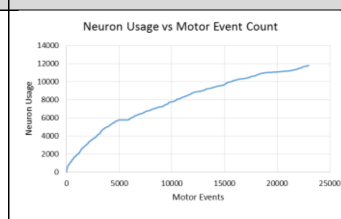
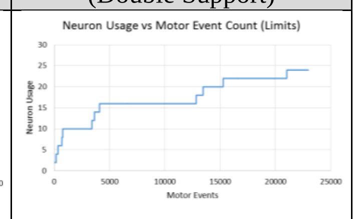
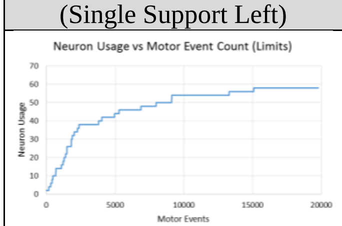
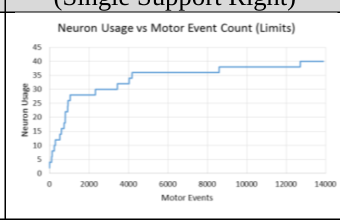
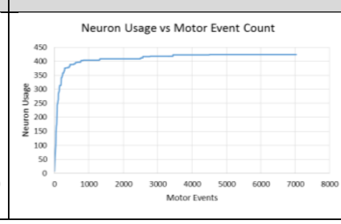
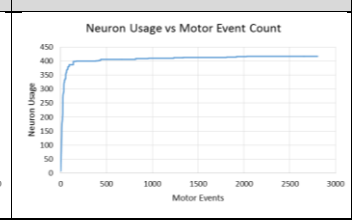
Referring to Figure 8, we observed that the average number of completed steps taken by the bipedal robot increased with training; a video is available online [30]. However, there are still some instances where the biped is unable to reach the targeted 20 forward steps, and occasionally fell during the first step. While the biped is expected to balance itself indefinitely, this does not appear to be the case.

Table 7: Table detailing the active NNM controllers, preset motions and their respective controlled joints during each sub-step in the bipedal's walking cycle

Sub-Step	**1 and 7**	**2**	**2b**	**3**	**3b**
Description	Shift weight to the *left*	Lift *right* foot while maintaining balance	Maintain balance while keeping *right* foot up	Tilt body forward	Put *right* foot down
NB Controller (1)	Double Support Controller	Single Support Controller (Left)	Single Support Controller (Left)	Single Support Controller (Left)	Linear Extension Controller (Left)
NB Controller (1) Target	CoP within range of [+1 +2]	*Left* foot CoP >= 0	*Left* foot CoP within range of [-1 0]	*Left* foot CoP within range of [-1 0]	Full extension of *right* leg. Does not use any CoP values for prediction or feedback.
Joints involved	9,10,17,18	9,10,17,18	9,10,17,18	9,10,17,18	11,13,15
NB Controller (2)					Single Support Controller (Left)
NB Controller (2) Target					*Left* foot CoP within range of [-1 1]
Joints involved					9,10,17,18
Preset motion		Foot lifting		Forward tilt	
Joints involved		11,13,15		11,12,15,16	

Sub-Step	**4**	**5**	**5b**	**6**	**6b**
Description	Shift weight to the *right*	Lift *left* foot while maintaining balance	Maintain balance while keeping *left* foot up	Tilt body forward	Put *left* foot down
NB Controller (1)	Double Support Controller	Single Support Controller (Right)	Single Support Controller (Right)	Single Support Controller (Right)	Linear Extension Controller (Right)
NB Controller (1) Target	CoP within range of [-2 -1]	*Right* foot CoP <= 0	*Right* foot CoP within range of [0 +1]	*Right* foot CoP within range of [0 +1]	Full extension of *left* leg. Does not use any CoP values for prediction or feedback.
Joints involved	9,10,17,18	9,10,17,18	9,10,17,18	9,10,17,18	12,14,16
NB Controller (2)					Single Support Controller (Right)
NB Controller (2) Target					*Right* foot CoP within range of [-1 1]
Joints involved					9,10,17,18
Preset motion		Foot lifting		Forward tilt	
Joints involved		12,14,16		11,12,15,16	

Table 8: Total Neuron Consumption vs. Number of Motor Events for Each NNM Controller in the Bipedal Walking System

Neuron Usage			
Single Support (Left)	Single Support (Right)	Double Support	Joint Limits (Double Support)
			
Joint Limits (Single Support Left)	Joint Limits (Single Support Right)	Linear Extension (Left)	Linear Extension (Right)
			

This is attributed to new CoP sequence combinations caused by either new CoP transitions arising from previous motor actions, or sensor noise from the force pads due to mechanical vibrations during the walking process. Apart from those, this could also be caused by the imperfect walking gait programmed into the system.

As shown in Table 8, the neuron usage curves for each controller is observed to have begun plateauing (with the exception of the double support phase controller), with a total neuron count of ~21,500 at the end of training. As expected, the single and double support controllers recorded the highest neuron consumption due to there being many CoP sequence permutations for training.

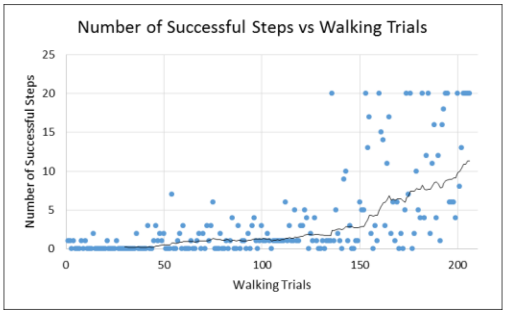


Figure 8: Number of completed steps vs. number of walking trials

5 Conclusion

The NNM controller model presented in this paper is a proof of concept that the NNM can be adapted to control the walking and balancing of a bipedal robot in the absence of dynamical models, where learning is evidenced by the increase in the number of steps successfully taken with training. However, it is also noted that this controller model is still in its early stages, and the walking gait presented in this paper is a slow and relatively stable one. Future work involves enhancing the controller model to also handle velocity-based stability control in order to make faster walking possible.

6 References

[1] Huang, Q., Yokoi, K., Kajita, S., Kaneko, K., Arai, H., Koyachi, N., Tanie, K., (2001) Planning walking patterns for a biped robot, *IEEE Transactions on Robotics and Automation*, Vol. 17, No. 3, pp. 280–289.

[2] Cheng, M.-Y., Lin, C.-S., (1995) Genetic algorithm for control design of biped locomotion, *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 1315–1320.

[3] Hirai, K., Hirose, M., Haikawa, Y., Takenaka, T., (1998) The development of the honda humanoid robot, *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 16-20 May, pp. 1321-1326.

[4] Goswami, A., (1999) Postural stability of biped robots and the foot rotation indicator (FRI) point, *International Journal of Robotics Research*, Vol, 18, pp. 523–533.

[5] Vukobratovic, M., Juricic, D., (1969) Contribution to the synthesis of biped gait, IEEE. Trans. Bio-*Med. Eng. BME*, Vol. 16, No. 1, pp. 1–6.

[6] Sardain, P., & Bessonnet, G. (2004). Forces acting on a biped robot. Center of pressure-zero moment point, *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 34(5), 630-637.

[7] Vukobratovic, M., Brovac, B., Surla, D., Stokic, S., (1990) Biped locomotion. Springer, New York.

[8] Shih, C.L., Li, Y.Z., Churng, S., Lee, T.T., Cruver, W.A., (1990) Trajectory synthesis and physical admissibility for a biped robot during the single support phase, *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 1646–1652.

[9] Dasgupta, A., Nakamura, Y., (1999) Making feasible walking motion of humanoid robots from human motion capture data, *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 1044–1049.

[10] Stephens, B., (2007) Humanoid Push Recovery, *Proc. 7th IEEE-RAS Int Humanoid Robots Conf*, pp. 589–595

[11] Pratt, J., Carff, J., Drakunov, S., Goswami, A., Capture point: A Step Toward Humanoid Push Recovery, *Proc. 6th IEEE-RAS Int Humanoid Robots Conf*, pp. 200–207

[12] Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., Hirukawa, H., (2001) The 3D Linear Inverted Pendulum Mode: a Simple Modeling for a Biped Walking Pattern Generation," *Proc. IEEE/RSJ Int Intelligent Robots and Systems Conf*, Vol. 1, pp. 239–246

[13] Ott, C., Roa, M., Hirzinger, G., (2011) Posture and Balance Control for Biped Robots Based on Contact Force Optimization, *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference* on, pp. 26 –33, Oct

[14] Jha, R.K., Singh, B., Pratihar, D.K., (2005) On-line stable gait generation of a two-legged robot using a genetic–fuzzy system, *Robot. Auton. Syst*., Vol. 53, pp. 15–35.

[15] Murakami, S., Yamamoto, E., Fujimoto, K., (1995) Fuzzy control of dynamic biped walking robot, *Proc. of the 4th IEEE International Joint Conference on Fuzzy Systems*, Vol. 1, pp. 77–82.

[16] Wongsuwarn, H., Laowattana, D., (2007) Neuro-fuzzy algorithm for a biped robotic system, *Int. J. Appl.Math. Comput. Sci.*, Vol. 3, No. 4, pp. 195–201.

[17] Shieh, M.Y., Chang, K.H., Lia, Y.S., (2008) Design of a biped locomotion controller based on adaptive neuro-fuzzy inference systems, *J. Phys. Conf. Ser*. Vol. 96, pp. 1–8.

[18] Miller, W.T., (1994) Real-time neural network control of a biped walking robot, *IEEE Control Syst. Mag*., Vol. 14, pp. 41–48.

[19] Shieh, M.Y., Chang, K.H., Chuang, C.Y., Lia, Y.S., (2007) Development and implementation of an artificial neural network based controller for gait balance of a biped robot, *Proc. 33rd Annual Conf. the IEEE Industrial Electronics Society (IECON)*, pp. 2778–2782.

[20] Kim, D., Seo, S.-J., Park, G.-T., (2005) Zero-moment point trajectory modelling of a biped walking robot using an adaptive neuro-fuzzy system, *IEE Proceedings - Control Theory and Applications*, Volume 152, issue4, July 2005, p. 411 – 426.

[21] Ferreira, J.P., Crisóstomo, M.M, Coimbra, A.P., (2009) Control of a Biped Robot With Support Vector Regression in Sagittal Plane, *IEEE Transactions of Instrumentation and Measurement*, Vol. 58, No. 9, pp. 3167-3176.

[22] Wu, Y., Song Q., Yang. X., (2007) Robust Recurrent Neural Network Control of Biped Robot, *Journal of Intelligent and Robotic Systems*, Vol. 49, No. 2, pp 151-169.

[23] Lin, C.-M., Fan, W.-C., Chen, C.-H., Hou, Y.-L., (2006) Robust Control for Biped Robot Using Cerebellar Model Articulation Controller, *International Joint Conference on Neural Networks*, July 16-21, pp. 2485-2490.

[24] R. G. Hercus, "Neural networks with learning and expression capability," U.S. Patent 7412426 B2, 2008.

[25] R. Hercus, K.-Y. Wong and K.-F. Ho, "Balancing of a Simulated Inverted Pendulum using the NeuraBase Network Model," *LNCS*, Vol. 8131, 2013, pp. 527-536.

[26] R. Hercus, K.-Y. Wong, S.-K. Shee and K.-F. Ho, "Control of an Inverted Pendulum Using the NeuraBase Network Model," ICONIP 2013, Part II, *LNCS*, Vol. 8227, 2013, pp. 605–615.

[27] R. Hercus, K.-Y. Wong and K.-F. Ho, "Control of a Muscle Actuated Manipulator using the NeuraBASE Network Model," *JOACE*, Vol. 2(3), 2014, pp. 302-309.

[28] R. Hercus, H.-S. Kong and K.-F. Ho, "Control of an Unmanned Aerial Vehicle Using a Neuronal Network," *IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB)*, Singapore, 2013, pp.73-79.

[29] NeuraBase Generic Toolbox, http://www.neuramatix.com

[30] R. Hercus, L.-P. Hiew, N.-A. Saaidon, K.-Y. Wong, and K.-F. Ho. NeuraBase – Learning of a Bipedal Robot in Walking and Balancing II: http://www.youtube.com/watch?v=Ru8Uzl8mvgc